ANL-80-64

Hh. 1580

ANL-80-64

201
8/1/80 T.S.

MASTER

SOLUTION OF THE GENERAL NONLINEAR PROGRAMMING PROBLEM WITH SUBROUTINE VMCON

by

Roger L. Crane, Kenneth E. Hillstrom,
and Michael Minkoff

ARGONNE NATIONAL LABORATORY

ARGONNE NATIONAL LABORATORY, ARGONNE, ILLINOIS

Prepared for the U. S. DEPARTMENT OF ENERGY
under Contract W-31-109-Eng-38

Printed in the United States of America
Available from
National Technical Information Service
U. S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes
Printed copy: A03
Microfiche copy: A01

Distribution Category:
Mathematics and Computers
(UC-32)

ANL-80-64

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

SOLUTION OF THE GENERAL NONLINEAR PROGRAMMING PROBLEM WITH SUBROUTINE VMCON

Roger L. Crane*
Kenneth E. Hillstrom
Michael Minkoff

July 1980

This work was supported by the Applied Mathematical Sciences Research Program (KC-04-02) of the Office of Energy Research of the U.S. Department of Energy under Contract W-31-109-Eng-38.

* RCA David Sarnoff Research Center
Princeton, New Jersey

Table of Contents

Abstract

This report describes how to solve the general nonlinear programming problem by means of a subroutine called VMCON. VMCON uses an algorithm proposed and first implemented by M. J. D. Powell, based on earlier work of S-P Han. Powell's algorithm solves a sequence of positive definite quadratic programming subproblems. Each solution determines a direction in which a one-dimensional minimization is performed.

In developing this code, we have left Powell's basic algorithm unchanged. Changes in Powell's original implementation were made to make the program easier to use and maintain and to incorporate some recently developed LINPACK subprograms. The current implementation contains extensive in-line documentation; an interface subroutine, VMCON1, with a simplified calling sequence; and print options to aid the user in interpreting results.

SOLUTION OF THE GENERAL NONLINEAR PROGRAMMING PROBLEM
WITH SUBROUTINE VMCON

Roger L. Crane*
Kenneth E. Hillstrom
Michael Minkoff

Introduction

This report is addressed primarily to the person who wishes to solve a real problem by use of optimization methods but who is unsure of how to proceed. Several difficulties confront such a person. He must select an optimization routine, develop a mathematical model of the real problem in the form required by this routine, master numerous programming details, and ultimately interpret the program output.

A mathematical optimization subroutine does not directly enable a user to solve a real problem. A link must be created between the problem and the subroutine. This link, which we have called an application program, interfaces to the problem through a mathematical model and to the subroutine through a programming language. Figure 1 illustrates this relationship.

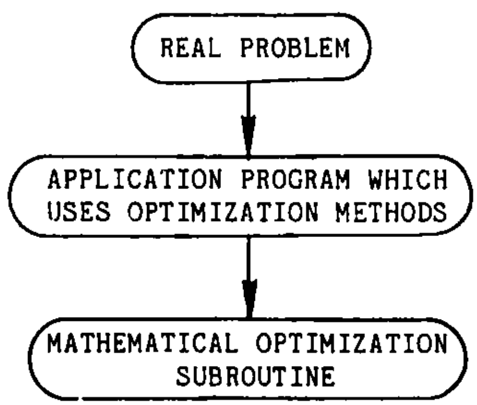

Figure 1. Solution of an Optimization Problem

---

* RCA David Sarnoff Research Center
Princeton, New Jersey

This report discusses how to create the link to a particular optimization subroutine, named VMCON. VMCON uses an algorithm proposed and first implemented by M. J. D. Powell of Cambridge University [1, 2], based on earlier work of S-P Han [3]. Powell's algorithm solves a sequence of positive definite quadratic programming subproblems. Each solution determines a direction in which a one-dimensional minimization is performed.

Methods based on the solution of quadratic subproblems represent only one class of techniques for solving the general nonlinear programming problem. Other approaches include penalty function methods, generalized reduced gradient methods, and augmented Lagrangian or multiplier methods. No one method nor one class of methods can be expected to solve all problems accurately and efficiently; each method has particular strengths and weaknesses. Some feeling for the variety of methods that have been proposed and implemented can be obtained by study of a recent survey paper of Lasdon and Waren [4].

Recent tests [5] have shown methods based on quadratic approximation to be especially efficient in terms of the number of function and gradient evaluations required. Powell's algorithm as implemented in the Harwell Library Subroutine VF02AD was included in these tests.

The test results and some direct experience in using Subroutine VF02AD motivated us to develop the implementation reported here. We have left Powell's original algorithm unchanged. We have, however, changed his implementation to make the program easier to use and maintain and to incorporate some recently developed LINPACK subprograms. The current implementation contains extensive in-line documentation; an interface subroutine, VMCON1, with a simplified calling sequence; and print options to aid the user in interpreting results.

Mathematical Problem

The general nonlinear programming problem, often called the nonlinear constrained optimization problem, can be stated as

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to} \\ &\quad c_i(x) = 0, \quad i=1,\ldots,k, \\ &\quad c_i(x) \geq 0, \quad i=k+1,\ldots,m, \end{aligned} \tag{1}$$

where the objective function, f, and the constraint functions, c_i, are functions of n variables. Subroutine VMCON solves problems of this form, where the functions f and c_i are assumed to be continuously differentiable. Some thoughts about how to formulate such problems will be given in the next section.

Unconstrained problems and linearly constrained problems are included in the general form (1) and can be solved with Subroutine VMCON. However, methods specifically designed to solve these simpler problems will generally be more efficient. Programs are available for the following special cases:

PROBLEM	OBJECTIVE FUNCTION	CONSTRAINTS
Unconstrained	General	None
Linear Programming	Linear	Linear
Quadratic Programming	Quadratic	Linear
Linearly Constrained	General	Linear

A solution of the general nonlinear programming problem can be characterized mathematically, and some understanding of this theory is necessary to use Subroutine VMCON effectively. In particular, one must appreciate the role of Lagrange multipliers in characterizing a solution. Let the Lagrangian function be defined as

$$L(x,\lambda) = f(x) - \sum_{i=1}^{m} \lambda_i c_i(x), \tag{2}$$

where the m parameters, λ_i, are called the Lagrange multipliers. If x^* is to be a solution of the general nonlinear programming problem, then it is necessary that there exist an associated set of values of the Lagrange multipliers, λ_i^*, such that

$$\nabla_x L(x^*,\lambda^*) = \nabla f(x^*) - \sum_{i=1}^{m} \lambda_i^* \nabla c_i(x^*) = 0, \qquad (3a)$$

$$\lambda_i^* \geq 0, \qquad i=k+1,\ldots,m, \qquad (3b)$$

$$\lambda_i^* c_i(x^*) = 0, \qquad i=1,\ldots,m, \qquad (3c)$$

$$c_i(x^*) = 0, \qquad i=1,\ldots,k, \qquad (3d)$$

$$c_i(x^*) \geq 0, \qquad i=k+1,\ldots,m, \qquad (3e)$$

provided that an additional regularity condition is imposed on the constraints. This condition, called the constraint qualification, may be defined in several different ways [6]. It is not easy to check in practice, however, and no attempt is made to do so in Powell's algorithm.

The necessary conditions shown in (3) are known as the Kuhn-Tucker conditions. Condition (3a), which states that the gradient of the Lagrangian must vanish, is a generalization of the condition in the unconstrained case that the gradient of the objective function must be zero. Condition (3b) states that each Lagrange multiplier corresponding to an inequality constraint must be nonnegative. No such requirement need be satisfied by the Lagrange multipliers that correspond to equality constraints. Condition (3c), called the complementarity condition, shows that at the solution, a constraint is satisfied at equality, or the associated Lagrange multiplier is zero, or both conditions hold. Finally, Conditions (3d) and (3e) ensure that the solution is feasible, i.e., that it satisfies all the constraints.

Powell's algorithm is an iterative method designed to converge to a point that satisfies these first-order necessary conditions. Theoretical results that define sufficient conditions for a solution of the general nonlinear programming problem are also available [6]. These results are not discussed in this report because Powell's algorithm does not attempt to satisfy sufficiency conditions.

In addition to their use in characterizing the necessary conditions (3), the Lagrange multipliers provide information about the sensitivity of the solution to perturbations in the constraints. In particular, λ_i^* gives the rate of change of the objective function, f, with respect to a change in the ith constraint [7]. The complementarity condition (3c), which states that $\lambda_i^* = 0$ if $c_i(x^*) > 0$, illustrates this result for an inequality constraint that is strictly satisfied. In this case, the constraint can be perturbed without changing the objective function.

Problem Formulation

As mentioned earlier, the person who wishes to use optimization methods to solve a real problem must construct a mathematical model in the form required by the chosen optimization program. For Subroutine VMCON, that form was defined in the preceding section. The task of problem formulation is obviously highly dependent on the specific application being considered. Because optimization methods are used to solve problems from a variety of application areas, one cannot discuss the process of problem formulation in the detail required for any specific application. Nevertheless, some general remarks may be useful.

One needs to consider first the primary purpose in using optimization methods for real applications. The most obvious purpose is to find the the minimum of a well defined constrained function accurately and efficiently. This is appropriate for the designer of an optimization subroutine. It is, however, unduly restrictive where real problems are to be solved. Few real problems are so simple or so well formulated and understood that they can be solved simply by minimizing a single function of n variables, even when one allows the solution to be constrained. In an applications environment, then, the primary purpose in using optimization methods is to enhance one's understanding of the real problem that is being modelled, not to compute optimal answers. Viewed in this context, a mathematical optimization program is a sophisticated tool to be used in an applications program for the solution of a real problem. Insights gained by using optimization methods with a particular mathematical model imbedded in an application program will often lead to a revision of the model that better approximates the real problem under consideration. Ultimately, the precise minimum may be of interest.

The first steps in actually formulating a real problem so that it can be studied by the use of optimization methods do not depend upon the particular optimization subroutine which is to be used. They depend only upon the problem to be solved. As an aid in comprehending and evaluating the ideas that follow, the reader should have some particular problem or class of problems in mind. For example, one might choose to think of engineering design problems. First, one should carefully enumerate the properties that a useful solution of the problem would possess. When this list is completed, it will contain the potential candidates for the objective function and the inequality constraint functions that must be supplied to any subroutine for constrained optimization. Next, one should list all those quantities upon which these desired characteristics depend. This list will contain the items that may ultimately be associated with optimization variables, i.e., with the vector x in the mathematical problem (1).

The formulation process, up to this point, could be very qualitative. To proceed further, however, one must start to think quantitatively. For each item in the list of properties possessed by a useful solution, one must attempt to find a computable measure, i.e., some mathematical function, by which that property can be judged. The measure is most useful if it is monotone in the sense that as its value decreases (or increases), the property is judged to be more desirable. This is the most difficult part of the formulation process. Perhaps some of the desirable properties cannot be characterized mathematically, or some of the factors that influence the solution cannot be quantified. Even if neither of these difficulties arise, it may be best to ignore, at least initially, some of the desirable properties or some of the factors that influence the solution. One should attempt to formulate the simplest model that has a reasonable chance of adequately describing the real problem being considered. An unduly complicated model is more difficult to solve and as a result may not lead to increased understanding of the real problem. Most of the skills required to construct useful mathematical models come from one's formal education and practical experience in the application area of interest. One can also profit by studying some cases in which mathematical models of real problems have been constructed so that optimization methods can be applied [8, 9].

At this point in the formulation process, one has a set $S = \{s_i(x)\}$ of mathematical functions. Each $s_i(x)$ measures one aspect of the quality of solution to the real problem of interest. The objective and inequality constraint functions of (1) will be constructed from this set. Equality constraints will be considered later. The form of (1) is somewhat restrictive in that only one of the $s_i(x)$ can be selected for the objective function; i.e., only one function can be minimized, and the others must be regarded as constraints. Note that it is no restriction to think only of minimizing $s_i(x)$ because, if a maximum is desired, one could use the function $-s_i(x)$. Assume that the jth function is selected for minimization, i.e., that $f(x) = s_j(x)$ in (1). Now, the constraint functions $c_i(x)$ of (1) for $i \neq j$ can be defined by introducing constants t_i such that

$$c_i(x) = s_i(x) - t_i \geq 0.$$

Inequalities of the opposite sense can be handled easily by using the negative of $s_i(x)$. Each constant, t_i, sets the level of the associated quality measure that will be regarded as satisfactory in attempting to solve the constrained problem. Some experimentation with these constants may be necessary in solving the real problem. The final step in formulating the problem is to identify any relations among the optimization variables or functions of those variables which result in equality constraints. Such functions can then be used directly to define the equality constraints of (1).

Overview of Powell's Algorithm

In this section, we provide enough information about Powell's algorithm to enable a user of VMCON to interpret the subroutine's output. A more complete description of the underlying theory can be found in Reference [1]. In the course of solving real problems, one often inadvertently formulates incomplete mathematical models. It is common, for example, to assume implicitly that certain variables or functions will remain positive or not tend to infinity when it is known from physical arguments that such will be the case for the real problem. The conditions may not hold, however, if the model is inadequate to describe the real problem. Thus, one must understand enough about the optimization algorithm to be able to judge whether the results produced by the subroutine are due to the nature of the real problem, to the mathematical model, or perhaps to the algorithm itself. Such judgments are often easier to make if one requests subroutine output during the course of the solution rather than waiting until the iterative process either converges or diverges. The following description of Powell's algorithm provides some background to help the user of Subroutine VMCON decide what output to request.

An iterative procedure is used in VMCON to solve Problem (1). Two major tasks are performed during each iteration. First, a positive definite quadratic programming problem is solved; then a one-dimensional minimization is performed, as illustrated in the simple flow chart of Fig. 2.

The solution of the quadratic programming problem provides estimates of the Lagrange multipliers and also determines a search direction for use in the subsequent one-dimensional minimization. The function that is minimized balances the two competing goals which result from the desire to decrease the objective function while reducing the amount by which the constraints fail to be satisfied. The solution of this minimization problem produces a revised estimate of the solution of (1).

A positive definite quadratic programming problem is a problem of form (1), where f is a positive definite quadratic function and where the constraints are linear functions. The quadratic function in Powell's algorithm is obtained by approximating the Lagrangian function (2), and the constraints for the quadratic programming problem are obtained by linearizing the constraints of (1) about the current solution estimate, x^{j-1}. Here, and below, a superscript is used to denote the iteration on which a quantity is computed.

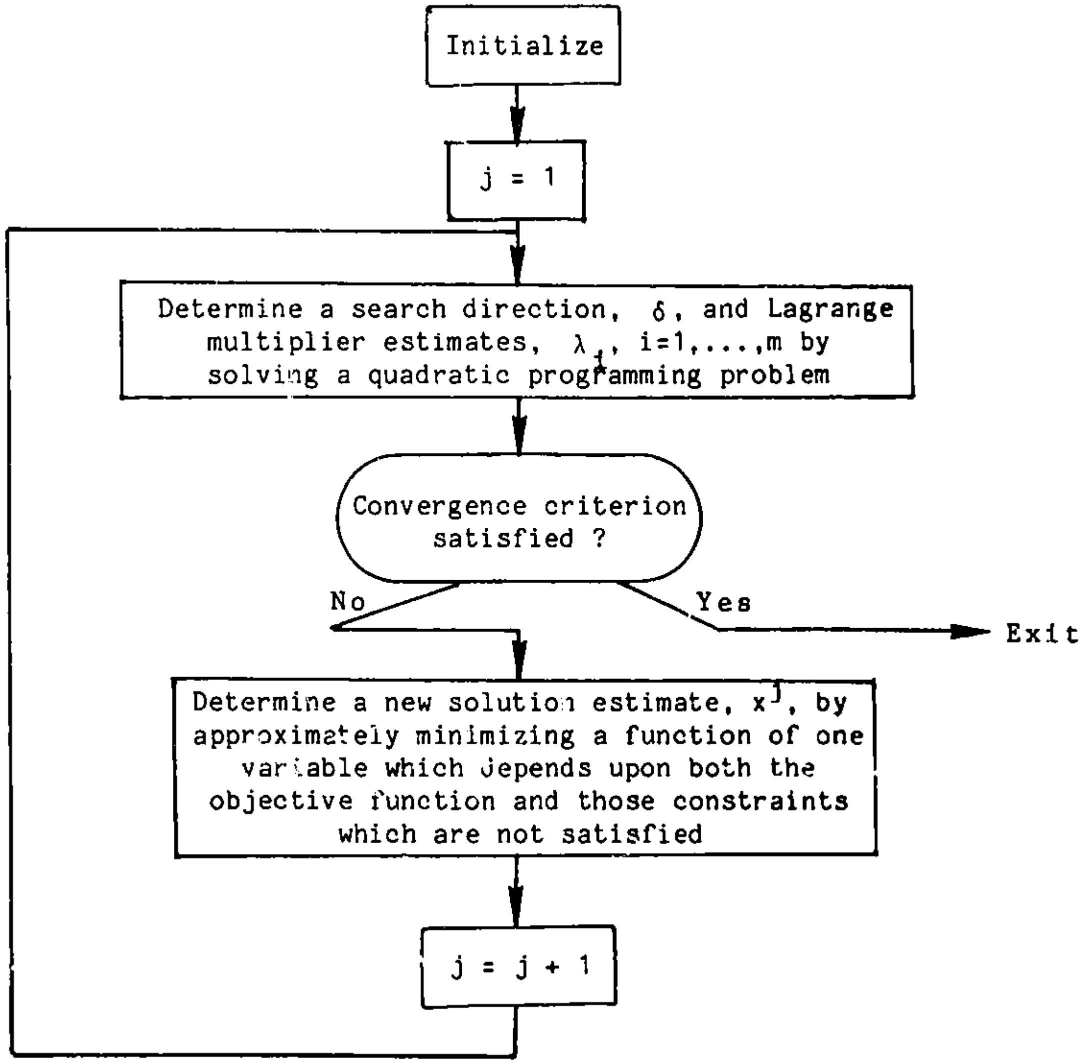

Figure 2. Simplified Flow Chart of Algorithm Used in VMCON

The quadratic programming problem to be solved at each iteration can be reduced to the form

$$\text{minimize } Q(\delta) = f(x^{j-1}) + \delta^T \nabla f(x^{j-1}) + (1/2)\delta^T B(x^{j-1}, \lambda^{j-1})\delta \ ,$$

$$\text{subject to} \qquad (4)$$

$$\nabla c_i^T(x^{j-1})\delta + c_i(x^{j-1}) = 0, \quad i=1,\dots,k,$$

$$\nabla c_i^T(x^{j-1})\delta + c_i(x^{j-1}) \geq 0, \quad i=k+1,\dots,m.$$

The solution of (4) on the jth iteration is denoted below by δ^j, and the Lagrange multipliers generated by solving (4) are denoted by λ_i^j, i=1,...,m. A quadratic approximation in x of the Lagrangian about x^{j-1} has the form

$$Q(x) = L(x^{j-1},\lambda^{j-1}) + (x - x^{j-1})^T \nabla_x L(x^{j-1},\lambda^{j-1}) \tag{5}$$
$$+ (1/2)(x - x^{j-1})^T \nabla_{xx} L(x^{j-1},\lambda^{j-1})(x - x^{j-1}).$$

The simplified form of $Q(\delta)$ in (4) follows from (5) by first expressing $L(x,\lambda)$ as given by (2), by expressing $\nabla_x L(x,\lambda)$ similarly, and then by making use of the constraint relations to simplify the first two terms. Finally, one identifies $B(x^{j-1}, \lambda^{j-1})$ as an approximation to $\nabla_{xx}L(x^{j-1},\lambda^{j-1})$ and sets $\delta = x - x^{j-1}$. As indicated by the notation, the matrix B changes from iteration to iteration. The initialization of this matrix and the strategy used to revise it will be discussed later.

One should note the qualitative relations between the solution of the quadratic programming problem and the necessary conditions (3) of the general problem. Condition (3a) states that the gradient of the Lagrangian for the general problem must vanish at a solution. The simplified form of $Q(\delta)$ is an approximation to the Lagrangian, as shown above. To the extent that this approximation is valid and that linear approximations to the constraints are valid, the solution of the quadratic programming problem approximates the solution of the general problem. Powell [2] points out the need to supplement this basic idea with some technique that tends to force convergence from poor starting approximations. The one-dimensional minimization is introduced for this purpose.

The form of the function that is minimized in the line search on the jth iteration is

$$\phi(\alpha) = \psi(x,\mu) = f(x) + \sum_{i=1}^{k} \mu_i |c_i(x)|$$
$$+ \sum_{i=k+1}^{m} \mu_i |\min(0, c_i(x))| \,, \tag{6}$$

$$\text{where } x = x^{j-1} + \alpha\delta^j \quad \text{and } \mu_i \geq 0.$$

The latter two terms in (6) are weighted sums of the absolute constraint violations. The weights used in VMCON are

$$\mu_i = |\lambda_i^1|$$

for the first iteration and

$$\mu_i = \max\left[|\lambda_i^j|, \frac{1}{2}\left(\mu_i^{j-1} + |\lambda_i^j|\right)\right]$$

on subsequent iterations. This choice of weights is motivated by theoretical results on convergence derived by Han [3] and by numerical experiments performed by Powell [1, 2].

An iterative procedure based on quadratic approximations is used to determine an approximate minimum of (6). Details are given in Ref. [1]. A maximum limit of ten steps is allowed for the minimization in this implementation of VMCON. Powell's original implementation had a limit of five. Little change in efficiency was noted in solving several test problems with the increased limit, and a problem of premature termination of the algorithm was eliminated. The value of the solution estimate to be used for the next iteration is defined as $x^j = x^{j-1} + \alpha^j \delta^j$, where α^j is the value of α determined by the linear search procedure above.

Upon completion of the line search, the information required to revise the estimate of the second derivative of the Lagrangian is available. The information is in the form of two differences

$$\xi = x^j - x^{j-1} \quad \text{and} \quad \gamma = \nabla_x L(x^j,\lambda^j) - \nabla_x L(x^{j-1},\lambda^j). \tag{7}$$

The method used to revise the Hessian estimate is based on the BFGS Quasi-Newton update formula

$$B_{NEW} = B - \frac{B\xi\xi^T B}{\xi^T B\xi} + \frac{\gamma\gamma^T}{\xi^T\xi} \tag{8}$$

which is widely used for unconstrained minimization. Here, the superscripts indicating iteration dependence have been dropped for simplicity. For the constrained problem, γ is modified to ensure that the revised matrix remains positive definite. The method suggested by Powell [1,2] is to replace γ with

$$\theta\gamma + (1-\theta)B\xi , \tag{9}$$

where $0 \leq \theta \leq 1$ is defined by

$$\theta = \begin{cases} 1 & \xi^T\gamma \geq 0.2\xi^T B\xi \\ \dfrac{0.8\xi^T B\xi}{\xi^T B\xi - \xi^T\gamma} & \xi^T\gamma < 0.2\xi^T B\xi \end{cases} . \tag{10}$$

This completes the discussion of the estimation of $\nabla_{xx}L(x,\lambda)$ except for the specification of the initial estimate, $B(x^0,\lambda^0)$. This estimate is normally taken to be be the identity matrix. Because the algorithm depends on the scaling of the initial Hessian estimate, however, a constant multiple of the identity matrix may be preferable for some problems. If better information is available for estimating the Hessian initially (for example, information obtained by solving closely related problems), its use may improve both the reliability and the efficiency of the iterative algorithm.

As indicated in Fig. 2, a convergence test is made on each iteration after the quadratic programming problem is solved. The algorithm terminates if the condition

$$\left|\nabla f(x^{j-1})^T \cdot \delta^j\right| + \sum_{i=1}^{m} \left|\lambda_i^j \, c_i(x^{j-1})\right| < \varepsilon \tag{11}$$

is satisfied, where ε is a user-supplied error tolerance. The first term is the predicted change in magnitude of the objective function if another line search is performed, and the second term is a measure of the complementarity error, i.e., the amount by which necessary condition (3c) fails to be satisfied. Thus, if the change in the objective function and the complementarity error are sufficiently small, x^{j-1} is accepted as the solution of (1).

Use of Subroutines VMCON and VMCON1

In this section the programming details required for the use of VMCON and VMCON1 are discussed. In each of the subroutines supplied, all floating-point variables are declared DOUBLE PRECISION (REAL*8). The subroutines have been compiled with the IBM G1 and H Extended Fortran compilers and have been tested on IBM Model 370/168 370/195, and 3033 processors. Modified versions of two Harwell library programs, VE02AD [10] and LA02AD [11], are included. These subroutines are used to solve the quadratic programming subproblems described above and were modified to use some recently developed LINPACK [12] and Basic Linear Algebra [13] subprograms. No changes were made to the algorithms defined in Refs. [10] and [11].

Both VMCON and VMCON1 solve the general nonlinear programming problem (1). VMCON1 provides an interface to VMCON with a simplified calling sequence. Details regarding variable names and calling sequences for both programs are provided by extensive comment statements included in the Fortran code. These are also included below.

To use either VMCON or VMCON1, the user must supply a Fortran subroutine that computes $f(x)$; $\nabla f(x)$; $c_i(x)$; $\nabla c_i(x)$, $i=1,\ldots,m$ of (1) given the vector x; the number of variables, n; and the number of constraints, m. Some consideration must be given to the proper scaling of these functions. There is a somewhat artificial restriction on the vector x. The subroutine used to solve the quadratic programming subproblems requires that upper and lower bounds be specified for the independent variables, x_i, $i=1,\ldots,n$. The values used in this implementation are set in Subroutine QPSUB as $\pm 10^6$. An error indicator is set in VMCON if the solution of (1) is restricted by one of these artificial bounds. The initial estimate of the second derivative matrix discussed in the preceding section is related to the scaling of f and ∇f. Because the identity matrix is used as an initial estimate unless an estimate is provided by the user, the functions f and ∇f should be scaled to have magnitudes near unity Although it is not crucial to the performance of the algorithm, it is also advisable to scale the constraints and constraint gradients to be of order one as an aid in interpreting intermediate subroutine output.

All additional information required to define the problem and to control the execution of VMCON or VMCON1 is supplied through the argument list. No COMMON storage is used. For most problems, VMCON1 is recommended because it is simpler to use. If, however, one desires to specify an initial estimate of the second derivative matrix or to specify an upper limit on the number of function evaluations other than the limit of 100*(n+1) used in VMCON1, then VMCON must be called directly. The options for selecting intermediate subroutine output are the same for both programs. The information discussed in the preceding section is useful in determining what output to request.

A variable, INFO, is set by VMCON1 or VMCON and returned to the user to indicate normal or abnormal termination. A brief description of the conditions identified by INFO is included with the in-line documentation below. The factors that result in values of 0, 1, or 2 for INFO are clear. The remaining cases require some interpretation.

Values of 3 or 4 for INFO are most likely to occur because the results produced by evaluating the user-supplied subroutine FCN which computes $f(x)$, $\nabla f(x)$, $c_i(x)$, and $\nabla c_i(x)$ are inconsistent. It may be that subroutine FCN has been coded incorrectly and that the algorithm has not been able to make substantial progress. Alternatively, the solution may have proceeded to a point where noise in the functions has produced difficulty. This noise could be due to roundoff errors or perhaps to limited precision in the computation of the functions. The latter case can occur when the functions are evaluated by solving differential equations or evaluating integrals which attempt to satisfy a user-supplied error tolerance [14].

An INFO value of 5 is most likely to occur because there is no feasible solution to the nonlinear problem (1). An illustration of this case is given in the examples later in this report. However, it is also possible that the linearized constraints in the quadratic programming subproblem have no solution even though there does exist a feasible solution to the nonlinear problem. If this difficulty is suspected and the subroutine has terminated close to the starting point, other initial solution estimates should be considered. It may be, however, that the starting estimate is reasonable but the algorithm has taken an inappropriately large initial step. This can occur when the initial estimate of the second derivative matrix is poor. Use of a better initial Hessian estimate, often simply a constant multiple of the identity matrix, may result in a more reasonable initial step and ultimate convergence to a solution.

INFO is set to 6 if a singular matrix is encountered in solving a quadratic programming subproblem or if the solution of the subproblem is restricted by an artificial bound as discussed earlier in this section.

***** Use of Subroutine VMCON1 *****

```
      SUBROUTINE VMCON1(FCN,N,M,MEQ,X,OBJF,FGRD,CONF,CNORM,LCNORM,
     1                  VLAM,TOL,IPRINT,NWRITE,INFO,WA,LWA,IWA,LIWA)
      INTEGER N,M,MEQ,LCNORM,IPRINT,NWRITE,INFO,LWA,LIWA
      INTEGER IWA(LIWA)
      DOUBLE PRECISION OBJF,TOL
      DOUBLE PRECISION X(N),FGRD(N),CONF(M),CNORM(LCNORM,M),VLAM(M),
     1                 WA(LWA)
      EXTERNAL FCN
C     **********
C
C     SUBROUTINE VMCON1
C
C     THIS SUBROUTINE CALCULATES THE LEAST VALUE OF A FUNCTION OF
C     SEVERAL VARIABLES SUBJECT TO LINEAR AND/OR NONLINEAR EQUALITY
C     AND INEQUALITY CONSTRAINTS.  MORE PARTICULARLY, IT SOLVES THE
C     PROBLEM
C
C               MINIMIZE F(X)
C
C        SUBJECT TO C (X) =  0.0 ,   I = 1,...,MEQ
C                    I
C
C               AND C (X) >= 0.0 ,   I = MEQ+1,...,M
C                    I
C
C     THE SUBROUTINE IMPLEMENTS A VARIABLE METRIC METHOD FOR
C     CONSTRAINED OPTIMIZATION DEVELOPED BY M.J.D. POWELL.
C
C
C     THE SUBROUTINE STATEMENT IS
C
C       SUBROUTINE VMCON1(FCN,N,M,MEQ,X,OBJF,FGRD,CONF,CNORM,LCNORM,
C                         VLAM,TOL,IPRINT,NWRITE,INFO,WA,LWA,IWA,LIWA)
C
C     WHERE
C
C       FCN IS THE NAME OF THE USER SUPPLIED SUBROUTINE WHICH
C         CALCULATES THE OBJECTIVE AND CONSTRAINT FUNCTIONS, AND THE
C         GRADIENTS (FIRST DERIVATIVE VECTORS) OF THE OBJECTIVE AND
C         CONSTRAINT FUNCTIONS. FCN SHOULD BE DECLARED IN AN EXTERNAL
C         STATEMENT IN THE USER CALLING PROGRAM, AND SHOULD BE WRITTEN
C         AS FOLLOWS
C
C         SUBROUTINE FCN(N,M,X,OBJF,FGRD,CONF,CNORM,LCNORM,INFO)
C         INTEGER N,M,LCNORM,INFO
C         DOUBLE PRECISION OBJF
C         DOUBLE PRECISION X(N),FGRD(N),CONF(M),CNORM(LCNORM,M)
C         ---------------
C         STATEMENTS TO CALCULATE THE OBJECTIVE AND CONSTRAINT
C         FUNCTIONS AND THE GRADIENTS OF THE OBJECTIVE AND CONSTRAINT
C         FUNCTIONS AT X. THE OBJECTIVE AND CONSTRAINT FUNCTIONS AND
C         THE GRADIENT OF THE OBJECTIVE FUNCTION MUST BE RETURNED IN
```

```
C       OBJF, CONF AND FGRD RESPECTIVELY. NOTE THAT THE EQUALITY
C       CONSTRAINTS MUST PRECEDE THE INEQUALITY CONSTRAINTS IN CONF.
C       THE CONSTRAINT GRADIENTS OR NORMALS MUST BE RETURNED AS THE
C       COLUMNS OF CNORM.
C       ----------
C       RETURN
C       END
C
C       THE VALUE OF INFO SHOULD NOT BE CHANGED BY FCN UNLESS THE
C       USER WANTS TO TERMINATE EXECUTION OF VMCON1. IN THIS CASE
C       SET INFO TO A NEGATIVE INTEGER.
C
C     N IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE NUMBER OF
C       VARIABLES.
C
C     M IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE NUMBER OF
C       CONSTRAINTS.
C
C     MEQ IS A NON-NEGATIVE INTEGER INPUT VARIABLE SET TO THE NUMBER
C       OF EQUALITY CONSTRAINTS. MEQ MUST BE LESS THAN OR EQUAL TO N.
C
C     X IS A REAL*8 ARRAY OF LENGTH N. ON INPUT IT MUST CONTAIN AN
C       INITIAL ESTIMATE OF THE SOLUTION VECTOR. ON OUTPUT X
C       CONTAINS THE FINAL ESTIMATE OF THE SOLUTION VECTOR.
C
C     OBJF IS A REAL*8 OUTPUT VARIABLE THAT CONTAINS THE VALUE OF THE
C       OBJECTIVE FUNCTION AT THE OUTPUT X.
C
C     FGRD IS A REAL*8 OUTPUT ARRAY OF LENGTH N WHICH CONTAINS THE
C       COMPONENTS OF THE GRADIENT OF THE OBJECTIVE FUNCTION AT
C       THE OUTPUT X.
C
C     CONF IS A REAL*8 OUTPUT ARRAY OF LENGTH M WHICH CONTAINS THE
C       VALUES OF THE CONSTRAINT FUNCTIONS AT THE OUTPUT X. THE
C       EQUALITY CONSTRAINTS PRECEDE THE INEQUALITY CONSTRAINTS.
C
C     CNORM IS A REAL*8 LCNORM BY M ARRAY WHOSE COLUMNS CONTAIN THE
C       CONSTRAINT NORMALS AT THE OUTPUT X IN THE FIRST N POSITIONS.
C
C     LCNORM IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE ROW
C       DIMENSION OF CNORM WHICH IS AT LEAST N+1.  THE (N+1)ST ROW
C       OF CNORM IS USED FOR WORK SPACE.
C
C     VLAM IS A REAL*8 OUTPUT ARRAY OF LENGTH M WHICH CONTAINS THE
C       LAGRANGE MULTIPLIERS AT THE OUTPUT X.  THE LAGRANGE
C       MULTIPLIERS PROVIDE THE SENSITIVITY OF THE OBJECTIVE
C       FUNCTION TO CHANGES IN THE CONSTRAINT FUNCTIONS.
C
C     TOL IS A NONNEGATIVE REAL*8 INPUT VARIABLE. A NORMAL RETURN
C       OCCURS WHEN THE OBJECTIVE FUNCTION PLUS SUITABLY WEIGHTED
C       MULTIPLES OF THE CONSTRAINT FUNCTIONS ARE PREDICTED TO
C       DIFFER FROM THEIR OPTIMAL VALUES BY AT MOST TOL.
C
C     IPRINT IS AN INTEGER INPUT PARAMETER WHICH CONTROLS THE PRINTED
```

```
C       OUTPUT FROM VMCON1.  IT SHOULD BE SET AS FOLLOWS
C
C       IPRINT <= 0  NO OUTPUT
C
C       IPRINT  = 1  FOR EACH QUADRATIC SUBPROBLEM, X, OBJF, AND
C                    THE NORM OF THE LAGRANGIAN GRADIENT ARE OUTPUT
C
C       IPRINT  = 2  OUTPUT ABOVE PLUS THE SEARCH DIRECTION AND THE
C                    LAGRANGE MULTIPLIERS FROM THE QUADRATIC SUB-
C                    PROBLEM, AND THE MULTIPLIERS FOR THE LINE
C                    SEARCH
C
C       IPRINT  = 3  OUTPUT ABOVE PLUS LINE SEARCH OUTPUT WHICH
C                    INCLUDES, FOR EACH ITERATION, X, THE LINE
C                    SEARCH OBJECTIVE FUNCTION AND ITS COMPONENTS,
C                    AND THE STEP FACTOR USED IN CONJUNCTION WITH
C                    THE SEARCH DIRECTION
C
C       IPRINT >= 4  OUTPUT ABOVE PLUS, FOR EACH QUADRATIC SUB-
C                    PROBLEM, FGRD, CONF, CNORM, AND THE HESSIAN
C                    ESTIMATE
C
C     NWRITE IS AN INTEGER INPUT VARIABLE WHICH SPECIFIES THE UNIT
C       NUMBER OF THE DATASET OR FILE TO WHICH THE OUTPUT SELECTED BY
C       IPRINT IS TO BE WRITTEN.  IF NWRITE IS SET TO ANY NONPOSITIVE
C       VALUE THE DEFAULT UNIT (UNIT 6) WILL BE USED FOR THE PRINTED
C       OUTPUT.
C
C     INFO IS AN INTEGER OUTPUT PARAMETER SET AS FOLLOWS
C
C       IF INFO IS NEGATIVE THEN USER TERMINATION. OTHERWISE
C
C       INFO = 0  IMPROPER INPUT PARAMETERS. TESTS ARE MADE TO INSURE
C                 THAT N AND M ARE POSITIVE, TOL IS NON-NEGATIVE,
C                 MEQ IS LESS THAN OR EQUAL TO N, AND THAT LCNORM,
C                 LWA, AND LIWA ARE SUFFICIENTLY LARGE.
C
C       INFO = 1  A NORMAL RETURN. SEE DESCRIPTION OF TOL.
C
C       INFO = 2  NUMBER OF CALLS TO FCN IS AT LEAST 100*(N+1).
C
C       INFO = 3  LINE SEARCH REQUIRED TEN CALLS OF FCN.
C
C       INFO = 4  UPHILL SEARCH DIRECTION WAS CALCULATED.
C
C       INFO = 5  QUADRATIC PROGRAMMING ALGORITHM WAS UNABLE TO FIND
C                 A FEASIBLE POINT.
C
C       INFO = 6  QUADRATIC PROGRAMMING ALGORITHM WAS RESTRICTED BY
C                 AN ARTIFICIAL BOUND OR FAILED DUE TO A SINGULAR
C                 MATRIX.
C
C     WA IS A REAL*8 WORK ARRAY OF LENGTH LWA.
C
```

```
C    LWA IS A POSITIVE INTEGER INPUT VARIABLE SET EQUAL TO THE
C      DIMENSION OF WA WHICH IS AT LEAST
C      2*M + N*(5*N + 21) + 10 + MAX(7*(N+1),4*(N+1)+M).
C
C    IWA IS AN INTEGER WORK ARRAY OF LENGTH LIWA.
C
C    LIWA IS A POSITIVE INTEGER INPUT VARIABLE SET EQUAL TO THE
C      DIMENSION OF IWA WHICH IS AT LEAST 6*(N+1) + M.
C
C  SUBROUTINES CALLED
C
C    USER SUPPLIED ...... FCN
C
C    FORTRAN SUPPLIED ... MAXO
C
C    AMDLIB SUPPLIED ... VMCON
C
C  ALGORITHM VERSION OF JUNE 1979.
C
C  ROGER L. CRANE, KENNETH E. HILLSTROM, MICHAEL MINKOFF
C
C  **********
```

***** Use of Subroutine VMCON *****

```
      SUBROUTINE VMCON(FCN,MODE,N,M,MEQ,X,OBJF,FGRD,CONF,CNORM,LCNORM,
     1                 B,LB,TOL,MAXFEV,IPRINT,NWRITE,INFO,NFEV,VLAM,
     2                 GLAG,VMU,CM,GLAGA,GAMMA,ETA,XA,BDELTA,DELTA,
     3                 LDEL,GM,BDL,BDU,H,LH,WA,LWA,IWA,LIWA)
      INTEGER MODE,N,M,MEQ,LCNORM,LB,MAXFEV,IPRINT,NWRITE,INFO,NFEV,
     1        LDEL,LH,LWA,LIWA
      INTEGER IWA(LIWA)
      DOUBLE PRECISION OBJF,TOL
      DOUBLE PRECISION X(N),FGRD(N),CONF(M),CNORM(LCNORM,M),B(LB,LB),
     1                 VLAM(M),GLAG(N),VMU(M),CM(M),GLAGA(N),GAMMA(N),
     2                 ETA(N),XA(N),BDELTA(N),DELTA(LDEL),GM(1),
     3                 BDL(1),BDU(1),H(LH,LH),WA(LWA)
C  **********
C
C  SUBROUTINE VMCON
C
C  THIS SUBROUTINE CALCULATES THE LEAST VALUE OF A FUNCTION OF
C  SEVERAL VARIABLES SUBJECT TO LINEAR AND/OR NONLINEAR EQUALITY
C  AND INEQUALITY CONSTRAINTS.  MORE PARTICULARLY, IT SOLVES THE
C  PROBLEM
C
C            MINIMIZE F(X)
C
C     SUBJECT TO C (X) =  0.0 ,  I = 1,...,MEQ
C                 I
C
C            AND C (X) >= 0.0 ,  I = MEQ+1,...,M
C                 I
```

```
C
C  THE SUBROUTINE IMPLEMENTS A VARIABLE METRIC METHOD FOR
C  CONSTRAINED OPTIMIZATION DEVELOPED BY M.J.D. POWELL.
C
C  THE SUBROUTINE STATEMENT IS
C
C    SUBROUTINE VMCON(FCN,MODE,N,M,MEQ,X,OBJF,FGRD,CONF,CNORM,
C                     LCNORM,B,LB,TOL,MAXFEV,IPRINT,NWRITE,INFO,
C                     NFEV,VLAM,GLAG,VMU,CM,GLAGA,GAMMA,ETA,XA,
C                     BDELTA,DELTA,LDEL,GM,BDL,BDU,H,LH,WA,LWA.IWA,
C                     LIWA)
C
C  WHERE
C
C    FCN IS THE NAME OF THE USER SUPPLIED SUBROUTINE WHICH
C      CALCULATES THE OBJECTIVE AND CONSTRAINT FUNCTIONS, AND THE
C      GRADIENTS (FIRST DERIVATIVE VECTORS) OF THE OBJECTIVE AND
C      CONSTRAINT FUNCTIONS. FCN SHOULD BE DECLARED IN AN EXTERNAL
C      STATEMENT IN THE USER CALLING PROGRAM, AND SHOULD BE WRITTEN
C      AS FOLLOWS
C
C      SUBROUTINE FCN(N,M,X,OBJF,FGRD,CONF,CNORM,LCNORM,INFO)
C      INTEGER N,M,LCNORM,INFO
C      DOUBLE PRECISION OBJF
C      DOUBLE PRECISION X(N),FGRD(N),CONF(M),CNORM(LCNORM,M)
C      --------------
C      STATEMENTS TO CALCULATE THE OBJECTIVE AND CONSTRAINT
C      FUNCTIONS AND THE GRADIENTS OF THE OBJECTIVE AND CONSTRAINT
C      FUNCTIONS AT X. THE OBJECTIVE AND CONSTRAINT FUNCTIONS AND
C      THE GRADIENT OF THE OBJECTIVE FUNCTION MUST BE RETURNED IN
C      OBJF, CONF AND FGRD RESPECTIVELY. NOTE THAT THE EQUALITY
C      CONSTRAINTS MUST PRECEDE THE INEQUALITY CONSTRAINTS IN CONF.
C      THE CONSTRAINT GRADIENTS OR NORMALS MUST BE RETURNED AS THE
C      COLUMNS OF CNORM.
C      --------------
C      RETURN
C      END
C
C      THE VALUE OF INFO SHOULD NOT BE CHANGED BY FCN UNLESS THE
C      USER WANTS TO TERMINATE EXECUTION OF VMCON. IN THIS CASE
C      SET INFO TO A NEGATIVE INTEGER.
C
C    MODE IS A NON-NEGATIVE INTEGER INPUT VARIABLE SET TO 1 IF THE
C      SECOND DERIVATIVE MATRIX IN B (SEE BELOW) IS PROVIDED BY THE
C      USER, AND TO 0 IF IT IS TO BE INITIALIZED TO THE IDENTITY
C      MATRIX.
C
C    N IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE NUMBER OF
C      VARIABLES.
C
C    M IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE NUMBER OF
C      CONSTRAINTS.
C
```

```
C     MEQ IS A NON-NEGATIVE INTEGER INPUT VARIABLE SET TO THE NUMBER
C       OF EQUALITY CONSTRAINTS. MEQ MUST BE LESS THAN OR EQUAL TO N.
C
C     X IS A REAL*8 ARRAY OF LENGTH N. ON INPUT IT MUST CONTAIN AN
C       INITIAL ESTIMATE OF THE SOLUTION VECTOR. ON OUTPUT X
C       CONTAINS THE FINAL ESTIMATE OF THE SOLUTION VECTOR.
C
C     OBJF IS A REAL*8 OUTPUT VARIABLE THAT CONTAINS THE VALUE OF THE
C       OBJECTIVE FUNCTION AT THE OUTPUT X.
C
C     FGRD IS A REAL*8 OUTPUT ARRAY OF LENGTH N WHICH CONTAINS THE
C       COMPONENTS OF THE GRADIENT OF THE OBJECTIVE FUNCTION AT
C       THE OUTPUT X.
C
C     CONF IS A REAL*8 OUTPUT ARRAY OF LENGTH M WHICH CONTAINS THE
C       VALUES OF THE CONSTRAINT FUNCTIONS AT THE OUTPUT X. THE
C       EQUALITY CONSTRAINTS MUST PRECEDE THE INEQUALITY CONSTRAINTS.
C
C     CNORM IS A REAL*8 LCNORM BY M ARRAY WHOSE COLUMNS CONTAIN THE
C       CONSTRAINT NORMALS AT THE OUTPUT X IN THE FIRST N POSITIONS.
C
C     LCNORM IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE ROW
C       DIMENSION OF CNORM WHICH IS AT LEAST N+1.  THE (N+1)ST ROW
C       OF CNORM IS USED FOR WORK SPACE.
C
C     B IS A REAL*8 LB BY LB ARRAY WHOSE FIRST N ROWS AND COLUMNS
C       CONTAIN THE APPROXIMATION TO THE SECOND DERIVATIVE MATRIX
C       OF THE LAGRANGIAN FUNCTION. OFTEN, AN ADEQUATE INITIAL
C       B MATRIX CAN BE OBTAINED BY APPROXIMATING THE HESSIAN
C       OF THE OBJECTIVE FUNCTION.  ON INPUT, THE APPROXIMATION IS
C       PROVIDED BY THE USER IF MODE = 1 AND IS SET TO THE IDENTITY
C       MATRIX IF MODE = 0. THE (N+1)ST ROW AND COLUMN ARE USED FOR
C       WORK SPACE.
C
C     LB IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE ROW
C       DIMENSION OF B WHICH IS AT LEAST N+1.
C
C     TOL IS A NON-NEGATIVE INPUT VARIABLE. A NORMAL RETURN OCCURS
C       WHEN THE OBJECTIVE FUNCTION PLUS SUITABLY WEIGHTED MULTIPLES
C       OF THE CONSTRAINT FUNCTIONS ARE PREDICTED TO DIFFER FROM
C       THEIR OPTIMAL VALUES BY AT MOST TOL.
C
C     MAXFEV IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE LIMIT
C       ON THE NUMBER OF CALLS TO FCN.
C
C     IPRINT IS AN INTEGER INPUT PARAMETER WHICH CONTROLS THE PRINTED
C       OUTPUT FROM VMCON.  IT SHOULD BE SET AS FOLLOWS
C
C       IPRINT <= 0  NO OUTPUT
C
C       IPRINT  = 1  FOR EACH QUADRATIC SUBPROBLEM, X, OBJF, AND
C                    THE NORM OF THE LAGRANGIAN GRADIENT ARE OUTPUT
C
C       IPRINT  = 2  OUTPUT ABOVE PLUS THE SEARCH DIRECTION AND THE
```

```
C                    LAGRANGE MULTIPLIERS FROM THE QUADRATIC SUB-
C                    PROBLEM, AND THE MULTIPLIERS FOR THE LINE
C                    SEARCH
C
C       IPRINT  = 3  OUTPUT ABOVE PLUS LINE SEARCH OUTPUT WHICH
C                    INCLUDES, FOR EACH ITERATION, X, THE LINE
C                    SEARCH OBJECTIVE FUNCTION AND ITS COMPONENTS,
C                    AND THE STEP FACTOR USED IN CONJUNCTION WITH
C                    THE SEARCH DIRECTION
C
C       IPRINT >= 4  OUTPUT ABOVE PLUS, FOR EACH QUADRATIC SUB-
C                    PROBLEM, FGRD, CONF, CNORM, AND THE HESSIAN
C                    ESTIMATE
C
C     NWRITE IS AN INTEGER INPUT VARIABLE WHICH SPECIFIES THE UNIT
C       NUMBER OF THE DATASET OR FILE TO WHICH THE OUTPUT SELECTED BY
C       IPRINT IS TO BE WRITTEN.  IF NWRITE IS SET TO ANY NONPOSITIVE
C       VALUE THE DEFAULT UNIT (UNIT 6) WILL BE USED FOR THE PRINTED
C       OUTPUT.
C
C     INFO IS AN INTEGER OUTPUT VARIABLE SET AS FOLLOWS
C
C       IF INFO IS NEGATIVE THEN USER TERMINATION. OTHERWISE
C
C       INFO = 0  IMPROPER INPUT PARAMETERS. TESTS ARE MADE TO INSURE
C                 THAT N AND M ARE POSITIVE, TOL IS NON-NEGATIVE,
C                 MEQ IS LESS THAN OR EQUAL TO N, AND THAT LCNORM,
C                 LB, LDEL, LH, LWA, AND LIWA ARE SUFFICIENTLY LARGE.
C
C       INFO = 1  A NORMAL RETURN. SEE DESCRIPTION OF TOL.
C
C       INFO = 2  NUMBER OF CALLS TO FCN IS AT LEAST MAXFEV.
C
C       INFO = 3  LINE SEARCH REQUIRED TEN CALLS OF FCN.
C
C       INFO = 4  UPHILL SEARCH DIRECTION WAS CALCULATED.
C
C       INFO = 5  QUADRATIC PROGRAMMING TECHNIQUE WAS UNABLE TO FIND
C                 A FEASIBLE POINT.
C
C       INFO = 6  QUADRATIC PROGRAMMING TECHNIQUE WAS RESTRICTED BY
C                 AN ARTIFICIAL BOUND OR FAILED DUE TO A SINGULAR
C                 MATRIX.
C
C     NFEV IS AN INTEGER OUTPUT VARIABLE SET TO THE NUMBER OF CALLS
C       TO FCN.
C
C     VLAM IS A REAL*8 OUTPUT ARRAY OF LENGTH M WHICH CONTAINS THE
C       LAGRANGE MULTIPLIERS AT THE OUTPUT X.  THE LAGRANGE
C       MULTIPLIERS PROVIDE THE SENSITIVITY OF THE OBJECTIVE
C       FUNCTION TO CHANGES IN THE CONSTRAINT FUNCTIONS.
C
C     GLAG IS A REAL*8 OUTPUT ARRAY OF LENGTH N WHICH CONTAINS THE
C       COMPONENTS OF THE GRADIENT OF THE LAGRANGIAN FUNCTION AT
```

```
C        THE OUTPUT X.
C
C    VMU, CM ARE REAL*8 WORK ARRAYS OF LENGTH M.
C
C    GLAGA, GAMMA, ETA, XA, BDELTA ARE REAL*8 WORK ARRAYS OF
C      LENGTH N.
C
C    DELTA IS A REAL*8 WORK ARRAY OF LENGTH LDEL.
C
C    LDEL IS A POSITIVE INTEGER INPUT VARIABLE SET EQUAL TO THE
C      LENGTH OF DELTA WHICH IS AT LEAST MAX(7*(N+1),4*(N+1)+M).
C
C    GM, BDL, BDU ARE REAL*8 WORK ARRAYS OF LENGTH N+1.
C
C    H IS A REAL*8 LH BY LH WORK ARRAY.
C
C    LH IS A POSITIVE INTEGER INPUT VARIABLE SET TO THE DIMENSION
C      OF THE SQUARE ARRAY H WHICH IS AT LEAST 2*(N+1).
C
C    WA IS A REAL*8 WORK ARRAY OF LENGTH LWA.
C
C    LWA IS A POSITIVE INTEGER INPUT VARIABLE SET EQUAL TO THE
C      DIMENSION OF WA WHICH IS AT LEAST 2*(N+1).
C
C    IWA IS AN INTEGER WORK ARRAY OF LENGTH LIWA.
C
C    LIWA IS A POSITIVE INTEGER INPUT VARIABLE SET EQUAL TO THE
C      DIMENSION OF IWA WHICH IS AT LEAST 6*(N+1) + M.
C
C  SUBPROGRAMS REQUIRED
C
C    USER SUPPLIED ...... FCN
C
C    FORTRAN SUPPLIED ... DABS,DMAX1
C
C    MINPACK SUPPLIED ... ENORM
C
C    AMDLIB SUPPLIED ... QPSUB
C
C  ALGORITHM VERSION OF JUNE 1979.
C
C  ROGER L. CRANE, KENNETH E. HILLSTROM, MICHAEL MINKOFF
C
C  **********
```

Examples

The introductory chapter of Reference [9] contains five simple examples of mathematical programming problems. Each problem has only two variables, and graphs are included to identify the constraints and the solutions. Included below is the result obtained by using VMCON1 to solve one of these problems, the general nonlinear problem

$$\text{minimize } f(x_1,x_2) = (x_1 - 2)^2 + (x_2 - 1)^2 \quad (12)$$
$$\text{subject to}$$
$$c_1(x_1,x_2) = x_1 - 2*x_2 + 1 = 0$$
$$c_2(x_1,x_2) = - x_1^2 /4 - x_2^2 + 1 \geq 0.$$

The results for two closely related problems are also given. The main program listed below has been written so that any of the five examples in Chapter 1 of Ref. [9] can be run simply by changing SUBROUTINE FCN and by furnishing the appropriate input data. Each of these five problems is instructive and simple to code and thus can be used to gain more experience in the use of VMCON or VMCON1.

The main program also includes code to check how well the Kuhn-Tucker necessary conditions (3) are satisfied by the solution produced by VMCON1. The Lagrangian Gradient Error (see 3a) is computed as the sum of the magnitudes of the components of $\nabla_x L(x^*,\lambda^*)$, and the Lagrange Multiplier Error (see 3b) is determined as the sum of the magnitudes of the negative Lagrange multipliers associated with inequality constraints. The Complementarity Error (see 3c) is defined as

$$\sum_{i=1}^{m} |\lambda_i^* c_i(x^*)|,$$

and the Constraint Error (see 3d,3e) is defined as

$$\sum_{i=1}^{k} |c_i(x^*)| + \sum_{i=k+1}^{m} |\min(0,c_i(x^*)|.$$

```
C
C     MAIN PROGRAM FOR USE WITH THE FIVE EXAMPLES GIVEN IN THE
C     INTRODUCTION OF BRACKEN AND MCCORMICK (9).  AS DESCRIBED IN THE
C     DOCUMENTATION FOR VMCON AND VMCON1, THE USER MUST SUPPLY A
C     SUBROUTINE TO COMPUTE THE OBJECTIVE AND CONSTRAINT FUNCTIONS AND
C     THEIR DERIVATIVES.
C
C
C     DECLARATION OF VARIABLES USED IN VMCON1 ARGUMENT LIST
C
```

```
      INTEGER N,M,MEQ,LCNORM,IPRINT,NWRITE,INFO,LWA,IWA(22),LIWA
      REAL*8 X(2),OBJF,FGRD(2),CONF(4),CNORM(3,4),VLAM(4),TOL,WA(101)
C
C     DECLARATION OF COMMON AND LOCAL VARIABLES
C
      INTEGER I,J,NFUN
      REAL*8 SUM,DABS,ZERO,ERRLG,ERRLM,ERRCOM,ERRCON
C
      COMMON/STAT/ NFUN
C
      EXTERNAL FCN
C
      DATA LCNORM/3/,LIWA/22/,LWA/101/,NWRITE/6/
      DATA ZERO/0.D0/
C
C     READ  N      - NUMBER OF VARIABLES
C           M      - NUMBER OF CONSTRAINTS
C           MEQ    - NUMBER OF EQUALITY CONSTRAINTS
C           IPRINT - INTERMEDIATE OUTPUT OPTION
C           TOL    - TERMINATION ACCURACY
C
10    READ(5,*,END=300) N,M,MEQ,IPRINT,TOL
      WRITE(6,20) N,M,MEQ,IPRINT,TOL
20    FORMAT('1N =',I3,'   M = ',I3,'   MEQ =',I3,'   IPRINT =',I3,
     1       '   TOL =',1PD10.2)
C
C     READ INITIAL SOLUTION ESTIMATE
C
      READ(5,*) (X(I),I=1,N)
      WRITE(6,30) (X(I),I=1,N)
30    FORMAT('0INITIAL SOLUTION ESTIMATE, X',/,(1P5D24.16))
C
      NFUN = 0
      CALL VMCON1 (FCN,N,M,MEQ,X,OBJF,FGRD,CONF,CNORM,LCNORM,
     1             VLAM,TOL,IPRINT,NWRITE,INFO,WA,LWA,IWA,LIWA)
C
C     OUTPUT SOLUTION
C
      WRITE(6,40)INFO,NFUN
40    FORMAT('0INFO =',I3,I9,' FUNCTION EVALUATIONS')
      WRITE(6,50) (X(I),I=1,N)
50    FORMAT('0FINAL SOLUTION ESTIMATE, X',/,(1P5D24.16))
      WRITE(6,60) OBJF
60    FORMAT('0F(X) =',1PD24.16)
      WRITE(6,70) (CONF(I),I=1,M)
70    FORMAT('0CONSTRAINTS EVALUATED AT X',/,(1P5D24.16))
      WRITE(6,80) (VLAM(I),I=1,M)
80    FORMAT('0LAGRANGE MULTIPLIER ESTIMATES',/,(1P5D24.16))
C
C     EVALUATE KUHN-TUCKER NECESSARY CONDITIONS (3)
C
C     CALCULATE 1-NORM OF LAGRANGIAN GRADIENT ERRORS (3A)
C
      ERRLG = ZERO
```

```
      DO 110 I = 1,N
         SUM = FGRD(I)
         DO 100 J = 1,M
            SUM = SUM - VLAM(J)*CNORM(I,J)
100      CONTINUE
         ERRLG = ERRLG + DABS(SUM)
110   CONTINUE
C
C     CALCULATE 1-NORM OF NEGATIVE LAGRANGE MULTIPLIER ERRORS (3B)
C
      ERRLM = ZERO
      DO 120 I = 1,M
         IF(I .LE. MEQ .OR. VLAM(I) .GE. ZERO) GO TO 120
         ERRLM = ERRLM + DABS(VLAM(I))
120   CONTINUE
C
C     CALCULATE 1-NORM OF COMPLEMENTARITY ERRORS (3C)
C
      ERRCOM = ZERO
      DO 130 I = 1,M
         ERRCOM = ERRCOM + DABS(VLAM(I)*CONF(I))
130   CONTINUE
C
C     CALCULATE 1-NORM OF CONSTRAINT ERRORS (3D,3E)
C
      ERRCON = ZERO
      DO 140 I = 1,M
         IF(I .GT. MEQ .AND. CONF(I) .GE. ZERO) GO TO 140
         ERRCON = ERRCON + DABS(CONF(I))
140   CONTINUE
C
C     OUTPUT KUHN-TUCKER ERRORS
C
      WRITE(6,200) ERRLG
200   FORMAT('OLAGRANGIAN GRADIENT ERROR =',1PD24.16)
      WRITE(6,210) ERRLM
210   FORMAT(' LAGRANGE MULTIPLIER ERROR =',1PD24.16)
      WRITE(6,220) ERRCOM
220   FORMAT(' COMPLEMENTARITY ERROR =',1PD24.16)
      WRITE(6,230) ERRCON
230   FORMAT(' CONSTRAINT ERROR =',1PD24.16)
C
      GO TO 10
C
300   STOP
      END
C
C
      SUBROUTINE FCN (N,M,X,OBJF,FGRD,CONF,CNORM,LCNORM,INFO)
C
C     NONLINEAR PROBLEM WITH ONE INEQUALITY AND ONE EQUALITY CONSTR.
C
C          MINIMIZE F(X1,X2) = (X1 - 2)**2 + (X2 - 1)**2
C              SUBJECT TO
```

```
C                    C1(X1,X2) = X1 - 2*X2 + 1 = 0
C                    C2(X1,X2) = -X1**2/4 - X2**2 + 1 >= 0
C
C     REFERENCE.  BRACKEN AND MCCORMICK (9), PP. 18-19.
C
      INTEGER N,M,LCNORM,INFO
      REAL*8 X(N),OBJF,FGRD(N),CONF(M),CNORM(LCNORM,M)
C
      INTEGER NFUN
C
      COMMON/STAT/ NFUN
C
      OBJF = (X(1) - 2.D0)**2 + (X(2) - 1.D0)**2
C
      FGRD(1) = 2.D0*(X(1) - 2.D0)
      FGRD(2) = 2.D0*(X(2) - 1.D0)
C
      CONF(1) = X(1) - 2.D0*X(2) + 1.D0
      CONF(2) = -0.25D0*X(1)**2 - X(2)**2 + 1.D0
C
      CNORM(1,1) = 1.D0
      CNORM(2,1) = -2.D0
      CNORM(1,2) = -0.5D0*X(1)
      CNORM(2,2) = -2.D0*X(2)
C
      NFUN = NFUN + 1
      RETURN
      END
```

All of the following results were produced by use of an IBM Model 370/168 computer. The first two runs were made with the MAIN PROGRAM and SUBROUTINE FCN listed above to solve Problem (12). No intermediate output from VMCON was generated in the first run because IPRINT = 0. IPRINT was set to 3 in the second run to permit one to follow the iterative steps taken by VMCON in solving (12). In each case, a starting estimate of (2,2) was used.

```
N =  2   M =   2   MEQ =  1   IPRINT =  0   TOL =  1.00D-08
INITIAL SOLUTION ESTIMATE, X
 2.0000000000000000D+00  2.0000000000000000D+00
INFO =  1         6 FUNCTION EVALUATIONS
FINAL SOLUTION ESTIMATE, X
 8.2287565553287513D-01  9.1143782776643764D-01
F(X) =  1.3934649806878849D+00
CONSTRAINTS EVALUATED AT X
 1.3877787807814457D-17 -7.6716411001598317D-13
LAGRANGE MULTIPLIER ESTIMATES
-1.5944911182523063D+00  1.8465914396061125D+00
LAGRANGIAN GRADIENT ERROR =  3.3450880954077888D-12
LAGRANGE MULTIPLIER ERROR =  0.0
COMPLEMENTARITY ERROR =  1.4166608063379568D-12
CONSTRAINT ERROR =  7.6717798780379098D-13
```

```
N =  2   M =   2   MEQ =  1   IPRINT =  3   TOL =  1.00D-08
INITIAL SOLUTION ESTIMATE, X
 2.0000000000000000D+00  2.0000000000000000D+00
X FOR QUADRATIC SUBPROBLEM   1
 2.0000000000000000D+00  2.0000000000000000D+00
OBJF =  1.0000000000000000D+00
NORM OF LAGRANGIAN GRADIENT =  1.0671873729184791D+00
SEARCH DIRECTION
-6.6666666667013622D-01 -8.3333333334721112D-01
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
-6.3888888888888891D-01
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 2.7777777781247306D-02
WEIGHTS FOR LINE SEARCH
 6.3888888888888891D-01  2.7777777781247306D-02
X FOR LINE SEARCH ITERATION  1
 2.0000000000000000D+00  2.0000000000000000D+00
LINE SEARCH F(X) =  1.7500000000138780D+00
(OBJF =    1.000000D+00, CONSTR =   7.500000D-01)
LINE SEARCH STEPSIZE ALPHA =  1.0000000000000000D+00
X FOR LINE SEARCH ITERATION  2
 1.3333333333298636D+00  1.1666666666527887D+00
LINE SEARCH F(X) =  4.9459876544944636D-01
(OBJF =    4.722222D-01, CONSTR =   2.237654D-02)
X FOR QUADRATIC SUBPROBLEM   2
 1.3333333333298636D+00  1.1666666666527887D+00
OBJF =  4.7222222222222256D-01
NORM OF LAGRANGIAN GRADIENT =  9.2657637191068534D-01
SEARCH DIRECTION
-4.3939393939421147D-01 -2.1969696968496266D-01
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
-1.2606501345340593D+00
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 1.1880088183710331D+00
WEIGHTS FOR LINE SEARCH
 1.2606501345340593D+00  1.1880088183710331D+00
X FOR LINE SEARCH ITERATION  1
 1.3333333333298636D+00  1.1666666666527887D+00
LINE SEARCH F(X) =  1.4292293258993971D+00
(OBJF =    4.722222D-01, CONSTR =   9.570071D-01)
LINE SEARCH STEPSIZE ALPHA =  1.0000000000000000D+00
X FOR LINE SEARCH ITERATION  2
 8.9393939393565212D-01  9.4696969696782607D-01
LINE SEARCH F(X) =  1.3408649467711080D+00
(OBJF =    1.226182D+00, CONSTR =   1.146827D-01)
X FOR QUADRATIC SUBPROBLEM   3
 8.9393939393565212D-01  9.4696969696782607D-01
OBJF =  1.2261822773271165D+00
NORM OF LAGRANGIAN GRADIENT =  2.3501177267611324D-01
SEARCH DIRECTION
-6.9252305661793120D-02 -3.4626152830896526D-02
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
-1.5835406770665301D+00
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
```

```
 1.8082239730793084D+00
WEIGHTS FOR LINE SEARCH
 1.5835406770665301D+00  1.8082239730793084D+00
X FOR LINE SEARCH ITERATION  1
 8.9393939393565212D-01  9.4696969696782607D-01
LINE SEARCH F(X) =  1.4007364969411580D+00
(OBJF =    1.226182D+00, CONSTR =    1.745542D-01)
LINE SEARCH STEPSIZE ALPHA =  1.0000000000000000D+00
X FOR LINE SEARCH ITERATION  2
 8.2468708827385900D-01  9.1234354413692953D-01
LINE SEARCH F(X) =  1.3933801089817501D+00
(OBJF =    1.389044D+00, CONSTR =    4.336014D-03)
X FOR QUADRATIC SUBPROBLEM   4
 8.2468708827385900D-01  9.1234354413692953D-01
OBJF =  1.3890440947246536D+00
NORM OF LAGRANGIAN GRADIENT =  7.3128235955400283D-03
SEARCH DIRECTION
-1.8101942269716479D-03 -9.0509711348578934D-04
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
-1.5944760660530162D+00
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 1.8465349463527272D+00
WEIGHTS FOR LINE SEARCH
 1.5944760660530162D+00  1.8465349463527272D+00
X FOR LINE SEARCH ITERATION  1
 8.2468708827385900D-01  9.1234354413692953D-01
LINE SEARCH F(X) =  1.3934719764322374D+00
(OBJF =    1.389044D+00, CONSTR =    4.427882D-03)
LINE SEARCH STEPSIZE ALPHA =  1.0000000000000000D+00
X FOR LINE SEARCH ITERATION  2
 8.2287689404688735D-01  9.1143844702344373D-01
LINE SEARCH F(X) =  1.3934649806000765D+00
(OBJF =    1.393462D+00, CONSTR =    3.025366D-06)
X FOR QUADRATIC SUBPROBLEM   5
 8.2287689404688735D-01  9.1143844702344373D-01
OBJF =  1.3934619552343219D+00
NORM OF LAGRANGIAN GRADIENT =  5.0534185475060226D-06
SEARCH DIRECTION
-1.2385140122115455D-06 -6.1925700608743976D-07
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
-1.5944911182381725D+00
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 1.8465914395513170D+00
WEIGHTS FOR LINE SEARCH
 1.5944911182381725D+00  1.8465914395513170D+00
X FOR LINE SEARCH ITERATION  1
 8.2287689404688735D-01  9.1143844702344373D-01
LINE SEARCH F(X) =  1.3934649806926351D+00
(OBJF =    1.393462D+00, CONSTR =    3.025458D-06)
LINE SEARCH STEPSIZE ALPHA =  1.0000000000000000D+00
X FOR LINE SEARCH ITERATION  2
 8.2287565553287513D-01  9.1143782776643764D-01
LINE SEARCH F(X) =  1.3934649806893016D+00
(OBJF =    1.393465D+00, CONSTR =    1.416661D-12)
```

```
X FOR QUADRATIC SUBPROBLEM   6
 8.2287565553287513D-01  9.1143782776643764D-01
OBJF =  1.3934649806878849D+00
NORM OF LAGRANGIAN GRADIENT =  2.3655407013876912D-12
SEARCH DIRECTION
-5.7993111865659503D-13 -2.8992456599272847D-13
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
-1.5944911182523063D+00
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 1.8465914396061125D+00
WEIGHTS FOR LINE SEARCH
 1.5944911182523063D+00  1.8465914396061125D+00
INFO =  1        6 FUNCTION EVALUATIONS
FINAL SOLUTION ESTIMATE, X
 8.2287565553287513D-01  9.1143782776643764D-01
F(X) =  1.3934649806878849D+00
CONSTRAINTS EVALUATED AT X
 1.3877787807814457D-17 -7.6716411001598317D-13
LAGRANGE MULTIPLIER ESTIMATES
-1.5944911182523063D+00  1.8465914396061125D+00
LAGRANGIAN GRADIENT ERROR =  3.3450880954077888D-12
LAGRANGE MULTIPLIER ERROR =  0.0
COMPLEMENTARITY ERROR =  1.4166608063379568D-12
CONSTRAINT ERROR =  7.6717798780379098D-13
```

The following output was obtained by defining the first constraint of (12) to be an inequality constraint, i.e., by setting

$$c_1(x_1,x_2) = x_1 - 2*x_2 + 1 \geq 0.$$

As can be seen by inspecting the output, the first constraint is strictly satisfied at the solution, i.e., $c_1(x_1^*, x_2^*) > 0$.

```
N =  2   M =  2   MEQ =  0   IPRINT =  2   TOL =  1.00D-08
INITIAL SOLUTION ESTIMATE, X
 2.0000000000000000D+00  2.0000000000000000D+00
X FOR QUADRATIC SUBPROBLEM   1
 2.0000000000000000D+00  2.0000000000000000D+00
OBJF =  1.0000000000000000D+00
NORM OF LAGRANGIAN GRADIENT =  2.0000000000000000D+00
SEARCH DIRECTION
 0.0                    -2.0000000000000000D+00
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                     0.0
WEIGHTS FOR LINE SEARCH
 0.0                     0.0
X FOR QUADRATIC SUBPROBLEM   2
 2.0000000000000000D+00  1.0000000000000000D+00
OBJF =  0.0
NORM OF LAGRANGIAN GRADIENT =  7.4535599249992988D-01
SEARCH DIRECTION
-3.3333333333333333D-01 -3.3333333333333333D-01
```

```
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                        3.3333333333333331D-01
WEIGHTS FOR LINE SEARCH
 0.0                        3.3333333333333331D-01
X FOR QUADRATIC SUBPROBLEM   3
 1.6666666666666665D+00  6.6666666666666667D-01
OBJF =  2.2222222222222231D-01
NORM OF LAGRANGIAN GRADIENT =  3.0357511185755286D-01
SEARCH DIRECTION
 9.1542781631979098D-02 -1.6138090518665336D-01
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                        7.2246448629005593D-01
WEIGHTS FOR LINE SEARCH
 0.0                        7.2246448629005593D-01
X FOR QUADRATIC SUBPROBLEM   4
 1.7116274847636603D+00  5.8740517391898890D-01
OBJF =  2.5339319805255274D-01
NORM OF LAGRANGIAN GRADIENT =  1.2811646350463538D-01
SEARCH DIRECTION
-4.8777874421563779D-02 -3.0402545234618000D-02
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                        7.7962167158415112D-01
WEIGHTS FOR LINE SEARCH
 0.0                        7.7962167158415112D-01
X FOR QUADRATIC SUBPROBLEM   5
 1.6628496103420964D+00  5.5700262868437089D-01
OBJF =  3.0991705623903357D-01
NORM OF LAGRANGIAN GRADIENT =  1.1735150097131356D-02
SEARCH DIRECTION
 2.2648544569566127D-03 -3.0540171764236416D-03
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                        8.0476906403356428D-01
WEIGHTS FOR LINE SEARCH
 0.0                        8.0476906403356428D-01
X FOR QUADRATIC SUBPROBLEM   6
 1.6651144647990530D+00  5.5394861150794725D-01
OBJF =  3.1111016286251286D-01
NORM OF LAGRANGIAN GRADIENT =  5.0355469328224549D-04
SEARCH DIRECTION
-1.4601565292086483D-04  1.0015096458053793D-04
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                        8.0489463690323858D-01
WEIGHTS FOR LINE SEARCH
 0.0                        8.0489463690323858D-01
X FOR QUADRATIC SUBPROBLEM   7
 1.6649684491461321D+00  5.5404876247252778D-01
OBJF =  3.1111864631983183D-01
NORM OF LAGRANGIAN GRADIENT =  3.9411810755140243D-07
SEARCH DIRECTION
 9.8090412324918410D-08 -8.7554639248454225D-08
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                        8.0489557166403237D-01
WEIGHTS FOR LINE SEARCH
 0.0                        8.0489557166403237D-01
```

```
X FOR QUADRATIC SUBPROBLEM   8
 1.6649685472365443D+00  5.5404867491788852D-01
OBJF =  3.1111865868328270D-01
NORM OF LAGRANGIAN GRADIENT =  1.6599489103916337D-11
SEARCH DIRECTION
-4.5881350475022549D-12  3.4378166279704850D-12
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0                     8.0489557193146243D-01
WEIGHTS FOR LINE SEARCH
 0.0                     8.0489557193146243D-01
INFO =  1       10 FUNCTION EVALUATIONS
FINAL SOLUTION ESTIMATE, X
 1.6649685472365443D+00  5.5404867491788852D-01
F(X) =  3.1111865868328270D-01
CONSTRAINTS EVALUATED AT X
 1.5568711974007674D+00 -1.0214051826551440D-14
LAGRANGE MULTIPLIER ESTIMATES
 0.0                     8.0489557193146243D-01
LAGRANGIAN GRADIENT ERROR =  2.3433338602885101D-11
LAGRANGE MULTIPLIER ERROR =  0.0
COMPLEMENTARITY ERROR =  8.2212450866697197D-15
CONSTRAINT ERROR =  1.0214051826551440D-14
```

To demonstrate the behavior of the program when an attempt is made to solve a problem for which no feasible solution exists, the first constraint of (12) was redefined as

$$c_1(x_1,x_2) = x_1 + x_2 - 3 = 0,$$

and the following output was produced.

```
N =  2   M =   2   MEQ =  1   IPRINT =  2   TOL =  1.00D-08
INITIAL SOLUTION ESTIMATE, X
 2.0000000000000000D+00  2.0000000000000000D+00
X FOR QUADRATIC SUBPROBLEM   1
 2.0000000000000000D+00  2.0000000000000000D+00
OBJF =  1.0000000000000000D+00
NORM OF LAGRANGIAN GRADIENT =  1.5811388300841893D+00
SEARCH DIRECTION
 5.0000000000000000D-01 -1.5000000000000000D+00
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
 5.0000000000000000D-01
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 0.0
WEIGHTS FOR LINE SEARCH
 5.0000000000000000D-01  0.0
X FOR QUADRATIC SUBPROBLEM   2
 2.5000000000000000D+00  5.0000000000000000D-01
OBJF =  5.0000000000000000D-01
NORM OF LAGRANGIAN GRADIENT =  8.4749631267634491D+00
SEARCH DIRECTION
-3.2500000000000020D+00  3.2500000000000016D+00
```

```
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
 4.4950000000000042D+01
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 3.8800000000000040D+01
WEIGHTS FOR LINE SEARCH
 4.4950000000000042D+01  3.8800000000000040D+01
X FOR QUADRATIC SUBPROBLEM   3
 2.3841584158415843D+00  6.1584158415841569D-01
OBJF =  2.9515537692383119D-01
NORM OF LAGRANGIAN GRADIENT =  1.6656672144959527D+03
SEARCH DIRECTION
 2.0207920792062648D+01 -2.0207920792062666D+01
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
 6.1913570494947743D+04
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 5.1574049999915223D+04
WEIGHTS FOR LINE SEARCH
 6.1913570494947743D+04  5.1574049999915223D+04
X FOR QUADRATIC SUBPROBLEM   4
 2.4043663366336467D+00  5.9563366336635303D-01
OBJF =  3.2702426840503161D-01
NORM OF LAGRANGIAN GRADIENT =  1.0127428365451781D+04
SEARCH DIRECTION
-9.5249690073447399D-02  9.5249690073443583D-02
LAGRANGE MULTIPLIERS FOR EQUALITY CONSTRAINTS
 1.3499423760088445D+06
LAGRANGE MULTIPLIERS FOR INEQUALITY CONSTRAINTS
 1.1249516708907119D+06
WEIGHTS FOR LINE SEARCH
 1.3499423760088445D+06  1.1249516708907119D+06
X FOR QUADRATIC SUBPROBLEM   5
 2.3999994310874733D+00  6.0000056891252611D-01
OBJF =  3.1999908974060504D-01
INFO =  5        12 FUNCTION EVALUATIONS
FINAL SOLUTION ESTIMATE, X
 2.3999994310874733D+00  6.0000056891252611D-01
F(X) =  3.1999908974060504D-01
CONSTRAINTS EVALUATED AT X
-6.6613381477509392D-16 -8.0000000000040350D-01
LAGRANGE MULTIPLIER ESTIMATES
 0.0                     0.0
LAGRANGIAN GRADIENT ERROR =  1.5999977243498942D+00
LAGRANGE MULTIPLIER ERROR =  0.0
COMPLEMENTARITY ERROR =  0.0
CONSTRAINT ERROR =  8.0000000000040417D-01
```

Acknowledgments

We wish to thank G. Pieper for her substantial help in editing several versions of this report and J. Beumer for her assistance in preparing the final copy.

References

1. Powell, M. J. D., "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations," Proceedings of the 1977 Dundee Conference on Numerical Analysis, Lecture Notes in Mathematics, Vol. 630, Springer-Verlag, Berlin, 1978, pp. 144-157.

2. Powell, M. J. D., "Algorithms for Nonlinear Constraints That Use Lagrangian Functions," Mathematical Programming, 14, 224-248 (1978).

3. Han, S-P, "A Globally Convergent Method for Nonlinear Programming," Report No. 75-257, Department of Computer Science, Cornell University, 1975.

4. Lasdon, L. S., and A. D. Waren, "The Status of Nonlinear Programming Software," Operations Research, 27, 431-456 (1979).

5. Schittkowski, K., "A Numerical Comparison of 13 Nonlinear Programming Codes with Randomly Generated Test Problems," Numerical Optimization of Dynamical Systems, L. C. W. Dixon and G. P. Szego, eds., North-Holland Publishing Co. (to appear)

6. Avriel, M., Nonlinear Programming: Analysis and Methods, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

7. Fiacco, A. V., and G. P. McCormick, Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley & Sons, Inc., New York, 1968.

8. Lasdon, L. S., and A. D. Waren, "A Survey of Nonlinear Programming Applications," Computer and Information Sciences Department, Working Paper 80-01, Cleveland State University, March 1980 (to appear in Operations Research).

9. Bracken, J., and G. P. McCormick, Selected Applications of Nonlinear Programming, John Wiley & Sons, Inc., New York, 1968.

10. Fletcher, R., "A Fortran Program for General Quadratic Programming," Report No. R6370, Atomic Energy Research Establishment, Harwell, Berkshire, U.K., 1970.

11. Fletcher, R., "The Calculation of Feasible Points for Linearly Constrained Optimization Problems," Report No. R6354, Atomic Energy Research Establishment, Harwell, Berkshire, U.K., 1970.

12. Dongarra, J. J., C. B. Moler, J. R. Bunch, and G. W. Stewart, LINPACK Users' Guide, Society for Industrial and Applied Mathematics, Philadelphia, Pa., 1979.

13. Lawson, C. L., R. J. Hanson, D. R. Kincaid, and F. T. Krogh, "Basic Linear Algebra Subprograms for Fortran Usage," ACM Transactions on Mathematical Software, 5, 308-323 (1979).

14. Lyness, J. N., and J. J. Kaganove, "Comments on the Nature of Automatic Quadrature Routines," ACM Transactions on Mathematical Software, 2, 65-81 (1976).