

INTERNAL REPORTS IN
SIMULATION, OPTIMIZATION
AND CONTROL

No. SOC-292

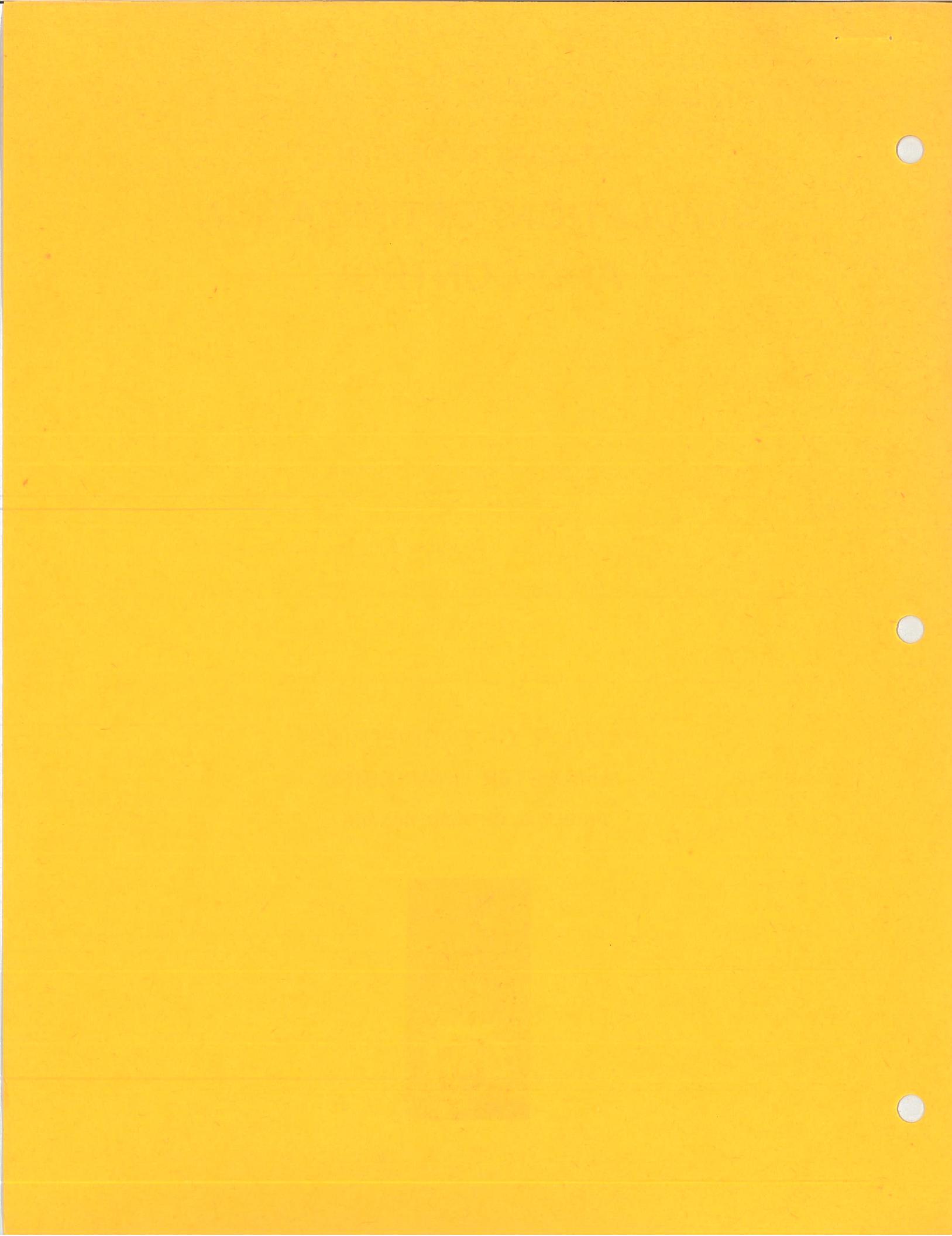
MMLC - A FORTRAN PACKAGE FOR LINEARLY CONSTRAINED MINIMAX OPTIMIZATION

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 MMA1Q 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 $f_i(\underline{x})$, $i=1,2,\dots,m$, of n variables $\underline{x} = [x_1 \ x_2 \ \dots \ x_n]^T$, it is the purpose of the package to find a local minimum of the minimax objective function

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

subject to linear constraints

$$\underline{c}_i^T \underline{x} + b_i = 0, \quad i = 1, \dots, \ell_{eq},$$

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

where \underline{c}_i and b_i , $i = 1, \dots, \ell$, 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 chosen) 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 k th iteration the change \underline{h}^k of the approximation \underline{x}^{k-1} is determined as the solution of the linear minimax problem

$$\text{Minimize}_{\underline{h}^k} \tilde{F}(\underline{x}^{k-1}, \underline{h}^k) = \max_{1 \leq i \leq m} (f_i(\underline{x}^{k-1}) + f_i^T(\underline{x}^{k-1}) \underline{h}^k)$$

subject to constraints

$$c_i^T(\underline{x}^{k-1} + \underline{h}^k) + b_i = 0, \quad i = 1, \dots, l_{eq},$$

$$\zeta_i^T (\tilde{x}^{k-1} + \tilde{h}^k) + b_i \geq 0, \quad i = \ell_{eq} + 1, \dots, \ell,$$
$$\|\tilde{h}^k\| \leq \delta_x^{k-1},$$

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

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

it is successful if

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

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

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

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

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

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

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 \underline{h}^k of the approximation \underline{x}^k of the solution is sufficiently small

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

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

- (3) the consecutive change \underline{h}^k reaches the machine accuracy

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

where ε_0 is the smallest positive number such that

$$1 + \varepsilon_0 > 1,$$

- (4) the consecutive change \underline{h}^k is insignificantly small, namely,

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

(when the solution \underline{x}^* is equal to 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

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

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).

Switch to Stage 2

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 , $k = 1, 2, \dots$, are updated in consecutive iterations, corresponding to consecutive approximations \underline{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 $F(\tilde{x}^k)$ decreases in consecutive switches to Stage 2 (for the first switch this condition is omitted)

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

where \tilde{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

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

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

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

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

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

where the unknowns are $[\tilde{h}^k, \lambda^k]$, 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 $f_j''(\tilde{x}^{k-1} + \tilde{h}^k)$ the approximated second-order derivatives are used.

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

$$r(\tilde{x}^{k-1} + \tilde{h}^k, \lambda^k, A^k) = \| \{ \lambda_j^k f'_{ji}(\tilde{x}^{k-1} + \tilde{h}^k) \mid j \in A_f^k, i = 1, 2, \dots, n \},$$

$$\{ \lambda_j^k (\tilde{c}_j^T (\tilde{x}^{k-1} + \tilde{h}^k) + b_j) \mid j \in A_c^k \},$$

$$\{ \tilde{c}_j^T (\tilde{x}^{k-1} + \tilde{h}^k) + b_j \mid j \in A_c^k \},$$

$$\{ f_j(\tilde{x}^{k-1} + \tilde{h}^k) - f_{j_0}(\tilde{x}^{k-1} + \tilde{h}^k) \mid j \in A_f^k - \{ j_0 \} \} \|$$

and if the residual decreases

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

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

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 \tilde{h}^k of the approximation \tilde{x}^k of the solution is sufficiently small

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

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

- (3) the consecutive change \tilde{h}^k reaches the machine accuracy

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

where ε_0 is the smallest positive number such that

$$1 + \varepsilon_0 > 1,$$

- (4) the consecutive change \tilde{h}^k is insignificantly small, namely,

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

(when the solution \tilde{x}^* is equal to 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 $r(x, \lambda, A)$ are decreasing

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

- (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

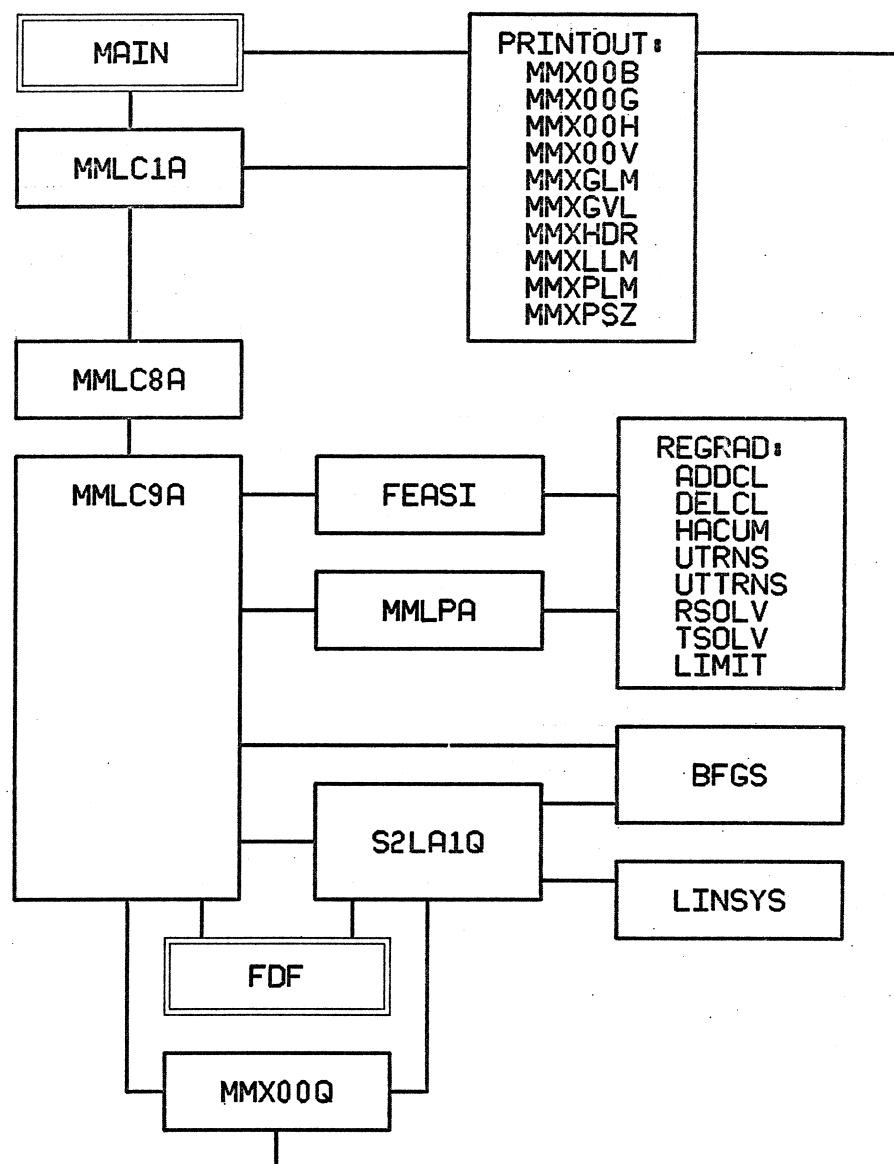


Fig. 1 Structure of the MMLC package corresponding to the standard entry (subroutine MMLC1A).

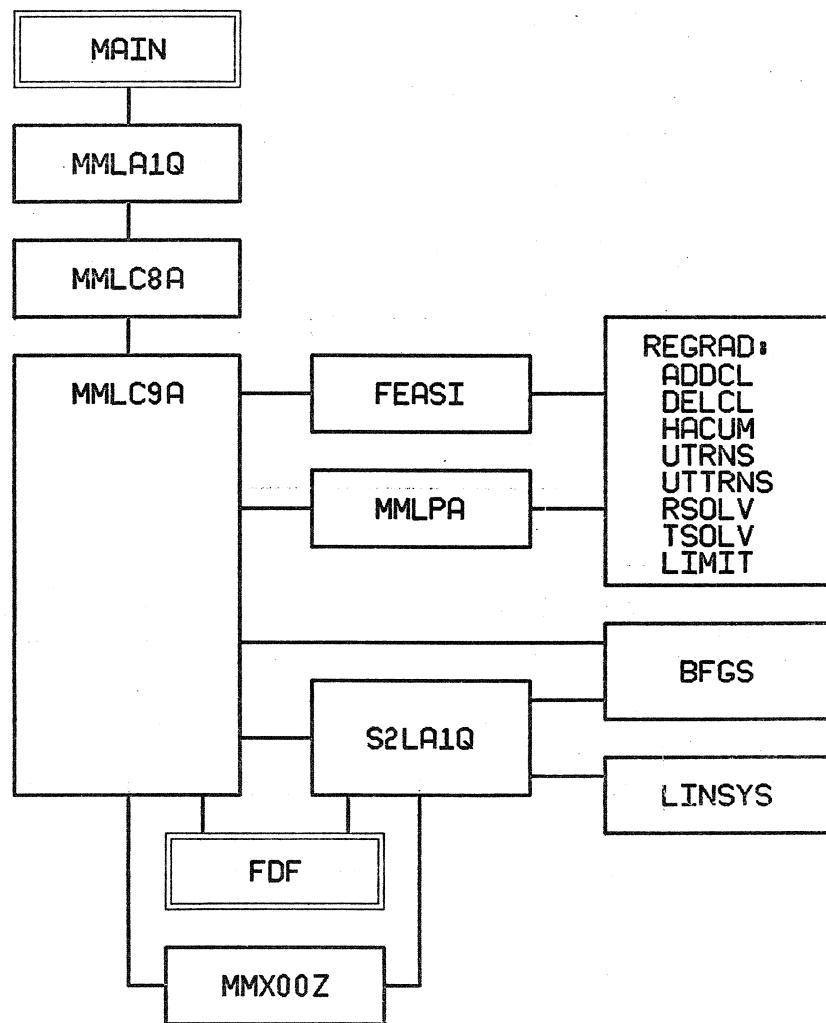


Fig. 2 Structure of the MMLC package corresponding to the original entry (subroutine MMLA1Q).

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 $f_i(x)$ and their derivatives $\partial f_i(x)/\partial x_j$ at the point x corresponding to $X(1), X(2), \dots, X(N)$, and store the values in the following way:

$$F(I) = f_I(x), \quad I=1, \dots, M,$$
$$DF(I,J) = \partial f_I(x)/\partial x_J, \quad I=1, \dots, M, \quad J=1, \dots, N.$$

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. $B(I) = b_I$, $I = 1, \dots, L$. 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 x in the linear constraints, i.e.

$$(C(I,1), C(I,2), \dots, C(I,N)) = g_I^T, \quad I = 1, \dots, L.$$

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, $X(I)=x_I^0$, $I=1, \dots, N$. On exit X contains the best solution found by the package.

DX is a REAL variable which controls the step length of the iterative algorithm. On entry 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 $\|\underline{h}^k\| \leq EPS*\|x^k\|$, where \underline{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 $>$ 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. On exit the first M elements of W contain the residual function values at the solution, i.e., $W(I)=f_I(x)$, $I=1, \dots, M$.

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

$$|IPR| = pqr s$$

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. $q=1, \dots, 9$) 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 MMIC PACKAGE FOR UNCONSTRAINED PROBLEMS

M:	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	599	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	690	823	956	1099	1252	1415	1588	1771	1964	2167	2380	2603	
11	82	127	182	247	322	407	502	607	702	822	947	1082	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 MMIC PACKAGE FOR UNCONSTRAINED PROBLEMS

M:	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	1589	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 \underline{x}^0 and corresponding residual function values $\underline{f}(\underline{x}^0)$ 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: $N > 0$, $M > 0$, $L \geq 0$, $LEQ \geq 0$, $LEQ \leq L$, $LEQ \leq N$, $LC \geq L$, $DX > 0$,
 $\text{EPS} \geq 0$, $\text{MAXF} > 0$, $\text{KEQS} > 0$, $\text{IW} \geq \text{IWR}$.

Date: April 1982.

VII. EXAMPLES

Example 1 [1, Example 1]

Minimize

$$F(x) = \max_{1 \leq i \leq 3} f_i(x)$$

subject to

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

where

$$f_1(x) = x_1^2 + x_2^2 + x_1 x_2 - 1,$$

$$f_2(x) = \sin(x_1),$$

$$f_3(x) = -\cos(x_2).$$

The starting point is

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

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 $\text{EPS} = 10^{-6}$, 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) 000001
C 000002
C J.HALD - EXAMPLE 1. 000003
C 000004
DIMENSION X(2),W(67),B(1),C(1,2),H(4) 000005
EXTERNAL FDF 000006
DATA H/10HPROGRAM TR,10HMML1 : J.H,10HALD - EXAM,10HPLE 1 000007
CALL MMXHDR(4,H) 000008
N=2 000009
M=3 000010
L=1 000011
LEQ=0 000012
LC=1 000013
B(1)=-2.5E0 000014
C(1,1)=-3.0 000015
C(1,2)=-1.0 000016
X(1)=-2.0 000017
X(2)=-1.0 000018
DX=0.2 000019
EPS=1.E-6 000020
MAXF=50 000021
KEQS=3 000022
IW=67 000023
ICH=1 000024
IPC=-10 000025
CALL MMLC1A(FDF,N,M,L,LEQ,B,C,LC,X,DX,EPS,MAXF,KEQS,W,IW,
1 ICH,IPC,IFALL) 000026
STOP 000027
END 000028

C SUBROUTINE FDF(N,M,X,DF,F) 000029
C DIMENSION X(N),F(M),DF(M,N) 000030
X1=X(1) 000031
X2=X(2) 000032
F(1)=X1*X1+X2*X2+X1*X2-1.0 000033
F(2)=SIN(X1) 000034
F(3)=-COS(X2) 000035
DF(1,1)=X1+X1+X2 000036
DF(1,2)=X2+X2+X1 000037
DF(2,1)=COS(X1) 000038
DF(2,2)=0.0 000039
DF(3,1)=0.0 000040
DF(3,2)=SIN(X2) 000041
RETURN 000042
END 000043
000044
000045
000046

DATE : 82/04/22. TIME : 15.17.59.
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE) PAGE : 1
PROGRAM TRMML1 : J.HALD - EXAMPLE 1 (V:82.04)

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	-2.00000000000000E+00	1 6.000000000000E+00
2	-1.00000000000000E+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 -7.78668934368E-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

Example 2 [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

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

subject to linear equality constraint

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

where

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

The solutions are shown for 4 different starting points \underline{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)          000001
C                                             000002
C BRENT EXAMPLE                               000003
C                                             000004
C                                             000005
DIMENSION X(2),XX(4,2),B(1),C(1,2),T(3),W(59) 000006
EXTERNAL FDF                                000007
DATA XX/2.0,-2.0,2.0,2.0,                   000008
1      2.0,-2.0,0.0,1.0/                      000009
DATA B/0.0/,C/4.0,4.0/                      000010
DATA T/10HTRMML2 : B,10HRENT EXAMP,10HLE    000011
CALL MMXHDR(3,T)                            000012
N=2                                         000013
M=2                                         000014
LEQ= 1                                      000015
L= 1                                         000016
IL= 1                                       000017
IPR=-10                                     000018
DO 20 I=1,4                                  000019
X(1)=XX(I,1)                                000020
X(2)=XX(I,2)                                000021
DX=.2                                         000022
EPS=1.E-6                                    000023
MAXF=50                                      000024
KEQS=2                                       000025
IW=59                                         000026
ICH=6                                         000027
CALL MMCLC1A(FDF,N,M,L,LEQ,B,C,IL,X,DX,EPS,MAXF,KEQS,W,IW,ICH, 000028
1 IPR,IFLAG)
IPR=0                                         000029
20 CONTINUE                                   000030
STOP                                         000031
END                                          000032
C                                             000033
C                                             000034
SUBROUTINE FDF(N,M,X,DF,F)                  000035
DIMENSION X(N),DF(M,N),F(M)                 000036
X1=X(1)                                      000037
X2=X(2)                                      000038
R1=X1-X2                                     000039
R2=(X1-2.0)**2+X2*X2                       000040
F(1)=R1*R2+3.0*X1+5.0*X2                   000041
F(2)=-F(1)                                    000042
DF(1,1)=R2+(R1+R1)*(X1-2.0)+3.0           000043
DF(1,2)=-R2+R1*(X2+X2)+5.0                 000044
DF(2,1)=-DF(1,1)                           000045
DF(2,2)=-DF(1,2)                           000046
RETURN                                       000047
END                                         000048
```

DATE : 82/04/22.

TIME : 15.26.07.

LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)

PAGE : 1
(V:82.04)

TRMML2 : BRENT EXAMPLE

INPUT DATA

NUMBER OF VARIABLES (N)	2
NUMBER OF FUNCTIONS (MD)	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

1 2.000000000000E+00
2 2.000000000000E+00

FUNCTION VALUES

1 1.600000000000E+01
2 -1.600000000000E+01

VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.

SOLUTION

VARIABLES

1 -1.894780628693E-14
2 1.326346440086E-13

FUNCTION VALUES

1 3.635071051258E-27
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.
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)

PAGE : 1
(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 (IPRO)	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.
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)

PAGE : 1
(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.
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)

PAGE : 1
(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

$$f_1(x) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_1x_3$$

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 $f_1(x^*) = 1/9$ at the point $x^* = [4/3 \ 7/9 \ 4/9]^T$.

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

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

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

DX	KEQS		
	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 than the results reported in [7, Example 5], where the constraints have been converted to additional residual functions.

PROGRAM TRMML3(OUTPUT,TAPE2=OUTPUT) 000001
C 000002
C BEALE CONSTRAINED FUNCTION 000003
C 000004
DIMENSION X(3),W(98),C(4),DC(4,3),T(4) 000005
EXTERNAL FDF 000006
DATA C/0.0,0.0,0.0,3.0/ 000007
DATA DC/1.0,0.0,0.0,-1.0, 000008
1 0.0,1.0,0.0,-1.0, 000009
2 0.0,0.0,1.0,-2.0/ 000010
DATA T/10HTRMML3 : B,10HEALE CONST,10HAINED FUN,5HCTION/ 000011
CALL MMXHDR(4,T) 000012
N=3 000013
M=1 000014
L=4 000015
LEQ=0 000016
IC=4 000017
X(1)=0.5 000018
X(2)=0.5 000019
X(3)=0.5 000020
DX=0.25 000021
EPS=1.E-6 000022
MAXF=50 000023
KEQS=2 000024
IW=98 000025
IPR=-10 000026
LCH=2 000027
CALL MMLC1A(FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQS,W,IW, 000028
1 LCH,IPR,IFALL) 000029
STOP 000030
END 000031
C 000032
C SUBROUTINE FDF(N,M,X,DF,F) 000033
DIMENSION X(N),F(M),DF(M,N) 000034
X1=X(1) 000035
X2=X(2) 000036
X3=X(3) 000037
F(1)=9.0-8.0*X1-6.0*X2-4.0*X3+2.0*(X1*(X1+X2+X3)+X2*X2)+X3*X3 000038
DF(1,1)=4.0*X1+2.0*(X2+X3)-8.0 000039
DF(1,2)=4.0*X2+2.0*X1-6.0 000040
DF(1,3)=2.0*(X1+X3)-4.0 000041
RETURN 000042
END 000043
000044

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

PAGE : 1
(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)

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

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]

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

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

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

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:

DX	KEQS		
	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) 000001
C 000002
C BEALE CONSTRAINED FUNCTION 000003
C 000004
DIMENSION X(3), W(104), C(3), DC(3,3), T(4) 000005
EXTERNAL FDF 000006
DATA C/0.0,0.0,0.0/ 000007
DATA DC/1.0,0.0,0.0, 000008
1 0.0,1.0,0.0, 000009
2 0.0,0.0,1.0/ 000010
DATA T/10HTRMML4 : B, 10HEALE CONST, 10HAINED FUN, 5HCTION/ 000011
CALL MMXHDR(4, T) 000012
N=3 000013
M=2 000014
L=3 000015
LEQ=0 000016
IC=3 000017
X(1)=0.5 000018
X(2)=0.5 000019
X(3)=0.5 000020
DX=0.25 000021
EPS=1.E-6 000022
MAXF=50 000023
KEQS=2 000024
IW=104 000025
LCH=2 000026
IPR=-10 000027
CALL MMLC1A(FDF, N, M, L, LEQ, C, DC, IC, X, DX, EPS, MAXF, KEQS, W, IW, 000028
1 LCH, IPR, IFALL) 000029
STOP 000030
END 000031
C 000032
C SUBROUTINE FDF(N, M, X, DF, F) 000033
DIMENSION X(N), F(M), DF(M, N) 000034
X1=X(1) 000035
X2=X(2) 000036
X3=X(3) 000037
F(1)=9.0-8.0*X1-6.0*X2-4.0*X3+2.0*(X1*(X1+X2+X3)+X2*X2)+X3*X3 000038
DF(1, 1)=4.0*X1+2.0*(X2+X3)-8.0 000039
DF(1, 2)=4.0*X2+2.0*X1-6.0 000040
DF(1, 3)=2.0*(X1+X3)-4.0 000041
F(2)=F(1)+X1+X2+X3+X3-3.0 000042
DF(2, 1)=DF(1, 1)+1.0 000043
DF(2, 2)=DF(1, 2)+1.0 000044
DF(2, 3)=DF(1, 3)+2.0 000045
RETURN 000046
END 000047
000048

DATE : 82/04/22. TIME : 16.40.03.
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)
TRMML4 : BEALE CONSTRAINED FUNCTION

PAGE : 1
(V:82.04)

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 1.33333333174E+00	1 1.111111111109E-01
2 7.77777778903E-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 $\underline{x}^* = [x_1^* \ x_2^*]^T$ that maximizes the relative tolerance r in the region R_c defined by the inequalities

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

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

$$-60 + 4x_1 + 15x_2 \geq 0,$$

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

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

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

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

subject to constraints

$$\begin{aligned} 2 + 2(x_1 + x_3) - (x_2 + x_4) &\geq 0, \\ 143 - 11(x_1 + x_3) - 13(x_2 + x_4) &\geq 0, \\ -60 + 4(x_1 + x_3) + 15(x_2 + x_4) &\geq 0, \\ x_3 &\geq 0, \\ x_4 &\geq 0, \end{aligned}$$

where

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

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

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

$$\begin{aligned} 2 + 2(x_1 - x_3) - (x_2 + x_4) &\geq 0, \\ 143 - 11(x_1 + x_3) - 13(x_2 + x_4) &\geq 0, \\ -60 + 4(x_1 - x_3) + 15(x_2 - x_4) &\geq 0, \end{aligned}$$

or, finally,

$$\begin{aligned} 2 + 2x_1 - x_2 - 2x_3 - x_4 &\geq 0, \\ 143 - 11x_1 - 13x_2 - 11x_3 - 13x_4 &\geq 0, \\ -60 + 4x_1 + 15x_2 - 4x_3 - 15x_4 &\geq 0. \end{aligned}$$

The solution is shown for the starting point $\underline{x}^0 = 1$, 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) 000001
C 000002
C TOLERANCING EXAMPLE 000003
C 000004
DIMENSION X(4),B(5),C(5,4),W(159),H(3) 000005
EXTERNAL FT 000006
DATA B/2.0,143.0,-60.0,0.0,0.0/ 000007
DATA C/2.0,-11.0,4.0,0.0,0.0, 000008
1 -1.0,-13.0,15.0,0.0,0.0, 000009
2 -2.0,-11.0,-4.0,1.0,0.0, 000010
3 -1.0,-13.0,-15.0,0.0,1.0/ 000011
DATA H/10HTRMML5 : T, 10HOLERANCING, 10H EXAMPLE 000012
CALL MMXHDR(3,H) 000013
N=4 000014
M=2 000015
DX= 1.0 000016
EPS= 1.E-6 000017
IC=5 000018
L=5 000019
LEQ=0 000020
X(1)=1.0 000021
X(2)=1.0 000022
X(3)=1.0 000023
X(4)=1.0 000024
MAXF=25 000025
KEQS=3 000026
IW=159 000027
ICH=6 000028
IPR=-1000 000029
CALL MMLC1A(FT,N,M,L,LEQ,B,C,IC,X,DX,EPS,MAXF,KEQS,W,IW,ICH,IPR, 000030
1 IFLAG) 000031
STOP 000032
END 000033
C 000034
SUBROUTINE FT(N,M,X,D,F) 000035
DIMENSION X(N),D(M,N),F(M) 000036
X1=X(1) 000037
X2=X(2) 000038
X3=X(3) 000039
X4=X(4) 000040
F(1)=-X3/X1 000041
F(2)=-X4/X2 000042
D(1,1)=X3/(X1*X1) 000043
D(1,2)=0.0 000044
D(1,3)=-1.0/X1 000045
D(1,4)=0.0 000046
D(2,1)=0.0 000047
D(2,2)=X4/(X2*X2) 000048
D(2,3)=0.0 000049
D(2,4)=-1.0/X2 000050
RETURN 000051
END 000052
000053

DATE : 82/05/19.

TIME : 14.56.41.

LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)

PAGE : 1

(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.
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE) PAGE : 2
(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

<u>Subroutine</u>	<u>Number of Lines</u> (source text)	<u>Number of Words</u> (compiled code)	<u>Listing from Page</u>
MMLC1A	87	742	48
MMLA1Q	11	121	49
MMX00Z	9	23	49
MMX00Q	35	216	49
MMX00V	26	235	50
MMX00G	35	267	50
MMX00H	67	435	51
MMX00B	28	151	52
MMXPSZ	12	42	52
MMXPLM	11	37	52
MMXLLM	11	36	53
MMXHDR	16	47	53
MMXGLM	13	44	53
MMXGVL	11	41	53
MMLC8A	66	330	54
MMLC9A	245	1516	55
S2LA1Q	271	1441	58
FEASI	229	1360	63
MMLPA	280	1545	66
LINSYS	93	333	70
BFGS	43	215	72
ADDCL	92	357	73
DELCL	53	220	74
UTTRNS	36	130	75
UTRNS	33	141	75
RSOLV	21	76	76
TSOLV	19	72	76
HACUM	55	255	77
LIMIT	18	63	78

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
C
LEVEL 1 INTERFACE (STANDARD ENTRY)

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 MMXP SZ (-1)
CALL MMXP LM (-1)
CALL MMXL LM (-1)
CALL MMX HDR (-1,HD)
CALL MMX GLM (-1,-1)
CALL MMX GVL (-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 (11H0 INPUT DATA/11H -----//
1 27H NUMBER OF VARIABLES (N) ,25(2H.),I4//
2 27H NUMBER OF FUNCTIONS (MD) ,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, MD)
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 000066
IF (NRL.LT.9) CALL MMX00B 000067
WRITE (LCH,50) 000068
50 FORMAT (//9H SOLUTION/9H -----) 000069
NRL=NRL-4 000070
LML=LML-4 000071
CALL MMX00V (MMX00B,X,N,W,MD 000072
CPU=TEND-TBEG 000073
IF (NRL.LT.9) CALL MMX00B 000074
WRITE (LCH,60) IFALL,MAXF,KEQS,CPU 000075
60 FORMAT (//29H TYPE OF SOLUTION (IFALL) ,24(2H.),I4// 000076
1 35H NUMBER OF FUNCTION EVALUATIONS ,21(2H. ),I4// 000077
2 31H NUMBER OF SHIFTS TO STAGE-2 ,23(2H. ),I4// 000078
3 31H EXECUTION TIME (IN SECONDS) ,21(2H. ),1H.,F7.3/) 000079
RETURN 000080
70 WRITE (LCH,80) 000081
80 FORMAT (//42H E M P T Y F E A S I B L E R E G I O N/) 000082
RETURN 000083
90 WRITE (LCH,100) 000084
100 FORMAT (//40H I N C O R R E C T P A R A M E T E R S/) 000085
RETURN 000086
END 000087
C 000088
C 000089
SUBROUTINE MMLA1Q (FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQS,W,IW,I 000090
1FALL) 000091
EXTERNAL FDF,MMX00Z 000092
C 000093
C LEVEL 2 INTERFACE (J.HALD ENTRY) 000094
C 000095
DIMENSION C(1), DC(1,1), X(1), W(1) 000096
CALL MMLC8A (MMX00Z,MMX00Z,FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQ 000097
1S,W,IW,IFALL)
RETURN
END
C 000100
C 000101
C SUBROUTINE MMX00Z (FUN,N,M,X,DF,F,K,NS) 000102
C 000103
C DUMMY SUBROUTINE WHICH FOR BASIC AND ORIGINAL ENTRIES SUBSTITUTES 000104
C SUBROUTINE MMX00Q/11Q. 000105
C 000106
C EXTERNAL FUN 000107
C DIMENSION X(N), DF(M,N), F(M) 000108
C RETURN 000109
C END 000110
C 000111
C 000112
C SUBROUTINE MMX00Q (FHH,N,M,X,DF,F,K,NS) 000113
C 000114
C PRINT RESULTS OF FUNCTION EVALUATION. 000115
C 000116
C 000117
C EXTERNAL FHH 000118
C DIMENSION X(N), DF(M,N), F(M) 000119
C COMMON /MMX00/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG 000120
1H,DAT,TIM,LHT,H(8) 000121
IF (LCH.LE.0) RETURN 000122
IF (LV1+LV2.EQ.0) RETURN 000123
IF (K.LE.LV2) GO TO 10 000124
IF (LV1.EQ.0) RETURN 000125
IF (MOD(K,LV1).NE.0) RETURN 000126
10 IF (NRP.LE.LMP.AND.LML.GE.0) GO TO 30 000127
LV1=0 000128
LV2=0 000129
WRITE (LCH,20) 000130
```

```
20 FORMAT (//26H ( LISTING LIMIT REACHED )//)          000131
NRL=NRL-5                                         000132
LML=LML-5                                         000133
RETURN
30 IF (NRL.LT.7) CALL FHH                         000134
WRITE (LCH,40) K,NS                               000135
40 FORMAT (22H0FUNCTION EVALUATION :,I4,2H/,I2)      000136
NRL=NRL-2                                         000137
LML=LML-2                                         000138
CALL MMX00V (FHH,X,N,F,MD)                      000139
IF (LG1+LG2.EQ.0) RETURN                         000140
IF (K.LE.LG2) GO TO 50                           000141
IF (K.LE.LV2) RETURN                            000142
IF (LG1.EQ.0) RETURN                            000143
IF (MOD(K,LG1).NE.0) RETURN                      000144
50 CALL MMX00G (FHH,DF,M,N)                      000145
RETURN
END
C
C SUBROUTINE MMX00V (FHH,X,N,F,MD)                000146
C PRINT VALUES OF VARIABLES AND RESIDUAL FUNCTIONS. 000147
C
DIMENSION X(N), F(M)
COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MLX,LMP,LML,LG
1H,DAT,TIM,LHT,H(8)                                000148
IF (LCH.LE.0) RETURN                            000149
K=MAX0(N,M)                                         000150
IF (NRL.LT.5) CALL FHH                           000151
WRITE (LCH,10)                                     000152
10 FORMAT (/30X,9HVARIABLES,18X,15HFUNCTION VALUES/) 000153
NRL=NRL-3                                         000154
LML=LML-3                                         000155
DO 40 I=1,K                                         000156
IF (NRL.LE.0) CALL FHH                           000157
IF (I.LE.N.AND.I.LE.M) WRITE (LCH,20) I,X(I),I,F(I) 000158
IF (I.LE.N.AND.I.GT.M) WRITE (LCH,20) I,X(I)        000159
IF (I.GT.N.AND.I.LE.M) WRITE (LCH,30) I,F(I)        000160
20 FORMAT (18X,I4,2X,1PE19.12,5X,I4,2X,1PE19.12)   000161
30 FORMAT (48X,I4,2X,1PE19.12)                     000162
NRL=NRL-1                                         000163
LML=LML-1                                         000164
40 CONTINUE
RETURN
END
C
C SUBROUTINE MMX00G (FHH,G,M,N)                   000165
C PRINT PARTIAL DERIVATIVES OF RESIDUAL FUNCTIONS. 000166
C
DIMENSION G(M,N)
COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MLX,LMP,LML,LG
1H,DAT,TIM,LHT,H(8)                                000167
IF (LCH.LE.0) RETURN                            000168
IF (NRL.LT.7) CALL FHH                           000169
MM=MIN0(M,LMF)
NN=MIN0(N,LMV)
WRITE (LCH,10)
10 FORMAT (30H0 GRADIENTS ( DF.I / DX.J ))::)    000170
NRL=NRL-2                                         000171
LML=LML-2                                         000172
DO 60 K=1,NN,LGH
IF (NRL.LT.5) CALL FHH                         000173
60 CONTINUE
RETURN
END
```

J1=K 000196
J2=MIN0(NN,K+LGH-1) 000197
WRITE (LCH,20) (J,J=J1,J2) 000198
20 FORMAT (1H0,9X,12HVARIABLES(J),10(15,5X)) 000199
WRITE (LCH,30) 000200
30 FORMAT (10X,12HFUNCTIONS(I)) 000201
NRL=NRL-3 000202
LML=LML-3 000203
DO 50 I=1,MM 000204
IF (NRL.LE.0) CALL FHH 000205
WRITE (LCH,40) I,(G(I,J),J=J1,J2) 000206
40 FORMAT (10X,I6,4X,10(1PE10.2)) 000207
NRL=NRL-1 000208
LML=LML-1 000209
50 CONTINUE 000210
60 CONTINUE 000211
RETURN 000212
END 000213
C 000214
C SUBROUTINE MMX00H (FHH,FDF,N,M,X,DF,F,DG,DH,G) 000215
C 000216
C NUMERICAL VERIFICATION OF USER-DEFINED PARTIAL DERIVATIVES 000217
C (VARIABLES ARE DISTURBED ONE BY ONE). 000218
C 000219
C 000220
DIMENSION X(N), DF(M,N), F(M), DG(M), DH(M,N), G(M) 000221
COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MLX,LMP,LML,LG 000222
1H,DAT,TIM,LHT,H(8) 000223
IF (LCH.LE.0) RETURN 000224
K=0 000225
CALL FDF (N,M,X,DF,F) 000226
DO 60 I=1,N 000227
Z=X(I) 000228
DX=1.E-6*Z 000229
IF (ABS(DX).LT.1.E-10) DX=1.E-10 000230
DX2=DX+DX 000231
X(I)=Z+DX 000232
CALL FDF (N,M,X,DH,F) 000233
DO 10 J=1,M 000234
DG(J)=DH(J,I) 000235
10 CONTINUE 000236
X(I)=Z-DX 000237
CALL FDF (N,M,X,DH,G) 000238
X(I)=Z 000239
DO 50 J=1,M 000240
Y=DF(J,I) 000241
Z=F(J)-G(J) 000242
IF (ABS(Z).LE.0.5E-13*(F(J)+G(J))) Z=0.0 000243
Z=Z/DX2 000244
IF (ABS(Y).LE.1.E-20.AND.ABS(Z).LE.1.E-20) GO TO 50 000245
IF (ABS(Z).LT.1.E-20) Z=SIGN(1.E-20,Z) 000246
R=100.0*ABS((Z-Y)/Z) 000247
IF (R.LE.1.0) GO TO 50 000248
IF (SIGN(1.0,DG(J))+SIGN(1.0,DH(J,I)).EQ.0.0) GO TO 50 000249
IF (K.NE.0) GO TO 30 000250
IF (NRL.LT.5) CALL FHH 000251
WRITE (LCH,20) 000252
20 FORMAT (38H0VERIFICATION OF PARTIAL DERIVATIVES ://
1 1H0,18X,52H DF.I / DX.J : USER DEFINED NUMERICAL DIFFERENCE) 000253
NRL=NRL-4 000254
LML=LML-4 000255
30 K=K+1 000256
IF (NRL.LE.0) CALL FHH 000257
WRITE (LCH,40) J,I,Y,Z,R 000258
40 FORMAT (19X,I5,3X,I4,6X,1PE10.3,2X,1PE10.3,4X,0PF6.1,2H %) 000259
000260

```
NRL=NRL-1          000261
LML=LML-1          000262
50 CONTINUE         000263
60 CONTINUE         000264
   IF (K.NE.0) GO TO 80 000265
   IF (NRL.LT.2) CALL FHH 000266
   WRITE (LCH,70)        000267
70 FORMAT (47H0VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.) 000268
   NRL=NRL-2          000269
   LML=LML-2          000270
80 RETURN           000271
END                000272
C                   000273
C                   SUBROUTINE MMX00B 000274
C                   CHANGE PAGE AND PRINT PAGE HEADER. 000275
C                   COMMON /MMX000/ LCH, LV1, LV2, LG1, LG2, LMF, LMV, NRP, NRL, MXL, LMP, LML, LG 000276
1H, DAT, TIM, LHT, H(8) 000277
   IF (LCH.LE.0) RETURN 000278
   IF (NRP.LT.LMP) GO TO 20 000279
   LV1=0               000280
   LV2=0               000281
   WRITE (LCH,10)        000282
10 FORMAT (//27H ( LIMIT OF PAGES REACHED )) 000283
20 NRP=NRP+1          000284
   NRL=MXL-5          000285
   LML=LML-5          000286
   WRITE (LCH,30) DAT, TIM, NRP 000287
30 FORMAT (1H1/7H DATE :, A10, 19X, 6HTIME :, A10, 20X, 6HPAGE :, I3/ 000288
1 57H LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE), 15X, 000289
2 9H(V:82.04)) 000290
   IF (LHT.LE.0) GO TO 50 000291
   WRITE (LCH,40) (H(J), J=1, LHT) 000292
40 FORMAT (1H0,8A10) 000293
   NRL=NRL-2          000294
   LML=LML-2          000295
50 WRITE (LCH,60)        000296
60 FORMAT (1H0)          000297
   RETURN             000298
END                000299
C                   SUBROUTINE MMXPSZ (L) 000300
C                   DEFINE THE PAGE SIZE (I.E. THE NUMBER OF LINES PER PAGE). 000301
C                   COMMON /MMX000/ LCH, LV1, LV2, LG1, LG2, LMF, LMV, NRP, NRL, MXL, LMP, LML, LG 000302
1H, DAT, TIM, LHT, H(8) 000303
   DATA LL/65/          000304
   IF (L.GT.0) LL=MAX0(25, L) 000305
   IF (L.EQ.0) LL=0 000306
   MXL=LL              000307
   RETURN              000308
END                000309
C                   SUBROUTINE MMXPML (L) 000310
C                   DEFINE THE LIMIT OF PRINTED PAGES. 000311
C                   COMMON /MMX000/ LCH, LV1, LV2, LG1, LG2, LMF, LMV, NRP, NRL, MXL, LMP, LML, LG 000312
1H, DAT, TIM, LHT, H(8) 000313
   DATA LL/10/          000314

```

IF (L.GT.0) LL=MIN0(50,L)	000326
LMP=LL	000327
RETURN	000328
END	000329
C	000330
C SUBROUTINE MMXLLM (L)	000331
C	000332
C DEFINE THE LIMIT OF PRINTED LINES.	000333
C	000334
COMMON /MMX000/ LCH, LV1, LV2, LG1, LG2, LMF, LMV, NRP, NRL, MXL, LMP, LML, LG	000335
1H, DAT, TIM, LHT, H(8)	000336
DATA LL/750/	000337
IF (L.GT.0) LL=L	000338
LML=LL	000339
RETURN	000340
END	000341
C	000342
C SUBROUTINE MMXHDR (L, T)	000343
C	000344
C DEFINE THE HEADER LINE.	000345
C	000346
DIMENSION T(1)	000347
COMMON /MMX000/ LCH, LV1, LV2, LG1, LG2, LMF, LMV, NRP, NRL, MXL, LMP, LML, LG	000348
1H, DAT, TIM, LHT, H(8)	000349
DATA LL/0/	000350
IF (L.GE.0) LL=MIN0(8,L)	000351
LHT=LL	000352
IF (L.LE.0) RETURN	000353
DO 10 I=1,LL	000354
H(I)=T(I)	000355
10 CONTINUE	000356
RETURN	000357
END	000358
C	000359
C SUBROUTINE MMXGLM (K, L)	000360
C	000361
C DEFINE THE SIZE OF PRINTED JACOBIAN.	000362
C	000363
COMMON /MMX000/ LCH, LV1, LV2, LG1, LG2, LMF, LMV, NRP, NRL, MXL, LMP, LML, LG	000364
1H, DAT, TIM, LHT, H(8)	000365
DATA KK/25/, LL/10/	000366
IF (K.GT.0) KK=K	000367
IF (L.GT.0) LL=L	000368
LMF=KK	000369
LMV=LL	000370
RETURN	000371
END	000372
C	000373
C SUBROUTINE MMXGVL (L)	000374
C	000375
C DEFINE THE NUMBER OF JACOBIAN COLUMNS PRINTED IN ONE LINE.	000376
C	000377
C	000378
C	000379
COMMON /MMX000/ LCH, LV1, LV2, LG1, LG2, LMF, LMV, NRP, NRL, MXL, LMP, LML, LG	000380
1H, DAT, TIM, LHT, H(8)	000381
DATA LL/10/	000382
IF (L.GT.0) LL=MAX0(MIN0(10,L),5)	000383
LGH=LL	000384
RETURN	000385
END	000386
C	000387
C	000388
C	000389
C	000390

SUBROUTINE MMLC8A (FQQ, FHH, FDF, N, M, L, LEQ, C, DC, IC, X, DX, EPS, MAXF, KEQ, 000391
1S, W, IW, IFALL) 000392
000393
C MMLC8A MINIMIZES THE MAXIMUM VALUE OF A SET OF NONLINEAR FUNCTIONS 000394
C SUBJECT TO LINEAR EQUALITY AND INEQUALITY CONSTRAINTS. DERIVATIVES 000395
C OF NONLINEAR FUNCTIONS ARE REQUIRED. 000396
C 000397
C FOR A PROGRAM DESCRIPTION SEE: 000398
C J. HALD: "MMLA1Q, A FORTRAN SUBROUTINE FOR LINEARLY CONSTRAINED 000399
C MINIMAX OPTIMIZATION", REPORT NO. NI-81-1, INSTITUTE FOR NUMERICAL 000400
C ANALYSIS, TECHNICAL UNIVERSITY OF DENMARK, DK-2800 LYNGBY, DENMARK 000401
C 000402
C THE SUBROUTINES: MMLPA, FEASI, S2LA1Q, BFGS, ADDCL, DELCL, HACUM, 000403
C UTRNS, UTRNS, RSOLV, TSOLV, LIMIT, LINSYS MUST BE AVAILABLE. 000404
C 000405
C DIMENSION C(1), DC(1,1), X(1), W(1) 000406
C EXTERNAL FQQ, FHH, FDF 000407
C COMMON /MML000/ MARK 000408
C DATA ZERO/0.0/ 000409
C MARK=1 000410
C 000411
C CHECK INPUT QUANTITIES 000412
C 000413
IWR=2*M*N+5*N*N+4*M+8*N+4*IC+3 000414
IFALL=-1 000415
IF (IWR.LT. IWR.OR. N.LT. 1 .OR. M.LT. 1 .OR. L.LT. 0 .OR. LEQ.LT. 0 .OR. LEQ.GT. 000416
1L.OR. LEQ.GT. N.OR. IC.LT.L.OR.DX.LE.ZERO.OR.EPS.LT.ZERO.OR.MAXF.LE.0 000417
2) GO TO 10 000418
C 000419
C SPLIT UP THE WORK AREA 000420
C 000421
N1=N+1 000422
NN=N+N 000423
NF=1 000424
NF1=NF+M 000425
NDF=NF1+M 000426
NDF1=NDF+M*N 000427
NX1=NDF1+M*N 000428
NB=NX1+N 000429
NU=NB+N*N 000430
NR=NU+N*N 000431
NA=NU 000432
NCL=NA+NN*NN 000433
NWL=NCL+IC 000434
NWLI=NWL+IC 000435
NXX=NWL1+IC 000436
NW=NXX+N 000437
NW1=NW+N 000438
NW2=NW1+N 000439
NWM=NW2+N 000440
NAS=NWM+M 000441
NKS=NAS+N1 000442
NKS0=NKS+N1 000443
NKSTC=NKS0+N1 000444
NKSTF=NKSTC+IC 000445
IL=MAX0(1, IC) 000446
CALL MMLC9A (FQQ, FHH, FDF, N, M, L, LEQ, C, DC, IL, X, DX, EPS, MAXF, KEQS, N1, N 000447
1N, W(NF), W(NF1), W(NDF), W(NDF1), W(NX1), W(NB), W(NU), W(NR), W(NA), W(NCL 000448
2), W(NWL), W(NWL1), W(NXX), W(NW), W(NW1), W(NW2), W(NWM), W(NAS), W(NKS), W 000449
3(NKS0), W(NKSTC), W(NKSTF), IFALL) 000450
IF (IFALL.LT.0) GO TO 10 000451
RETURN 000452
10 MAXF=0 000453
KEQS=0 000454
RETURN 000455

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(MD), F1(MD), WM(MD), 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(MD)
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
C SEPS=2.0*16.0**(-12)
C
C SET SOME CONSTANTS
C
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
000456
000457
000458
000459
000460
000461
000462
000463
000464
000465
000466
000467
000468
000469
000470
000471
000472
000473
000474
000475
000476
000477
000478
000479
000480
000481
000482
000483
000484
000485
000486
000487
000488
000489
000490
000491
000492
000493
000494
000495
000496
000497
000498
000499
000500
000501
000502
000503
000504
000505
000506
000507
000508
000509
000510
000511
000512
000513
000514
000515
000516
000517
000518
000519
000520

C
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
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 SUBPROBLEMS
C
C 130 CALL MMLPA (F,DF,CLOC,DC,M,N,N1,IC,LEQ,L1,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
C 140 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
C IF ((FMM0-FMM1).GT.0.25*(FMM0-FMM1)) 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-FMM1)) DX=XXN+XXN
170 DIV4=.FALSE.

C C UPDATE THE HESSIAN APPROXIMATION
C
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
000521
000522
000523
000524
000525
000526
000527
000528
000529
000530
000531
000532
000533
000534
000535
000536
000537
000538
000539
000540
000541
000542
000543
000544
000545
000546
000547
000548
000549
000550
000551
000552
000553
000554
000555
000556
000557
000558
000559
000560
000561
000562
000563
000564
000565
000566
000567
000568
000569
000570
000571
000572
000573
000574
000575
000576
000577
000578
000579
000580
000581
000582
000583
000584
000585

W1(J)=W1(J)+T*DF1(KK,J)	000586
WC(J)=W(J)+T*DF(KK,J)	000587
200 CONTINUE	000588
210 CONTINUE	000589
DO 220 I=1,N	000590
W2(I)=W1(I)-W(I)	000591
220 CONTINUE	000592
CALL BFGS (B,N,W2,XX,W,SEPS)	000593
C TEST IF THE NEW POINT IS ACCEPTABLE	000594
C IF ((FMM0-FMM1).LE.0.01*(FMM0-FMM)) GO TO 320	000595
C COMPARE THE NEW ACTIVE SET WITH THE PRECEDING	000596
C IF (NACT0.NE.NACT) GO TO 250	000597
DO 240 I=1,NACT	000598
K=KSET(I)	000599
DO 230 J=1,NACT	000600
IF (K.EQ.KSET0(J)) GO TO 240	000601
230 CONTINUE	000602
GO TO 250	000603
240 CONTINUE	000604
KEQSET=KEQSET+1	000605
GO TO 260	000606
250 KEQSET=1	000607
C INTRODUCE THE NEW POINT	000608
C	000609
260 NSTEP=NSTEP+1	000610
XN=XZERO	000611
FMM0=FMM1	000612
NACT0=NACT	000613
DO 270 I=1,N	000614
X(I)=X1(I)	000615
XN=XN+X(I)**2	000616
DO 270 J=1,M	000617
270 DF(J,I)=DF1(J,I)	000618
XN=SQRT(XN)	000619
DO 280 I=1,M	000620
F(I)=F1(I)	000621
280 CONTINUE	000622
DO 290 I=1,NACT0	000623
KSET0(I)=KSET(I)	000624
290 CONTINUE	000625
IF (L.EQ.0) GO TO 320	000626
DO 310 I=1,L	000627
T=C(I)	000628
DO 300 J=1,N	000629
T=T+DC(I,J)*X(J)	000630
300 CONTINUE	000631
CLOC(I)=T	000632
310 CONTINUE	000633
C TEST OF CONVERGENCE CRITERION	000634
C	000635
320 IF (XXN.LE.EPS*XN) GO TO 410	000636
IF (XXN.LE.SEPS*XN) GO TO 400	000637
IF (XXN.LE.XM50) GO TO 410	000638
IF (NCALL.GE.MAXF) GO TO 420	000639
C TEST FOR SWITCH TO STAGE-2	000640
C	000641
SHIFT=FMM0.LE.FMMREF.AND.KEQSET.GE.KEQS AND.NSTEP.GE.N	000642
IF (.NOT.SHIFT) GO TO 110	000643
	000644
	000645
	000646
	000647
	000648
	000649
	000650

IF (NACT.EQ.N1) GO TO 380 000651
C C TEST FOR POSITIVE DEFINITENESS OF THE HESSIAN APPROX. 000652
C IN A RELEVANT DIRECTION 000653
C
DO 340 I=1,NACT 000654
K=KSET(I) 000655
IF (K.GT.L) GO TO 340 000656
T=ASET(I) 000657
DO 330 J=1,N 000658
W1(J)=W1(J)+T*DC(K,J) 000659
330 CONTINUE 000660
340 CONTINUE 000661
DO 360 I=1,N 000662
T=XZERO 000663
DO 350 J=1,N 000664
T=T+B(I,J)*W1(J) 000665
350 CONTINUE 000666
W(I)=T 000667
360 CONTINUE 000668
T=XZERO 000669
DO 370 I=1,N 000670
T=T+W(I)*W1(I) 000671
370 CONTINUE 000672
IF (T.LE.XZERO) GO TO 110 000673
C SHIFT TO STAGE-2 000674
C
380 NSHIFT=NSHIFT+1 000675
FMMREF=FMM0-10.0*SEPS*ABS(FMM0) 000676
XXNMAX=AMAX1(DX0,DX+DX) 000677
CALL S2LA1Q (FQQ,FHH,FDF,N,M,L,LEQ,C,CLOC,DC,IC,X,XXNMAX,B,NACT,KS 000678
1ET,ASET,N1,KSTATF,KSTATC,A,XX,NN,F,DF,X1,F1,DF1,W1,W2,EPS,MAXF,NCA 000679
2LL,XXN,NSTEP,SEPS,IFALL) 000680
IF (IFALL.LE.4) GO TO 410 000681
FMM0=-XP73 000682
DO 390 I=1,M 000683
FMM0=AMAX1(FMM0,F(I)) 000684
390 CONTINUE 000685
DX=AMAX1(DX,0.5*XXN) 000686
KEQSET=1 000687
GO TO 110 000688
C RETURN 000689
C
400 IFALL=2 000690
410 MAXF=NCALL 000691
KEQS=NSHIFT 000692
EPS=XXN 000693
RETURN 000694
420 IFALL=3 000695
GO TO 410 000696
END 000697
C
SUBROUTINE S2LA1Q (FQQ,FHH,FDF,N,M,L,LEQ,C,CLOC,DC,IC,X,XXNMAX,B,N 000698
1ACT,KSET,ASET,N1,KSTATF,KSTATC,DZ,ZZ,NN,F,DF,X1,F1,DF1,W,W1,EPS,MA 000699
2XF,NCALL,XXN,NSTEP,SEPS,IFALL) 000700
C STAGE-2 (QUASI-NEWTON) ALGORITHM FOR LINEARLY CONSTRAINED 000701
C MINIMAX OPTIMIZATION. 000702
C
DIMENSION C(IC), CLOC(IC,N), X(N), BN(N), ASET(N1), DZ(NN) 000703
1,NN), ZZ(NN), F(M), DF(M,N), X1(N), F1(M), DF1(M,N), W(N), W1(N) 000704
INTEGER KSET(N1), KSTATF(M), KSTATC(IC) 000705
000706
000707
000708
000709
000710
000711
000712
000713
000714
000715

EXTERNAL FQQ, FHH, FDF
COMMON /MML000/ MARK
DATA XZERO, XONE, XP73, XM50/0.0, 1.0, 1.E73, 1.E-50/

C INITIALIZE

C LI=L-LEQ
C LE1=LEQ+1
C IFALL=0
C SSEPS=SQRT(SEPS)
C KK0=KSET(NACT)-L
C NACT1=NACT-1
C NZ=N+NACT1
C NSTEP2=0
C XXN=XZERO
C DO 10 I=1,M
C KSTATF(I)=0

10 CONTINUE 000716
IF (L.EQ.0) GO TO 30 000717
DO 20 I=1,L 000718
KSTATC(I)=0 000719
000720
000721
000722
000723
000724
000725
000726
000727
000728
000729
000730
000731
000732
000733
000734
000735
000736
000737
000738
000739
000740
000741
000742
000743
000744
000745
000746
000747
000748
000749
000750
000751
000752
000753
000754
000755
000756
000757
000758
000759
000760
000761
000762
000763
000764
000765
000766
000767
000768
000769
000770
000771
000772
000773
000774
000775
000776
000777
000778
000779
000780

C ITERATIVE LOOP STARTS HERE

C SET UP THE ITERATION MATRIX AND THE RIGHTHAND SIDE

C 50 DO 70 I=1,N
C DO 60 J=1,N
C DZ(I,J)=B(I,J)

60 CONTINUE 000748
ZZ(I)=DF(KK0,I) 000749
000750
000751
000752
000753
000754
000755
000756
000757
000758
000759
000760
000761
000762
000763
000764
000765
000766
000767
000768
000769
000770
000771
000772
000773
000774
000775
000776
000777
000778
000779
000780

70 CONTINUE 000743
IF (NACT.EQ.1) GO TO 150 000744
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 000745
GO TO 110 000746
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 000747
110 DO 120 I=N1,NZ
DZ(I,JN)=XZERO

120 CONTINUE 000748
T=ASET(J)
DO 130 I=1,N
ZZ(I)=ZZ(I)-T*DZ(JN,I)

130 CONTINUE 000749
140 CONTINUE 000750
150 RES0=XZERO
DO 160 I=1,NZ

```
RES0=RES0+ZZ(I)**2          000781
160 CONTINUE                 000782
RES0=SQRT(RES0)             000783
C
C      CALCULATE THE QUASI-NEWTON STEP 000784
C
CALL LINSYS (DZ,ZZ,NN,NZ,K,SEPS) 000785
IF (K.EQ.NZ) GO TO 170          000786
IFALL=8                         000787
RETURN                          000788
C
C      CONTROL STEP LENGTH          000789
C
170 XXN1=XZERO                000790
ALFA=XONE                      000791
DO 180 I=1,N                   000792
XXN1=XXN1+ZZ(I)**2            000793
180 CONTINUE                   000794
XXN1=SQRT(XXN1)               000795
IF (XXN1.GT.XXNMAX) ALFA=XXNMAX/XXN1 000796
C
C      WILL OTHER CONSTRAINTS OR FUNCTIONS BECOME ACTIVE ?
C
STEP=XP73                     000797
IF (LI.EQ.0) GO TO 210         000798
DO 200 I=LE1,L                000799
IF (KSTATC(I).NE.0) GO TO 200 000800
T=XZERO                        000801
DO 190 J=1,N                   000802
T=T+ZZ(J)*DC(I,J)            000803
190 CONTINUE                   000804
IF (T.GE.XZERO) GO TO 200     000805
T=-CLOC(I)/T                  000806
IF (T.GT.STEP) GO TO 200     000807
STEP=T                         000808
200 CONTINUE                   000809
210 T0=XZERO                  000810
DO 220 I=1,N                   000811
T0=T0+ZZ(I)*DF(KK0,I)        000812
220 CONTINUE                   000813
F0=F(KK0)                     000814
DO 240 I=1,M                   000815
IF (KSTATF(I).NE.0) GO TO 240 000816
T=XZERO                        000817
DO 230 J=1,N                   000818
T=T+ZZ(J)*DF(I,J)            000819
230 CONTINUE                   000820
T=T0-T                         000821
IF (T.GE.XZERO) GO TO 240     000822
T=(F(I)-F0)/T                000823
IF (T.GT.STEP) GO TO 240     000824
STEP=T                         000825
240 CONTINUE                   000826
IF (STEP.GT.ALFA) GO TO 250   000827
IFALL=9                         000828
ALFA=STEP                       000829
C
C      SCALE THE STEP           000830
C
250 DO 260 I=1,NZ              000831
ZZ(I)=ALFA*ZZ(I)              000832
260 CONTINUE                   000833
XXN1=ABS(ALFA)*XXN1           000834
C
C      CALCULATE FUNCTION VALUES AND RESIDUALS IN THE NEW POINT 000835
C
000836
000837
000838
000839
000840
000841
000842
000843
000844
000845
```

C XN1=XZERO 000846
DO 270 I=1,N 000847
X1(I)=X(I)+ZZ(I) 000848
XN1=XN1+X1(I)**2 000849
270 CONTINUE 000850
XN1=SQRT(XN1) 000851
NCALL=NCALL+1 000852
CALL FDF (N,M,X1,DF1,F1) 000853
CALL FQQ (FHH,N,M,X1,DF1,F1,NCALL,2) 000854
IF (MARK.EQ.0) GO TO 520 000855
DASET0=XZERO 000856
IF (NACT.EQ.1) GO TO 290 000857
DO 280 I=N1,NZ 000858
IF (KSET(I-N).GT.L) DASET0=DASET0-ZZ(I) 000859
280 CONTINUE 000860
290 RES=XZERO 000861
T=ASET(NACT)+DASET0 000862
DO 300 I=1,N 000863
W(I)=-T*DF(KK0,I) 000864
W1(I)=-T*DF1(KK0,I) 000865
300 CONTINUE 000866
IF (NACT.EQ.1) GO TO 350 000867
DO 340 J=1,NACT1 000868
K=KSET(J) 000869
JN=J+N 000870
T=ASET(J)+ZZ(JN) 000871
IF (K.GT.L) GO TO 320 000872
S=C(K) 000873
DO 310 I=1,N 000874
SS=DC(K,I) 000875
W(I)=W(I)+T*SS 000876
W1(I)=W1(I)+T*SS 000877
S=S+SS*X1(I) 000878
310 CONTINUE 000879
RES=RES+S**2 000880
GO TO 340 000881
320 KK=K-L 000882
DO 330 I=1,N 000883
W(I)=W(I)-T*DF(KK,I) 000884
W1(I)=W1(I)-T*DF1(KK,I) 000885
330 CONTINUE 000886
RES=RES+(F1(KK0)-F1(KK))**2 000887
340 CONTINUE 000888
350 DO 360 I=1,N 000889
RES=RES+W1(I)**2 000890
360 CONTINUE 000891
RES=SQRT(RES) 000892
C UPDATE THE HESSIAN APPROXIMATION 000893
C DO 370 I=1,N 000894
W1(I)=W1(I)-W(I) 000895
370 CONTINUE 000896
CALL BFGS (B,N,W1,ZZ,W,SEPS) 000897
C TEST IF THE RESIDUAL HAS DECREASED 000898
C IF (NSTEP2.EQ.0) GO TO 390 000899
IF (RES.LE.0.999*RES0) GO TO 390 000900
C IF NO - TEST FOR MACHINE ACCURACY 000901
C IF (XXN1.GT.SSEPS*(XXNMAX+XN1).OR.NSTEP2.LT.2) GO TO 380 000902
IF ALL=2 000903
000904
000905
000906
000907
000908
000909
000910

```
      RETURN
380 IFALL=5
      RETURN
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
         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      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      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
000911
000912
000913
000914
000915
000916
000917
000918
000919
000920
000921
000922
000923
000924
000925
000926
000927
000928
000929
000930
000931
000932
000933
000934
000935
000936
000937
000938
000939
000940
000941
000942
000943
000944
000945
000946
000947
000948
000949
000950
000951
000952
000953
000954
000955
000956
000957
000958
000959
000960
000961
000962
000963
000964
000965
000966
000967
000968
000969
000970
000971
000972
000973
000974
000975
```

C END 000976
C 000977
C 000978
C SUBROUTINE FEASI (C, DG, IC, LE, LI, N, X, NACT, KSET, ASET, U, R, DL, RIGHT, CU 000979
1P, DLDC, W, KSTAT, IFALL, ACCUM, SEPS) 000980
C 000981
C THE SUBROUTINE FINDS A FEASIBLE POINT FOR A SET OF LINEAR 000982
C EQUALITY AND INEQUALITY CONSTRAINTS. 000983
C 000984
C DIMENSION C(IC), DC(IC,N), X(N), ASET(N), U(N,N), R(N,N), DL(N), R 000985
1IGHT(N), CUP(IC), DLDC(IC), W(N)
INTEGER KSET(N), KSTAT(IC)
LOGICAL ACCUM, OBJECT
DATA XZERO, XP73/0.0, 1.E73/ 000986
C 000987
C INITIALIZE 000988
C 000989
EPS=(N+10)*SEPS 000990
ACCUM=.FALSE. 000991
NACTIN=NACT 000992
NACT=0 000993
LE1=LE+1 000994
LELI=LE+LI 000995
DO 10 I=1,N 000996
X(I)=XZERO 000997
10 CONTINUE 000998
IFALL=0 000999
IF (LELI.EQ.0) RETURN 001000
DO 20 I=1,LELI 001001
KSTAT(I)=0 001002
20 CONTINUE 001003
C MAKE ACTIVE THE EQUALITY CONSTRAINTS PLUS OTHER CONSTRAINTS 001004
C AS DEFINED IN KSET 001005
C 001006
IF (LE.EQ.0) GO TO 50 001007
IF (LE.GT.N) GO TO 410 001008
DO 40 I=1,LE 001009
RIGHT(I)=-C(I) 001010
DO 30 J=1,N 001011
RC(J,I)=DC(I,J) 001012
30 CONTINUE 001013
KSET(I)=I 001014
KSTAT(I)=1 001015
40 CONTINUE 001016
CALL ADDCL (U,R,N,NACT,LE,RIGHT,W,ACCUM,.FALSE.,EPS) 001017
IF (NACT.LT.LE) GO TO 410 001018
50 IF (NACTIN.LT.1) GO TO 80 001019
DO 70 K=1,NACTIN 001020
KK=KSET(K)
IF (KK.LT.LE1.OR.KK.GT.LELI) GO TO 70 001021
NACT1=NACT+1 001022
IF (NACT1.GT.N) GO TO 80 001023
DO 60 I=1,N 001024
RC(I,NACT1)=DC(KK,I) 001025
60 CONTINUE 001026
CALL UTTRNS (U,N,NACT,ACCUM,RC(1,NACT1),W) 001027
CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.FALSE.,EPS) 001028
IF (NACT.LT.NACT1) GO TO 70 001029
RIGHT(NACT1)=-C(KK) 001030
KSET(NACT1)=KK 001031
KSTAT(KK)=1 001032
70 CONTINUE 001033
80 CALL TSOLV (R,N,NACT,RIGHT,X) 001034
IF (NACT.EQ.N) GO TO 100 001035
001036
001037
001038
001039
001040

```
NACT1=NACT+1          001041
DO 90 I=NACT1,N      001042
X(I)=XZERO           001043
90 CONTINUE          001044
100 CALL UTRNS (U,N,NACT,ACCUM,X,W) 001045
C
C       UPDATE THE CONSTRAINTS 001046
C
IF (LI.EQ.0) RETURN 001047
DO 120 I=LE1,LELI   001048
T=C(I)
DO 110 J=1,N        001049
T=T+DC(I,J)*X(J)   001050
110 CONTINUE         001051
CUP(I)=T            001052
120 CONTINUE         001053
C
C       INITIALIZE INEQUALITY CONSTRAINT LOOP 001054
C
DO 130 I=LE1,LELI   001055
IF (CUP(I).LT.XZERO .AND. KSTAT(I).EQ.0) KSTAT(I)=-1 001056
130 CONTINUE         001057
C
C       ACTIVATE VIOLATED INEQUALITY CONSTRAINTS ONE BY ONE 001058
C       USE THE STRONGEST VIOLATED AS OBJECTIVE CONSTRAINT 001059
C
140 FMIN=XP73        001060
DO 150 I=LE1,LELI   001061
IF (KSTAT(I).NE.-1) GO TO 150 001062
IF (CUP(I).GE.FMIN) GO TO 150 001063
FMIN=CUP(I)
NEW=I
150 CONTINUE          001064
IF (FMIN.GE.XZERO) RETURN 001065
DO 160 I=1,N        001066
RIGHT(I)=DC(NEW,I)
160 CONTINUE          001067
CALL UTRNS (U,N,NACT,ACCUM,RIGHT,W) 001068
KSTAT(NEW)=1
C
C       CALCULATE MULTIPLIERS FOR THE NEW ACTIVE CONSTRAINT AND DROP 001069
C       CONSTRAINTS WITH POSITIVE MULTIPLIERS IN ORDER TO ACHIEVE 001070
C       THE DIRECTION OF STEEPEST INCREMENT 001071
C
170 IF (NACT.EQ.LE) GO TO 220 001072
CALL RSOLV (R,N,NACT,RIGHT,ASET)
AMAX=-XP73
DO 180 I=LE1,NACT   001073
IF (ASET(I).LT.AMAX) GO TO 180 001074
K=I
AMAX=ASET(I)
180 CONTINUE          001075
IF (AMAX.LT.XZERO) GO TO 220 001076
KSTAT(KSET(K))=0
DO 190 I=LE1,LELI   001077
IF (KSTAT(I).EQ.-2) KSTAT(I)=0 001078
190 CONTINUE          001079
IF (ACCUMD GO TO 200 001080
ACCUM=.TRUE.
CALL HACUM (U,N,NACT,W) 001081
200 CALL DELCL (K,U,R,N,NACT,RIGHT,.TRUE.) 001082
IF (K.GT.NACT) GO TO 170 001083
DO 210 I=K,NACT    001084
KSET(I)=KSET(I+1)
210 CONTINUE         001085

```

GO TO 170 001106
C 001107
C CALCULATE THE PROJECTED GRADIENT 001108
C 001109
220 T=XZERO 001110
DLN2=XZERO 001111
IF (NACT.EQ.0) GO TO 240 001112
DO 230 I=1,NACT 001113
T=T+RIGHT(I)**2 001114
DL(I)=XZERO 001115
230 CONTINUE 001116
240 NACT1=NACT+1 001117
IF (NACT.EQ.N) GO TO 260 001118
DO 250 I=NACT1,N 001119
DLM2=DLN2+RIGHT(I)**2 001120
DL(I)=RIGHT(I) 001121
250 CONTINUE 001122
260 T=T+DLM2 001123
IF (T.GT.XZERO.AND.DLM2.GT.EPS*EPS*T) GO TO 280 001124
S=(N+1)*ABS(C(NEW)) 001125
DO 270 I=1,N 001126
S=S+ABS(DC(NEW,I)*X(I))*(N+3-I) 001127
270 CONTINUE 001128
IF (CUP(NEW).LT.-EPS*S) GO TO 410 001129
KSTAT(NEW)=0 001130
GO TO 140 001131
280 CALL UTRNS (U,N,NACT,ACCUM,DL,W) 001132
C PROJECT GRADIENTS ON THE PROJECTED GRADIENT 001133
C 001134
C DO 300 I=LE1,LELI 001135
T=XZERO 001136
DO 290 J=1,N 001137
T=T+DL(J)*DC(I,J) 001138
290 CONTINUE 001139
DLDC(I)=T 001140
300 CONTINUE 001141
C CALCULATE STEP LENGTH "ANES" TO MAKE THE OBJECTIVE CONSTRAINT 001142
C EQUAL ZERO, AND CALCULATE THE STEP LENGTH "AMIN" TO THE 001143
C NEAREST INACTIVE CONSTRAINT UNDER CONSIDERATION 001144
C 001145
ANES=-CUP(NEW)/DLM2 001146
310 AMIN=XP73 001147
DO 320 I=LE1,LELI 001148
IF (KSTAT(I).NE.0) GO TO 320 001149
T=DLDC(I) 001150
IF (T.GE.XZERO) GO TO 320 001151
T=-CUP(I)/T 001152
IF (T.GT.AMIN) GO TO 320 001153
AMIN=T 001154
K=I 001155
320 CONTINUE 001156
C WILL THE OBJECTIVE CONSTRAINT GET ACTIVE ? 001157
C IF NOT, MAKE ACTIVE THE CLOSEST 001158
C 001159
OBJECT=ANES.LE.AMIN 001160
ALFA=AMIN1(AMIN,ANES) 001161
NACT1=NACT+1 001162
IF (OBJECT) GO TO 350 001163
DO 330 I=1,N 001164
R(I,NACT1)=DC(K,I) 001165
330 CONTINUE 001166
CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W) 001167
001168
001169
001170

```
CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.TRUE.,EPS)          001171
IF (NACT1.EQ.NACT) GO TO 340                               001172
KSTAT(K)=2                                                 001173
GO TO 310                                                 001174
340 KSTAT(K)=1                                             001175
KSET(NACT)=K                                              001176
C
C      TAKE THE STEP
C
350 IF (ALFA.EQ.XZERO) GO TO 380                           001177
DO 360 I=1,N                                               001178
X(I)=X(I)+ALFA*DLC(I)                                     001179
360 CONTINUE                                              001180
DO 370 I=LE1,LELI                                         001181
T=CUP(I)+ALFA*DLDG(I)                                     001182
IF (KSTAT(I).EQ.-1.AND.T.GE.XZERO) KSTAT(I)=0             001183
CUP(I)=T                                                 001184
370 CONTINUE                                              001185
380 IF (.NOT.OBJECT) GO TO 170                            001186
C
C      ACTIVATE THE OBJECTIVE CONSTRAINT
C
DO 390 I=1,N                                               001187
R(I,NACT1)=RIGHT(I)                                       001188
390 CONTINUE                                              001189
CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.FALSE.,EPS)       001190
IF (NACT.EQ.NACT1) GO TO 400                            001191
KSTAT(NEW)=0                                              001192
GO TO 140                                                 001193
400 KSET(NACT)=NEW                                         001194
GO TO 140                                                 001195
C
C      NO FEASIBLE POINTS
C
410 IFALL=3                                              001196
RETURN                                                 001197
END                                                   001198
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,DLDL,CUP,DLDG,KSTATF,KSTATC,W,SEPS,ACCUM,FMAX
2,IFALL)                                                 001199
C
THE SUBROUTINE SOLVES A LINEARLY CONSTRAINED LINEAR MINIMAX
PROBLEM. THE STARTING POINT MUST BE FEASIBLE.                001200
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), DLDL(M), CUP(IC), DLDG(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
LEL1=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
001201
001202
001203
001204
001205
001206
001207
001208
001209
001210
001211
001212
001213
001214
001215
001216
001217
001218
001219
001220
001221
001222
001223
001224
001225
001226
001227
001228
001229
001230
001231
001232
001233
001234
001235
```

```
DO 20 I=1,M          001236
KSTATF(I)=0          001237
T=F(I)              001238
IF (T.LE.FMAX) GO TO 20 001239
FMAX=T              001240
KSET0=I              001241
20 FUP(I)=T          001242
IF (LELI.EQ.0) GO TO 40 001243
DO 30 I=1,LELI      001244
KSTATC(I)=0          001245
CUP(I)=C(I)          001246
30 CONTINUE          001247
C
C   ACTIVATE INITIAL ACTIVE SET
C
40 NACTIN=NACT      001248
NACT=0               001249
IF (LE.EQ.0) GO TO 70 001250
DO 60 I=1,LE         001251
DO 50 J=1,N          001252
RC(J,I)=DC(I,J)     001253
50 CONTINUE          001254
KSETC(I)=I           001255
KSTATC(I)=1          001256
60 CONTINUE          001257
CALL ADDCL (U,R,N,NACT,LE,RIGHT,W,ACCUM,.FALSE.,EPS) 001258
IF (NACT.EQ.LE) GO TO 70 001259
XN=XZERO             001260
IFALL=3              001261
RETURN               001262
70 IF (NACTIN.LT.LE1) GO TO 100 001263
DO 90 K=1,NACTIN    001264
KK=KSET(K)           001265
IF (KK.LT.LE1.OR.KK.GT.LELI) GO TO 90 001266
NACT1=NACT+1         001267
IF (NACT1.GT.N) GO TO 100 001268
DO 80 I=1,N          001269
RC(I,NACT1)=DC(KK,I) 001270
80 CONTINUE          001271
CALL UTTRNS (U,N,NACT,ACCUM,RC(1,NACT1),W) 001272
CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.FALSE.,EPS) 001273
IF (NACT.LT.NACT1) GO TO 90 001274
EPS=EPS+SEPS        001275
KSET(NACT1)=KK       001276
KSTATC(KK)=1          001277
90 CONTINUE          001278
C
C   TRANSFORM OBJECTIVE FUNCTION GRADIENT
C
100 KSTATF(KSET0)=1  001279
DO 110 J=1,N          001280
RIGHT(J)=-DF(KSET0,J) 001281
110 CONTINUE          001282
KSET0=KSET0+LELI     001283
CALL UTTRNS (U,N,NACT,ACCUM,RIGHT,W) 001284
C
C   ITERATIVE LOOP
C
C   CALCULATE MULTIPLIERS AND FIND THE LARGEST
C
120 ASET0=-XONE      001285
IF (NACT.EQ.0) GO TO 240 001286
CALL RSOLV (R,N,NACT,RIGHT,ASET) 001287
IF (NACT.EQ.LE) GO TO 240 001288
AMAX=-XP73            001289
001290
001291
001292
001293
001294
001295
001296
001297
001298
001299
001300
```

```
DO 130 I=LE1,NACT          001301
IF (KSET(I).GT.LELI) ASET0=ASET0-ASET(I) 001302
IF (ASET(I).LE.AMAX) GO TO 130 001303
K= I
AMAX= ASET(I) 001304
130 CONTINUE 001305
IF (AMAX.LT.XZERO.AND.ASET0.LT.XZERO) GO TO 240 001306
IF (AMAX.GT.ASET0) GO TO 180 001307
C
C      CHANGE OBJECTIVE FUNCTION 001308
C
DO 140 I=LE1,NACT          001309
IF (KSET(I).LE.LELI) GO TO 140 001310
K= I
GO TO 150 001311
140 CONTINUE 001312
150 DO 170 I=1,K           001313
T=R(I,K)
IF (K.EQ.NACT) GO TO 170 001314
K1=K+1
DO 160 J=K1,NACT          001315
IF (KSET(J).GT.LELI) R(I,J)=R(I,J)-T 001316
160 CONTINUE 001317
170 RIGHT(I)=RIGHT(I)+T 001318
KK=KSET0 001319
KSET0=KSET(K) 001320
KSET(K)=KK 001321
C
C      DELETE ACTIVE CONSTRAINT NUMBER K 001322
C
180 KK=KSET(K) 001323
IF (KK.GT.LELI) KSTATF(KK-LELI)=0 001324
IF (KK.LE.LELI) KSTATC(KK)=0 001325
IF (ACCUM) GO TO 190 001326
ACCUM=.TRUE. 001327
CALL HACUM (U,N,NACT,W) 001328
190 CALL DELCL (K,U,R,N,NACT,RIGHT,.TRUE.) 001329
EPS=EPS+SEPS 001330
IF (K.GT.NACT) GO TO 210 001331
DO 200 I=K,NACT          001332
KSET(I)=KSET(I+1) 001333
200 CONTINUE 001334
C
C      DELETE LINEAR DEPENDENCE LABELS 001335
C
210 DO 220 I=1,M          001336
IF (KSTATF(I).EQ.-2) KSTATF(I)=0 001337
220 CONTINUE 001338
IF (LI.EQ.0) GO TO 120 001339
DO 230 I=LE1,LELI 001340
IF (KSTATC(I).EQ.-2) KSTATC(I)=0 001341
230 CONTINUE 001342
GO TO 120 001343
C
C      IS THERE AN UNBOUNDED SOLUTION ? 001344
C
240 IF (NACT.EQ.N) GO TO 490 001345
C
C      CALCULATE THE PROJECTED GRADIENT 001346
C
K=NACT+1 001347
T=XZERO 001348
DLN2=XZERO 001349
DO 250 I=K,N 001350
DLN2=DLN2+RIGHT(I)**2 001351
250 CONTINUE 001352
GO TO 120 001353
C
C      IS THERE AN UNBOUNDED SOLUTION ? 001354
C
240 IF (NACT.EQ.N) GO TO 490 001355
C
C      CALCULATE THE PROJECTED GRADIENT 001356
C
K=NACT+1 001357
T=XZERO 001358
DLN2=XZERO 001359
DO 250 I=K,N 001360
DLN2=DLN2+RIGHT(I)**2 001361
250 CONTINUE 001362
GO TO 120 001363
C
C      IS THERE AN UNBOUNDED SOLUTION ? 001364
C
240 IF (NACT.EQ.N) GO TO 490 001365
C
```

DL(I)=RIGHT(I)	001366
250 CONTINUE	001367
IF (K.EQ.1) GO TO 270	001368
DO 260 I=1,NACT	001369
T=T+RIGHT(I)**2	001370
DL(I)=XZERO	001371
260 CONTINUE	001372
270 T=T+DLN2	001373
IF (T.GT.XZERO.AND.DLN2.GT.EPS*EPS*T) GO TO 280	001374
IFALL=2	001375
GO TO 490	001376
280 CALL UTRNS (U,N,NACT,ACCUM,DL,W)	001377
C	001378
C PROJECT GRADIENTS ON THE PROJECTED GRADIENT	001379
C	001380
DO 300 I=1,M	001381
T=XZERO	001382
DO 290 J=1,N	001383
T=T+DL(J)*DF(I,J)	001384
290 CONTINUE	001385
DLDI(I)=T	001386
300 CONTINUE	001387
IF (LELI.EQ.0) GO TO 330	001388
DO 320 I=1,LELI	001389
T=XZERO	001390
DO 310 J=1,N	001391
T=T+DL(J)*DC(I,J)	001392
310 CONTINUE	001393
DLDC(I)=T	001394
320 CONTINUE	001395
C	001396
C CALCULATE STEP LENGTH	001397
C	001398
330 SMINC=XP73	001399
IF (LI.EQ.0) GO TO 350	001400
DO 340 I=LE1,LELI	001401
IF (KSTATC(I).NE.0) GO TO 340	001402
T=DLDI(I)	001403
IF (T.GE.XZERO) GO TO 340	001404
T=-CUP(I)/T	001405
IF (T.GT.SMINC) GO TO 340	001406
NEWC=I	001407
SMINC=T	001408
340 CONTINUE	001409
350 SMINF=XP73	001410
K=KSET0-LELI	001411
T0=DLDI(K)	001412
F0=FUP(K)	001413
DO 360 I=1,M	001414
IF (KSTATF(I).NE.0) GO TO 360	001415
T=T0-DLDI(I)	001416
IF (T.GE.XZERO) GO TO 360	001417
T=(FUP(I)-F0)/T	001418
IF (T.GT.SMINF) GO TO 360	001419
SMINF=T	001420
NEWF=I	001421
360 CONTINUE	001422
STEP=AMIN1(SMINF,SMINC)	001423
C	001424
C IN CASE THE STEP IS TOO LONG REDUCE AND RETURN	001425
C	001426
S=STEP	001427
CALL LIMIT (XNMAX2,X,XN2,DL,DLN2,S,N)	001428
IF (S.EQ.STEP) GO TO 370	001429
IFALL=1	001430

STEP=S 001431
GO TO 440 001432
C 001433
C INCLUDE THE NEW FUNCTION/CONSTRAINT 001434
C 001435
370 NACT1=NACT+1 001436
KK0=KSET0-LELI 001437
IF (SMINF.LT.SMINC) GO TO 400 001438
DO 380 I=1,N 001439
R(I,NACT1)=DC(NEWC,I) 001440
380 CONTINUE 001441
CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W) 001442
CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.TRUE.,EPS) 001443
IF (NACT.EQ.NACT1) GO TO 390 001444
KSTATC(NEWC)=-2 001445
GO TO 330 001446
390 KSTATC(NEWC)=1 001447
KSET(NACT)=NEWC 001448
GO TO 430 001449
400 DO 410 I=1,N 001450
R(I,NACT1)=DF(KK0,I)-DF(NEWF,I) 001451
410 CONTINUE 001452
CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W) 001453
CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.TRUE.,EPS) 001454
IF (NACT.EQ.NACT1) GO TO 420 001455
KSTATF(NEWF)=-2 001456
GO TO 330 001457
420 KSTATF(NEWF)=1 001458
KSET(NACT)=NEWF+LELI 001459
430 EPS=EPS+SEPS 001460
IF (STEP.EQ.XZERO) GO TO 120 001461
C 001462
C TAKE THE STEP AND UPDATE LINEAR FUNCTIONS 001463
C 001464
440 FMAX=-XP73 001465
XN2=XZERO 001466
DO 450 I=1,N 001467
X(I)=X(I)+STEP*DL(I) 001468
XN2=XN2+X(I)**2 001469
450 CONTINUE 001470
DO 460 I=1,M 001471
T=FUP(I)+STEP*DLD(I) 001472
IF (T.GT.FMAX) FMAX=T 001473
FUP(I)=T 001474
460 CONTINUE 001475
IF (LELI.EQ.0) GO TO 480 001476
DO 470 I=1,LELI 001477
CUP(I)=CUP(I)+STEP*DLDCC(I) 001478
470 CONTINUE 001479
480 IF (IFALL.EQ.0) GO TO 120 001480
C 001481
C RETURN 001482
C 001483
490 XN=SQRT(XN2) 001484
NACT=NACT+1 001485
KSET(NACT)=KSET0 001486
ASET(NACT)=ASET0 001487
RETURN 001488
END 001489
C 001490
C SUBROUTINE LINSYS (A,B, IDIM, N, NR, EPS) 001491
C 001492
C THE SUBROUTINE SOLVES A SYSTEM OF LINEAR EQUATIONS 001493
C USING GAUSSIAN ELIMINATION. 001494
C 001495

C DIMENSION A(IDIM, IDIM), B(N)
C DATA XONE, XM50/1.0, 1.E-50/
C 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
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)
001496
001497
001498
001499
001500
001501
001502
001503
001504
001505
001506
001507
001508
001509
001510
001511
001512
001513
001514
001515
001516
001517
001518
001519
001520
001521
001522
001523
001524
001525
001526
001527
001528
001529
001530
001531
001532
001533
001534
001535
001536
001537
001538
001539
001540
001541
001542
001543
001544
001545
001546
001547
001548
001549
001550
001551
001552
001553
001554
001555
001556
001557
001558
001559
001560

80 CONTINUE 001561
90 B(I)=B(I)-S*B(K) 001562
NR=N1 001563
IF (ABS(A(N,N)).LT.2*N*EPS) RETURN 001564
C 001565
C A HAS FULL RANK 001566
C 001567
C NR=N 001568
C 001569
C BACK SUBSTITUTION 001570
C 001571
B(N)=B(N)/A(N,N) 001572
K=N 001573
DO 110 I=2,N 001574
K1=K 001575
K=K-1 001576
S=B(K) 001577
DO 100 J=K1,N 001578
S=S-A(K,J)*B(J) 001579
100 CONTINUE 001580
B(K)=S/A(K,K) 001581
110 CONTINUE 001582
120 RETURN 001583
END 001584
C 001585
C SUBROUTINE BFCS (B,N,Y,XX,W,SEPS) 001586
C 001587
C UPDATES A HESSIAN APPROXIMATION USING BFCS-FORMULA. 001588
C 001589
C 001590
DIMENSION B(N,N), Y(N), XX(N), W(N) 001591
DATA XZERO/0.0/ 001592
EPS=(N+10)*SEPS 001593
DO 20 I=1,N 001594
T=XZERO 001595
DO 10 J=1,N 001596
T=T+B(I,J)*XX(J) 001597
10 CONTINUE 001598
W(I)=T 001599
20 CONTINUE 001600
YXX=XZERO 001601
WXX=XZERO 001602
YN=XZERO 001603
XXN=XZERO 001604
WN=XZERO 001605
DO 30 I=1,N 001606
YN=YN+Y(I)**2 001607
XXN=XXN+XX(I)**2 001608
WN=WN+W(I)**2 001609
YXX=YXX+Y(I)*XX(I) 001610
WXX=WXX+W(I)*XX(I) 001611
30 CONTINUE 001612
YN=SQRT(YN) 001613
XXN=SQRT(XXN) 001614
WN=SQRT(WN) 001615
IF (YN.EQ.XZERO.OR.WN.EQ.XZERO.OR.XXN.EQ.XZERO) RETURN 001616
IF (ABS(YXX).LT.EPS*YN*XXN) RETURN 001617
IF (ABS(WXX).LT.EPS*WN*XXN) RETURN 001618
DO 40 I=1,N 001619
B(I,I)=B(I,I)+Y(I)**2/YXX-W(I)**2/WXX 001620
40 CONTINUE 001621
IF (N.EQ.1) RETURN 001622
DO 50 I=2,N 001623
I1=I-1 001624
DO 50 J=1,I1 001625

B(I,J)=B(I,J)+Y(I)*Y(J)/YXX-W(I)*W(J)/WXX 001626
50 B(J,I)=B(I,J) 001627
RETURN 001628
END 001629
001630
001631
001632
C SUBROUTINE ADDCL (U,R,N,KCOL,KNEW,RIGHT,W,ACCUM,LRIGHT,EPS) 001633
C UPDATES HOUSEHOLDER FACTORIZATION. 001634
C THE NEW COLUMNS MUST HAVE BEEN TRANSFORMED AS RIGHHAND SIDES. 001635
C
DIMENSION U(N,N), R(N,N), RIGHT(N), W(N) 001636
LOGICAL ACCUM,LRIGHT 001637
DATA XZERO/0.0/ 001638
K1=KCOL+1 001639
K2=KCOL+KNEW 001640
001641
C COLUMN LOOP STARTS HERE 001642
C
DO 170 K=K1,K2 001643
S=XZERO 001644
T=XZERO 001645
IF (K.EQ.1) GO TO 20 001646
KK=K-1 001647
DO 10 I=1,KK 001648
T=T+R(I,K)**2 001649
10 CONTINUE 001650
20 DO 30 I=K,N 001651
S=S+R(I,K)**2 001652
30 CONTINUE 001653
T=T+S 001654
T=SQRT(T) 001655
S=SQRT(S) 001656
001657
001658
001659
C RETURN IF THE NEW COLUMN DEPENDS LINEARLY ON THE 001660
PRECEDING COLUMNS 001661
C
IF (T.EQ.XZERO) RETURN 001662
IF (S.LT.T*EPS) RETURN 001663
001664
C PERFORM HOUSEHOLDER TRANSFORMATION 001665
C
TT=R(K,K) 001666
T=ABS(TT) 001667
ALFA=SQRT(S*(S+T)) 001668
BETA=-SIGN(S,TT) 001669
R(K,K)=BETA 001670
W(K)=(TT-BETA)/ALFA 001671
IF (K.EQ.N) GO TO 80 001672
KK=K+1 001673
DO 40 I=KK,N 001674
W(I)=R(I,K)/ALFA 001675
40 CONTINUE 001676
001677
001678
C TRANSFORM THE REMAINING COLUMNS 001679
C
IF (K.EQ.K2) GO TO 80 001680
DO 70 J=KK,K2 001681
T=XZERO 001682
DO 50 I=K,N 001683
T=T+W(I)*R(I,J) 001684
50 CONTINUE 001685
DO 60 I=K,N 001686
R(I,J)=R(I,J)-T*W(I) 001687
60 CONTINUE 001688
001689
001690

70 CONTINUE 001691
C 001692
C TRANSFORM THE RIGHTHAND SIDE 001693
C 001694
80 IF (.NOT.LRIGHT) GO TO 110 001695
T=XZERO 001696
DO 90 I=K,N 001697
T=T+W(I)*RIGHT(I) 001698
90 CONTINUE 001699
DO 100 I=K,N 001700
RIGHT(I)=RIGHT(I)-T*W(I) 001701
100 CONTINUE 001702
C ACCUMULATE THE TRANSFORMATIONS IN U 001703
C U MUST HAVE BEEN INITIALIZED 001704
C 001705
110 IF (ACCUM) GO TO 130 001706
DO 120 I=K,N 001707
U(I,K)=W(I) 001708
120 CONTINUE 001709
GO TO 170 001710
130 DO 160 I=1,N 001711
T=XZERO 001712
DO 140 J=K,N 001713
T=T+U(I,J)*W(J) 001714
140 CONTINUE 001715
DO 150 J=K,N 001716
U(I,J)=U(I,J)-T*W(J) 001717
150 CONTINUE 001718
160 CONTINUE 001719
170 KCOL=KCOL+1 001720
RETURN 001721
END 001722
001723
C SUBROUTINE DELCL (K, U, R, N, KCOL, RIGHT, LRIGHT) 001724
C 001725
C DELETES COLUMN NUMBER K IN THE FACTORIZED MATRIX. 001726
C K MUST SATISFY 1.LE.K.LE.KCOL 001727
C U MUST HAVE BEEN ACCUMULATED. 001728
C 001729
C DIMENSION U(N,N), R(N,N), RIGHT(N) 001730
LOGICAL LRIGHT 001731
C 001732
C DELETE COLUMN NUMBER K 001733
C 001734
C KCOL=KCOL-1 001735
IF (K.GT.KCOL) RETURN 001736
DO 10 J=K,KCOL 001737
J1=J+1 001738
DO 10 I=1,J1 001739
10 R(I,J)=R(I,J1) 001740
C TRANSFORM TO UPPER TRIANGULAR FORM 001741
C USING STANDARD GIVENS TRANSFORMATIONS 001742
C 001743
DO 60 KK=K, KCOL 001744
K1=KK+1 001745
X=R(KK, KK) 001746
Y=R(K1, KK) 001747
A=SQRT(X*X+Y*Y) 001748
C=X/A 001749
S=Y/A 001750
R(KK, KK)=C*X+S*Y 001751
IF (KK.EQ.KCOL) GO TO 30 001752
001753
001754
001755

```
DO 20 J=K1, KCOL          001756
X=R(KK,J)                001757
Y=R(K1,J)                001758
R(KK,J)=C*X+S*Y          001759
R(K1,J)=C*Y-S*X          001760
20 CONTINUE                 001761
30 IF (.NOT.LRIGHT) GO TO 40 001762
X=RIGHT(KK)               001763
Y=RIGHT(K1)               001764
RIGHT(KK)=C*X+S*Y         001765
RIGHT(K1)=C*Y-S*X         001766
C
C   ACCUMULATE THE TRANSFORMATIONS
C
40 DO 50 I=1,N             001767
X=U(I,KK)                 001768
Y=U(I,K1)                 001769
U(I,KK)=C*X+S*Y          001770
U(I,K1)=C*Y-S*X          001771
50 CONTINUE                 001772
60 CONTINUE                 001773
RETURN                     001774
END                         001775
C
C   SUBROUTINE UTRNS (U,N,KCOL,ACCUM,R,W)
C   TRANSFORM THE VECTOR R AS A RIGHHAND SIDE.
C   DIMENSION U(N,N), R(N), W(N)
C   LOGICAL ACCUM
C   DATA XZERO/0.0/
C
C   IF THE TRANSFORMATIONS HAVE BEEN ACCUMULATED
C   DO SIMPLE MATRIX-MULTIPLICATION
C   ELSE TRANSFORM RIGHHAND SIDES
C
IF (ACCUM) GO TO 40        001788
IF (KCOL.EQ.0) RETURN       001789
DO 30 K=1,KCOL             001790
T=XZERO                     001791
DO 10 I=K,N                 001792
T=T+R(I)*U(I,K)            001793
10 CONTINUE                  001794
DO 20 I=K,N                 001795
R(I)=R(I)-T*U(I,K)          001796
20 CONTINUE                  001797
30 CONTINUE                  001798
RETURN                      001799
40 DO 50 I=1,N             001800
W(I)=R(I)                   001801
50 CONTINUE                  001802
DO 70 K=1,N                 001803
T=XZERO                     001804
DO 60 I=1,N                 001805
T=T+U(I,K)*W(I)            001806
60 CONTINUE                  001807
R(K)=T                      001808
70 CONTINUE                  001809
RETURN                      001810
END                         001811
C
C   SUBROUTINE UTRNS (U,N,KCOL,ACCUM,R,W)
C
```

C TRANSFORM THE VECTOR R OPPOSITE A RIGHHAND SIDE. 001821
C 001822
C 001823
C 001824
C 001825
C 001826
C 001827
C 001828
C 001829
C 001830
C 001831
C 001832
C 001833
C 001834
C 001835
C 001836
C 001837
C 001838
C 001839
C 001840
C 001841
C 001842
C 001843
C 001844
C 001845
C 001846
C 001847
C 001848
C 001849
C 001850
C 001851
C 001852
C 001853
C 001854
C 001855
C 001856
C 001857
C 001858
C 001859
C 001860
C 001861
C 001862
C 001863
C 001864
C 001865
C 001866
C 001867
C 001868
C 001869
C 001870
C 001871
C 001872
C 001873
C 001874
C 001875
C 001876
C 001877
C 001878
C 001879
C 001880
C 001881
C 001882
C 001883
C 001884
C 001885

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 SUBROUTINE RSOLV (R,N,KCOL,RIGHT,X)
C PERFORM BACK SUBSTITUTION ON RIGHT.
C DIMENSION R(N,N), RIGHT(N), X(N)
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 SUBROUTINE TSOLV (R,N,KCOL,RIGHT,X)
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          001886
I1=I-1                  001887
T=RIGHT(I)              001888
DO 10 J=1,I1             001889
T=T-X(J)*R(J,I)         001890
10 CONTINUE               001891
X(I)=T/R(I,I)           001892
20 CONTINUE               001893
RETURN                  001894
END                     001895
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
C DIMENSION U(N,N), W(N)
C DATA XZERO,XONE/0.0,1.0/
C
C INITIALIZE USING LAST TRANSFORMATION
C
K1=KCOL+1                001896
DO 10 I=KCOL,N            001897
W(I)=U(I,KCOL)            001898
10 CONTINUE                 001899
DO 20 I=KCOL,N            001900
U(I,I)=XONE-W(I)**2       001901
20 CONTINUE                 001902
IF (KCOL.EQ.N) GO TO 40   001903
DO 30 I=K1,N               001904
I1=I-1                   001905
T=W(I)                   001906
DO 30 J=KCOL,I1            001907
S=-T*W(J)                 001908
U(I,J)=S                  001909
30 U(J,I)=S                001910
40 IF (KCOL.EQ.1) RETURN    001911
C
C ACCUMULATE REMAINING TRANSFORMATIONS
C
DO 100 KK=2,KCOL          001912
K=K1-KK                  001913
DO 50 I=K,N                001914
W(I)=U(I,K)                001915
50 CONTINUE                 001916
T=W(K)                   001917
KP1=K+1                  001918
U(K,K)=XONE-T*T           001919
DO 60 I=KP1,N              001920
U(I,K)=-T*W(I)            001921
60 CONTINUE                 001922
DO 90 L=KP1,N              001923
S=XZERO                  001924
DO 70 I=KP1,N              001925
S=S+W(I)*U(I,L)            001926
70 CONTINUE                 001927
U(K,L)=-T*S                001928
DO 80 I=KP1,N              001929
U(I,L)=U(I,L)-S*W(I)        001930
80 CONTINUE                 001931
90 CONTINUE                 001932
100 CONTINUE                001933
001934
001935
001936
001937
001938
001939
001940
001941
001942
001943
001944
001945
001946
001947
001948
001949
001950
```

```
      RETURN          001951
      END            001952
C
C      SUBROUTINE LIMIT (XNMAX2,X,XN2,P,PN2,ALFA,N)    001953
C      LIMIT THE STEP LENGTH ALFA.                      001954
C
C      DIMENSION X(N), P(N)                            001955
C      DATA XZERO/0.0/                                001956
C      XTP=XZERO                                     001957
C      DO 10 I=1,N                                    001958
C      XTP=XTP+X(I)*P(I)
10   CONTINUE                                     001959
      B=XTP/PN2                                    001960
      T=SQRT(B*B+(XNMAX2-XN2)/PN2)               001961
      AP=T-B                                       001962
      AM=-T-B                                      001963
      IF (ALFA.GT.AP) ALFA=AP                     001964
      IF (ALFA.LT.AM) ALFA=AM                     001965
      RETURN                                       001966
      END                                         001967
                                                001968
                                                001969
                                                001970
                                                001971
                                                001972
```

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.

