NASA Technical Memorandum 4552

An Optimization Program Based on the Method of Feasible Directions

Theory and Users Guide

Ashok D. Belegundu
The Pennsylvania State University
University Park, Pennsylvania

Laszlo Berke
Lewis Research Center
Cleveland, Ohio

and

Surya N. Patnaik
Ohio Aerospace Institute
Brook Park, Ohio

National Aeronautics and
Space Administration

Office of Management

Scientific and Technical
Information Program

1994

Summary

The theory and user instructions for an optimization code based on the method of feasible directions are presented. The code was written for wide distribution and ease of attachment to other simulation software. Although the theory of the method of feasible direction was developed in the 1960's, many considerations are involved in its actual implementation as a computer code. Included in the code are a number of features to improve robustness in optimization. The search direction is obtained by solving a quadratic program using an interior method based on Karmarkar's algorithm. The theory is discussed, focusing on the important and often overlooked role played by the various parameters guiding the iterations within the program. Also discussed is a robust approach for handling infeasible starting points. The code was validated by solving a variety of structural optimization test problems that have known solutions obtained by other optimization codes. It has been observed that this code is accurate and robust: it has solved a variety of problems from different starting points. However, the code is inefficient in that it takes considerable CPU time as compared with certain other available codes. Further work is required to improve its efficiency while retaining its robustness.

Introduction

The theoretical basis for the method of feasible directions (MFD) was originally developed by Zoutendyk (ref. 1). In engineering, the text by Vanderplaats (ref. 2) contains details on the actual implementation of the method. Vanderplaats' algorithm, based on the method of feasible directions and published in 1983 (ref. 3) has had a great impact on engineering optimization. The program, as discussed here, is based on the same basic theory as that cited in references 2 and 3 although the implementation aspects are different. A major challenge in developing software for engineering optimization is in implementing conceptual algorithms. The main purpose of this report is to describe in detail various implementation aspects. It also serves as a users guide and describes the role played by the various parameters guiding the iterations in the program.

The basic steps in the method of feasible directions involve solving a quadratic program (QP) to find the direction vector and then to find the step size along this direction by performing a constrained one-dimensional line search. In the implementation herein, an interior method based on Karmarkar's approach (ref.4) is used to solve the QP problem for direction finding. Interior methods have been found to be effective for problems with a large number of variables. Also, special attention is given to the treatment of infeasible starting designs, to the dynamic adjustment of the constraint thickness parameter during the iterations, and to line search strategies. The optimization problem follows:

$$\text{Minimize} \qquad F(\mathbf{DV}) \tag{1}$$

$$\text{subject to} \qquad G_j(\mathbf{DV}) \le 0 \qquad j = 1,\ldots,\text{NCON} \tag{2}$$

$$\text{and} \qquad \mathbf{DVL}_i \le \mathbf{DV}_i \le \mathbf{DVU}_i \qquad i = 1,\ldots,\text{NDV} \tag{3}$$

where F is an objective function to be minimized subject to the constraints $G_j \le 0$, **DV** is an (NDVX1) vector of design variables, NDV is the number of design variables, NCON is the number of constraints, and **DVL** and **DVU** are the lower and upper bounds of the design variables, respectively. Equality constraints may be included by means of a penalty function (ref. 2).

Here, the capital variable names **DV**, **DVL**, and **DVU** are those that are actually used in the program. The default values for the various parameters are given in appendix A.

Direction Finding and Line Search From Feasible Design Points

Direction Vector

Let $\mathbf{DV}^0$ be a current design point which satisfies constraints (2) and (3). The direction finding problem is to find a search direction **d** that will (1) point into the feasible region, or is "a feasible direction," and (2) reduce the objective function, or is "usable." If $\nabla F \cdot \mathbf{d} < 0$ and $\nabla G_j \cdot \mathbf{d} < 0$ for each active constraint, then the direction vector **d** is usable-feasible. To find a direction that is both feasible and usable, the following subproblem is posed:

$$\begin{aligned} &\text{Minimize} && \theta \\ &\text{subject to} && \nabla F \cdot \mathbf{d} \le \theta \\ &&& \nabla G_j \cdot \mathbf{d} \le \beta_j\ \theta, \quad j \text{ in } J_{EP} \\ &&& 1/2\ \mathbf{d}^T\ \mathbf{d} \le 1 \end{aligned} \tag{4}$$

In problem (4), J_{EP} is an active set defined by the union of active constraints and active bounds

$$J_{EP} = \left\{j:\ G_j + \text{EP} \ge 0\right\} \cup \{\text{Active lower/upper bounds}\} \tag{5}$$

where θ is an artificial variable, β_j is the push-off factor associated with the j^{th} constraint, and EP is the constraint thickness parameter. Again, see appendix A for the default value of EP and the values of the other parameters. The following notation is now introduced:

$$\mathbf{y} = [\mathbf{d},\ \theta]^T = \text{an } (\text{NDV}+1 \times 1) \text{ vector}$$

$$\mathbf{p} = [0,...,0,1]^T = \text{an } (\text{NDV}+1 \times 1) \text{ vector}$$

$$A^T = \begin{bmatrix} \nabla G_1, & -\beta_1 \\ \nabla G_2, & -\beta_2 \\ \cdots & \cdots \\ \nabla G_{\text{NAC}}, & -\beta_{\text{NAC}} \\ \nabla F, & -1 \end{bmatrix} \tag{6}$$

where A is a matrix of dimension (NAC+1 × NDV+1), NAC equals the total number of active constraints (including active bounds). All gradient vectors in (6) have been normalized to be unit vectors. The problem in (4) can now be written as follows:

$$\begin{aligned} &\text{Minimize} && \mathbf{p}^T\mathbf{y} \\ &\text{subject to} && \mathbf{A}^T\mathbf{y} \le 0 \\ &\text{and} && 1/2\ \mathbf{y}^T\mathbf{y} \le 1 \end{aligned} \tag{7}$$

The dual formulation corresponding to problem (7) can be written as (ref. 2)

$$\begin{aligned} &\text{Minimize} && 1/2\ \mu^T \left[\mathbf{A}^T\mathbf{A}\right]\ \mu + \mu^T\left(\mathbf{A}^T\mathbf{p}\right) \\ &\text{subject to} && \mu \ge 0 \end{aligned} \tag{8}$$

where μ is an (NAC+1 × 1) vector. Problem (8) is a QP subproblem which can be solved in a number of ways. Once the QP is solved, the solution to (6) is obtained from

$$\mathbf{y} = -\mathbf{p} - \mathbf{A}\,\mu \tag{9}$$

which gives the search direction **d**.

Solution of the QP Using an Interior Approach

Interior methods gained greater acceptance after Karmarkar's paper in 1984. A variation of Karmarkar's approach, called the "linear affine scaling" algorithm, has also been developed and applied to linear programming problems (ref. 4). The approach was extended here to solve the QP in (8). The algorithm begins by first setting an initial estimate, $\mu^0 > 0$. The main steps are

(1) A transformation to a new set of variables μ^{new} is defined by

$$\mu = \mathbf{D}\ \mu^{\text{new}}$$

where **D** is a diagonal matrix whose elements are equal to the components of μ^0 = current value of μ. Thus, the current value of the new variables will always equal $[1,...,1]^T$.

(2) The search direction **s** is defined to be the steepest descent direction in the new space of variables

$$\mathbf{s} = -\mathbf{D}\left(\mathbf{A}^T\mathbf{A}\mu + \mathbf{A}^T\mathbf{p}\right)$$

(3) The step size ρ along **s** is obtained from

$$\rho = \text{Minimum}\ (\rho_1,\ \rho_2)$$

where ρ_1 is 98 percent of the maximum step to the boundary (to ensure that μ is positive), and ρ_2 is the minimum of the quadratic function along **s**. The variables at k^{th} iteration are updated from

$$\mu^{k+1} = \mu^k + \rho\ \mathbf{D}\,\mathbf{s} \tag{10}$$

and we go back to step 1. These steps are repeated until changes in the objective function of the QP are relatively small for three consecutive iterations.

Step Size Determination

At a feasible point $\mathbf{DV}^0$, we have now found a search direction **d** which is both usable and feasible. The step size α along **d** is determined in the program as follows. First, an initial step $\alpha_{\text{init}} \geq \alpha_{\text{min}} > 0$ is chosen based on the following criteria:

$$\alpha_{\text{init}} = \text{Minimum}\ (\alpha_B,\ \alpha_F,\ \alpha_{ML}) \tag{11}$$

where

α_B maximum step based on lower and upper limits on **DV**

α_F value of α that reduces the objective by 5 percent

α_{ML} value of α that results in a 15-percent maximum change in a design variable

Once the initial step is determined, a move is initiated based on the golden section ratio GR of 0.618. Thus, steps α_{init}, α_{init}/GR, $\alpha_{\text{init}}/(GR)^2$, . . . are taken along **d**, with $\alpha \leq \alpha_B$. At every step in this move, the design variables are updated as

$$\mathbf{DV}(\alpha) = \mathbf{DV}^0 + \alpha \cdot \mathbf{d} \tag{12}$$

and the objective and constraints are evaluated. If at any stage the constraints are violated, then the previous feasible point and the current infeasible point define an interval containing the zero of the constraint function. A bisection scheme is used to determine this zero, which determines the maximum limit on the step size. The bisection process is terminated when the maximum constraint value G_{max} satisfies

$$-G_{\text{low}} \leq G_{\text{max}} \leq G_{\text{high}} \tag{13}$$

where G_{high} is set equal to zero and should not be changed; G_{low} is set equal to 0.001 by default. If the objective function is linear, then the root of the constraint function is the desired step size α_0.

The other possibility is that the objective function is nonlinear along **d**. Thus, we also monitor the directional derivative of the objective function along **d** at every step during the move from

$$F'(a) = \nabla F[\mathbf{D}\,V(\alpha)] \cdot \mathbf{d}$$

Since **d** is a usable direction, $F' < 0$ at $\alpha = 0$. If $F' > 0$ at any feasible α during the move, then the previous point where F' is negative and the current point where F' is positive defines an interval containing the minimum of F. A bisection scheme is used based on computing $F'(\alpha)$ to determine the minimum. The bisection scheme is terminated when the interval length is less than a small number (TOLINT).

Once the step size α_0 is determined, the new design variables are obtained from

$$\mathbf{DV}^1 = \mathbf{DV}^0 + \alpha_0\,\mathbf{d}$$

We reset $\mathbf{DV}^1$ to equal $\mathbf{DV}^0$ and repeat the process. That is, we again find a new search direction and step size.

Selection and Adjustment of Push-Off Factors

The push-off factors β_j entering into the direction finding problem in (4) are determined from equation (2):

$$\beta_j = \beta_0 \cdot \left(1 + G_j/\text{EP}\right)^2 \tag{14}$$

where β_0 is equal to 1 by default. The push-off factors are set equal to zero for bounds and linear constraints. Push-off factors are also adjusted based on the following.

The solution of the QP in (8) and in equation (9) gives us a vector **d**. However, if $\nabla F \cdot \mathbf{d} < 0$ but $\nabla G_p \cdot \mathbf{d} > 0$ for some active p, then the search direction is usable but not feasible. In this case, the corresponding push-off factor β_p is increased a little and **d** recomputed. This increase is only attempted five times after which EP is reduced as discussed in the following section. In theory, the vector **d** from the QP solution should equal zero in such situations where a usable-feasible vector cannot be determined. However, it should be noted that the interior method yields solutions which are interior to the boundary. Round-off errors may also lead to such situations.

Adjustment of Constraint Thickness Parameter EP

The constraint thickness parameter EP is used to determine the active set as given in (5). The default value at the beginning of each design iteration is $\text{EP} = \text{EP}_0 = 0.01$. If at any stage, the QP solution yields a search direction **d** that is not usable-feasible, then EP is reduced. This happens particularly as the optimum is approached. We iteratively reduce EP as

$$\text{EP}^{\text{new}} = \text{EP}^{\text{old}}/3 \tag{15}$$

which is done at most five times after which the program terminates with the final solution.

Stopping Criteria

The three stopping criteria used are as follows:

Kuhn-Tucker condition.—First, determine if $|\mathbf{d}| \leq$ TOLKKT, where TOLKKT is a small number ($=10^{-4}$ by default) and $\mathbf{d}$ is the search direction. Usually, this stop criterion is satisfied only for simple and small problems.

Changes in the objective function.—Once we have obtained a usable-feasible direction, a line search is performed to obtain a new point with a lower value for the objective function. If either absolute changes in F or relative changes in F are small for three consecutive iterations, then the program terminates. Thus, we check

$$\begin{gathered}\text{abs (change in } F) \leq \text{TOLABS} \\ \text{abs (change in } F/F) \text{ TOLREL}\end{gathered} \tag{16}$$

Default values of TOLABS and TOLREL are 10^{-6}. Of course, if F has a very large magnitude, then TOLREL can be set equal to a smaller value.

Successive reductions in the constraint thickness parameter.—If, after successive reductions in EP, there is no usable-feasible direction, then the program terminates with the final solution.

Direction Finding and Line Search From an Infeasible Starting Point

If the starting design $\mathbf{DV}^0$ is infeasible, with at least one constraint $G_j > 0$, the violations are corrected first. Then proceed with the algorithm as described in the preceding section. At an infeasible point, the dot products of the violated constraint functions with the direction vector should be negative. Thus, we define the following subproblem for direction finding:

$$\begin{aligned}&\text{Minimize} && \theta \\ &\text{subject to} && \nabla G_j \cdot \mathbf{d} \leq \beta_j\, \theta,\ j \text{ in } J_{EP} \\ &&& 1/2\, \mathbf{d}^T \mathbf{d} \leq 1\end{aligned} \tag{17}$$

In reference 2, the objective function also enters into the subproblem. Herein, the goal is to exclusively correct constraint violations. As it turns out, the problem in (17) can be expressed in the form of a QP as given in (8) with only one difference: the last row in the $\mathbf{A}^T$ matrix in (6) is deleted.

Line Search

Assume that the QP in (17) has been solved and a search direction $\mathbf{d}$ has been obtained for constraint correction. First, α_B is determined along $\mathbf{d}$ so that lower and upper bounds on the design variables are satisfied. Within the interval $[0, \alpha_B]$, a move is initiated to determine the step size α_0. At any point in this move, if the maximum constraint value $G_{\max} \leq 0$, then the constraint correction phase terminates and we switch over to the algorithm in the section Direction Vector.

If the violation continues to reduce along $\mathbf{d}$ and $\alpha = \alpha_B$, then we set α equal to α_B and compute a new search direction for constraint correction. On the other hand, if we notice the violations to reduce and then increase, then the interval containing α_0 can be bracketed. That is, if there are three points α_1, α_2, α_3 for which $G_{\max_1} > G_{\max_2} > G_{\max_3}$, then we have $\alpha_1 \leq \alpha_0 \leq \alpha_3$. A golden section search is then performed within this interval to obtain the minimum of $G_{\max}$. A new search direction is again computed at the new point and this process is repeated until a feasible design is obtained.

Push-Off Factors and Constraint Thickness Parameter

For the constraint correction subproblem in equation (17), push-off factors β_j are computed from equation (14). However, if any $\beta_j >$ POMAX, then β_j is set equal to POMAX. The default value of POMAX is 50.

Sometimes, it is not possible to find a direction vector that makes negative dot products with all the active or violated constraints. In this case, the constraint thickness parameter EP is decreased in order to reduce the number of constraints in the active set. For lower and upper bounds, EP is reduced as given in equation (15). For the constraints G_j, EP is reduced by the formula

$$\text{EP}^{\text{new}} = \text{EP} - 0.5\ (\text{EP} + G_{\max}) \tag{18}$$

Since $G_{\max} > 0$, EP can take on a negative value. Equation (18) is applied a maximum of five times after which an error message is printed out to the effect that no feasible design can be found. The strategy in equation (18) is new and is quite effective.

Users Guide

The program MFD has been written in QBASIC as well as in Fortran. This section describes the user-supplied subroutine with an example (source listing in appendixes B and C). Appendix A contains a description of the main parameters used by the program together with their default values which may be

changed in the user routine.

User-Supplied Subroutine

To run the program, three steps are involved:

FIRST.—Upon initiation, the program will ask for the values of NDV and NCON, which are the number of design variables and the number of constraints.

SECOND.—In SUBROUTINE USER, there are four "CALL XXXX" statements which call the user-supplied subroutine. The user should give the appropriate name of the user-supplied subroutine in these CALL statements.

THIRD.—The user must supply a subroutine. This is now explained.

The user routine will be called with INF equal to −1, 1, or 2.

INF = −1.—This is the first call to the user routine. The initial values of the variables together with their lower/upper bounds have to be defined in arrays **DV**, **DVL**, **DVU**, respectively. Also, the user may change the defaults of the parameters as discussed subsequently in appendix A. For example, IGRAD = 1 may be set indicating that gradients will not be supplied, the number of iterations ITLIM may be set, or LINEARF = 1 may be set if the objective function is linear. The general appearance of statements will be

IGRAD ITLIM LINEARF	= See appendix B for explanations and default values.
DV	= Initial guess of the design variables
DV	= Lower bounds
DVV	= Upper bounds

INF = 1.—The objective and constraint functions have to be defined for the given values of the variables contained in the array **DV**. That is, we need to supply

F	=
G(1)	=
G(2)	=
G(NCON)	=

INF = 2.—If IGRAD = 0 (default), then gradients of the objective function and of the active constraints have to be supplied in the matrices DF and AA, respectively. The active constraint numbers are supplied to the routine in the array IACT(1),...,IACT(NAC). If NAC = 0 at any iteration, then there are no active constraints and only the objective function gradient DF needs to be computed.

For example, let NDV = 3, NCON = 10, NAC = 2, IACT (1) = 3, IACT(2) = 9. Then, we must supply

DF(1) = , DF(2) = , DF(3) =

AA(1,1) = , AA(2,1) = , AA(3,1) = ,- Gradient of third constraint in first column of [AA]

AA(1,2) = , AA(2,2) = , AA(3,2) = ,- Gradient of ninth constraint in second column

Example Problem

Consider the Rosen-Suzuki problem:

$$\text{Minimize} \quad F = X1^2 + X2^2 + 2\,X3^2 - X4^2 - 5\,X1 - 5\,X2 - 21\,X3 + 7\,X4 + 100$$

subject to

$$X1^2 + X2^2 + X3^2 + X4^2 + X1 - X2 + X3 - X4 - 8 \le 0$$

$$X1^2 + 2\,X2^2 + X3^2 + 2\,X4^2 - X1 - X4 - 10 \le 0$$

$$2\,X1^2 + X2 + X3^2 + 2\,X1 - X2 - X4 - 5 \le 0$$

The solution to this is (0.15, 1.0, 1.87, −1.14) with F^{opt} equal to 53.6. User routines for this problem with IGRAD = 1 and also a spring design problem with IGRAD = 0 are provided in appendix B. Note that the common blocks are required in the routine along with the double precision statement.

Test Problems

The program MFD has successfully solved a variety of test problems. The results for only three truss optimization problems will be given here. The results tally with those obtained from using other optimization codes from a test bed that was developed at NASA Lewis Research Center (ref. 5).

Ten-Bar Truss

This problem, given in reference 6, is to minimize the weight of the 10-bar truss shown in figure 1. The data are as follows: Young's modulus $E = 10^7$ psi; $\rho = 0.1$ lb/in.^{3}; allowable stress $\alpha_{all} = 25\,000$ psi for elements 1 to 8 and 10; $\sigma_{all} = 75\,000$ psi for element 9; NDV = 10; NCON = 10; the lower bounds on each area = 0.1 in.^{2}; the upper limit = 20.0 in.^{2}. In figure 1, load $P = 100\,000$ lb. The initial design variables were 4.0 in.^{2} (infeasible) with the initial weight $F^0 = 1678.6$ lb.

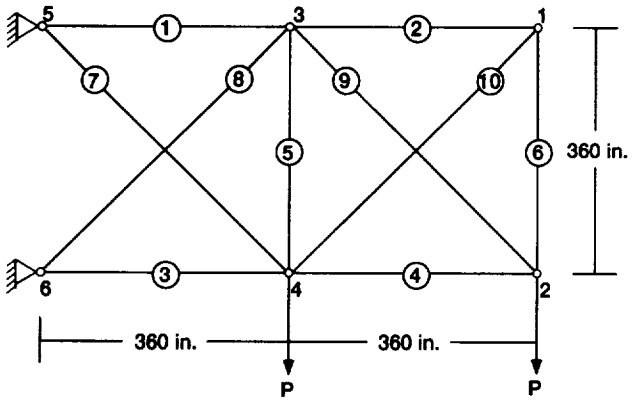

Figure 1.—Ten-bar truss.

The optimum design is as follows:

$$F^{opt} = 1498.3 \text{ lb}$$

$$\mathbf{DV}^{opt} = \{7.92,\ 0.1,\ 8.10,\ 3.90,\ 0.1,\ 0.1,\ 5.80,\ 5.52,\ 3.67,\ 0.14\} \text{ in.}^2$$

(The cross-sectional areas given above are rounded to two decimal places.)

Active set: stresses in elements 1,2,3,4,6,7,8,10

The results from other codes are given in table I. Program MFD solved this problem in 127 iterations with 1007 function evaluations. Because this number may seem large for a 10-bar truss, detailed discussions on this aspect are presented in the next section.

Tapered 10-Bar Truss

This problem is a variation of the previous problem (see fig. 2). There are constraints on the stress in each of the 10 elements and also on the displacements in the Y-direction of nodes 3 and 4. The constraints are to be satisfied under two loading conditions (table II). Thus, NDV = 10 and NCON = 24; $E = 10^7$; $\rho = 0.1$ lb/in.^{3}; $\sigma_{all} = 10\,000$ psi in each of the 10 elements; the lower bounds = 0.1 in.^{2}; the upper bounds = 100 in.^{2}. The initial design is (1.,...,1.) in.^{2} with $F^0 = 146$ lb (infeasible). The optimum solution is

$$F^{opt} = 3269.0 \text{ lb}$$

$$\mathbf{DV}^{opt} = \{57.34,\ 18.66,\ 2.25,\ 7.41,\ 36.13,\ 3.44,\ 28,79,\ 19.69,\ 16.20,\ 8.94\} \text{ in.}^2$$

Active set for load case 1: stresses in elements 1,7,8 and displacement at node 3

Active set for load case 2: stress in element 6

MFD took 121 iterations and 1091 function evaluations. Table I compares the performance of other codes for this problem.

Sixty-Bar Trussed Ring

The trussed ring in figure 3 consists of 60 elements and is subjected to 3 loading conditions (table III). The data are $E = 10^7$ psi; $\rho = 0.1$ lb/in.^{3}; $\sigma_{all} = 10\,000$ psi for each element; node 4 has a displacement limit of 1.5 in. in the Y-direction; the lower and upper limits on areas are 0.1 and 100.0 in.^{2}, respectively. The 60 element areas are linked to the 25 design variables given in table IV. Here, NDV = 25 and NCON = 183.

TABLE I.—OPTIMUM WEIGHTS FROM VARIOUS CODES
[Number of function evaluations shown in paretheses.]

Method	Truss optimization problem		
	Ten-bar	Tapered ten-bar	Ring
	Optimum weight, F, lb		
Method of feasible direction (MFD)	1500 (1007)	3269 (1091)	345 (1836)
Sequential unconstrained minimizational techinque (SUMT)[a]	1503 (243)	3270 (667)	345 (369)
Feasible direction (FD)[a]	1498 (73)	3424[b] (77)	343 (78)
Sequential quadratic programming (SQP)[a]	(c)	3271 (187)	344 (782)

[a]Reference 5.
[b]A different starting point was necessary.
[c]A feasible design using the default parameters in the code could not be obtained.

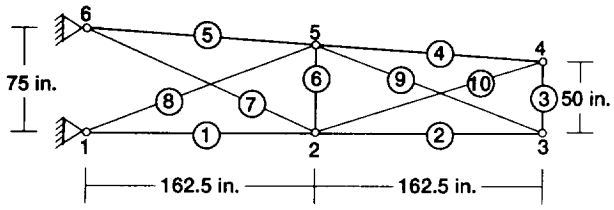

Figure 2.—Tapered 10-bar truss.

tively. The 60 element areas are linked to the 25 design variables given in table IV. Here, NDV = 25 and NCON = 183. The initial design is (1.,...,1.) $in.^2$ with an associated weight of 250 lb. The optimum design is

$$F^{opt} = 345.0 \text{ lb}$$

$$\mathbf{DV}^{opt} = \begin{Bmatrix} 1.19 & 2.17 & 0.14 & 2.18 & 2.13 \\ 0.68 & 2.20 & 2.20 & 1.08 & 2.26 \\ 2.18 & 0.26 & 2.33 & 1.25 & 1.22 \\ 0.88 & 0.88 & 1.23 & 1.12 & 1.15 \\ 1.27 & 1.15 & 0.88 & 1.23 & 1.26 \end{Bmatrix}$$

Active set for load case 1: stresses in elements 25, 49, and displacement at node 4

Active set for load case 2: stresses in elements 5, 14, 20

Active set for load case 3: stress in element 11

The optimum was obtained in 247 iterations with 1836 function evaluations. Again, see table I for the performance of other codes.

Discussion

TABLE II.—LOADING SPECIFICATIONS FOR TAPERED 10-BAR TRUSS

Load condition	Node	Load direction	
		X	Y
		Load, *P*, lb	
Case 1	2	60 000	120 000
	3	60 000	60 000
	4	17 500	12 500
	5	17 500	25 000
Case 2	2	0	−50 000
	3	↓	−25 000
	4	↓	−37 500
	5	↓	−75 000

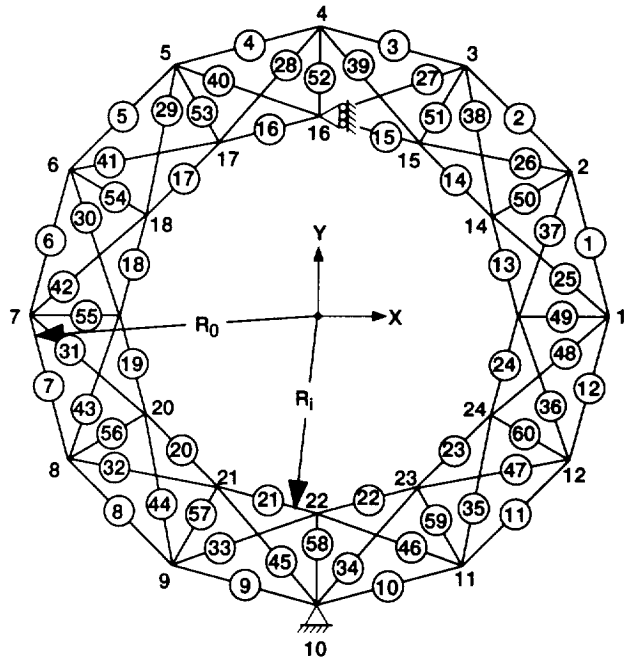

Figure 3.—Sixty-bar trussed ring.

In this report, the detailed implementation of a feasible directions code was explained with particular emphasis on line search, infeasible starting points, and the role of various parameters used in the algorithm.

The robustness of the program was very satisfactory for a variety of test problems, both with regard to correcting an infeasible starting point and to making subsequent progress towards an optimum solution. Its robustness may be attributed to the use of the interior method for direction finding and to the use of a bisection scheme in the line search.

The following factors contribute to the inefficiency of the code:

(1) A plot of truss volume (cost) versus the number of iterations for the 10-bar truss problem is shown in figure 4. We see that substantial reductions take place in the first 40 iterations while the remaining 120 iterations are needed only to achieve the convergence criterion of 10^{-6}. This asymptotical characteristic is typical of most mathematical programming codes. In an interactive setting, the user can terminate the optimization run as soon as the cost-versus-iteration curve flattens out.

(2) In the final design, there are no constraint violations whatsoever because of the line search control described by equation (13), which is in contrast to certain other programs that yield final designs with small violations.

(3) Push-off factors are user-defined. Their default values are unity. Their selections can be optimized for categories of problems, truss being one. Such studies have not been carried out here.

(4) Efficiency is sacrificed when pure bisection is used. In the

TABLE III.—LOADING SPECIFICATIONS FOR 60–BAR TRUSSED RING

Load condition	Node	Load direction	
		X	Y
		Load, *P*, lb	
Case 1	1	−10 000	0
	7	9 000	0
Case 2	15	−8 000	3 000
	18	−8 000	3 000
Case 3	22	−20 000	10 000

TABLE IV.—GROUPING OF ELEMENTS FOR 60-BAR TRUSSED RING

Design group	Element	Design group	Element
1	49–60	13	12, 24
2	1, 13	14	25, 37
3	2, 14	15	26, 38
4	3, 15		
5	4, 16	16	27, 39
		17	28, 40
6	5, 17	18	29, 41
7	6, 18	19	30, 42
8	7, 19	20	31, 43
9	8, 20		
10	9, 21	21	32 ,44
		22	33, 45
11	10, 22	23	34, 46
12	11, 23	24	35, 47
		25	36, 48

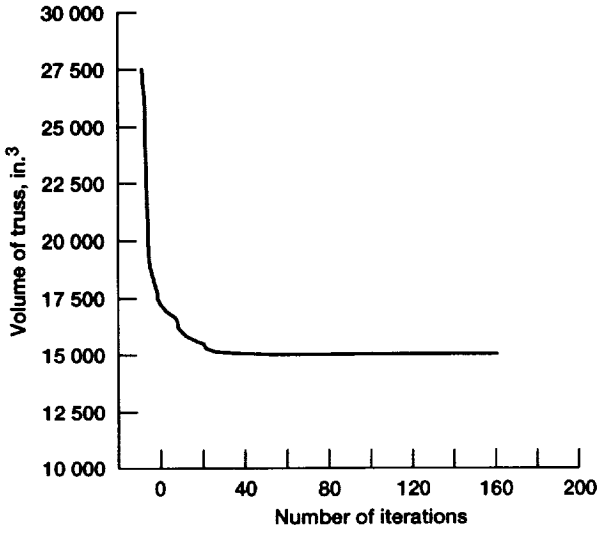

Figure 4.—Iteration history for 10-bar truss.

test problems, a line search within the feasible region has taken between 5 and 10 function evaluations (analyses) as determined by taking the ratio of the number of function evaluations to the number of iterations. There is latitude here to improve the efficiency of the algorithm by using hybrid approaches which combine sectioning with polynomial interpolations for line search.

Conclusions

The method of feasible directions incorporates Karmarkar's algorithm for the generation of search directions and the bisection technique for improvements in one-dimensional search.

The algorithm provides a feasible optimum design for problems with complex constraint space even when the initial design is infeasible.

For selected problems the computer code may require more computations than those that may be needed by other nonlinear programming software. However, there is latitude to improve the computational efficiency of the algorithm.

The computer code successfully solved several structural optimization problems.

Lewis Research Center
National Aeronautics and Space Administration
Cleveland, Ohio, November 2, 1993

Appendix A

Program Parameters and Their Default Values

This section describes various parameters used by the program and their default values. The user may change any of these values in the user-supplied subroutine when INF = –1 as explained in the section User-Supplied Subroutine. (The important parameters are indicated with an asterisk.)

ALPMIN / 0.001 /—minimum step length used in line search procedures

ALPQP / 0.98 /—a parameter used to come near the boundary in the interior approach to solve the QP

AMLIMIT, OBJFAC, ALPMIN /0.15, 0.05, 0.001/—used during line search

*EP /0.01/—constraint thickness parameter (base value)

*EPBD /0.01/—constraint thickness parameter for lower and upper bounds

*EPSPCT, EPSMIN /0.01, 0.001/—parameters relevant only when IGRAD = 1 for divided-difference gradients (The gradient f' is evaluated by {f(x+e) –f(x)}/e, where e = maximum (EPSPCT × abs(x), EPSMIN).)

GLOW, GHIGH /0.001, 0/—used during line search to determine the zero of a constraint function (GHIGH should not be changed from its zero value.)

*IGRAD / 0 /—IGRAD = 0 means that the user will provide gradients of objective and active constraint functions in the user-supplied subroutine (see next section). IGRAD = 1 means that gradients will not be provided by the user; they will be internally calculated in the program using divided differences. See EPSPCT and EPSMIN

ITEPMAX / 5 /—maximum number of reductions in the constraint thickness parameter EP to obtain a usable-feasible direction or to obtain a feasible direction during constraint correction

*ITLIM /150/—maximum number of iterations

ITLIMINF / 30 /—maximum number of iterations to obtain a feasible design from an infeasible starting point

ITPUSHMAX / 5 /—maximum number of iterations to increase the push-off factors to obtain a usable-feasible search direction

*LINEARF / 0 /—LINEARF = 0 if objective function is nonlinear and = 1 if objective function is linear

LINEAR(J) /0,...,0/— = 0 if constraint function is nonlinear and = 1 if constraint is linear

*MAXACT / 50 /—maximum number of active constraints; used for dimensioning purposes

POMAX / 50. /—maximum push-off factor used during constraint correction phase for infeasible starting designs

TOLINT / 0.001 /—used as stopping criterion during line search (associated with nonlinear objective functions)

TOLKKT / 0.0001 /—stopping criterion used for checking Kuhn-Tucker conditions

*TOLABS, TOLREL /1.e-6, 1.e-6/—absolute and relative tolerances used for stopping criteria (see (16))

TOLABS1, TOLREL1, ITMAX / 1.E-6, 1.E-6, 100 /—stopping parameters associated with the interior method for solving the QP for the direction vector

Appendix B

Fortran User Subroutines with IGRAD = 1 and IGRAD = 0

ROSEN-SUZUKI PROBLEM : IGRAD = 1 (divided-difference gradients)

```
      SUBROUTINE ROSEN
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /MFD1/ INF, NDV, NCON, NFV, IGRAD, EP, NAC, EPSPCT, EPSMIN,
     $       MAXACT, EPBD
      COMMON /MFD2/ DV(100), IACT(50), LINEAR(1000), F, G(1000),
     $ DF(100), AA(101,51)
      COMMON /MFD6/ DV0(100), F1, FM, G1(1000), D(100), DVL(100),
     $       DVU(100), BETA(1000)
      IF (INF .EQ. -1) THEN
C       NDV = 4, NCON = 3
        IGRAD = 1 ..... for divided-difference gradients
        DV(1) = 1
        DV(2) = 2
        DV(3) = 3
        DV(4) = 4.....DV contains initial values of design variables
        DVL(1) = -10
        DVL(2) = -10
        DVL(3) = -10
        DVL(4) = -10.....lower bounds
        DVU(1) = 10
        DVU(2) = 10
        DVU(3) = 10
        DVU(4) = 10.....upper bounds
        RETURN
      END IF
      B1 = DV(1)
      B2 = DV(2)
      B3 = DV(3)
      B4 = DV(4)
      NFV = NFV + 1
         .....Objective and constraint functions

      F = B1 ** 2 + B2 ** 2 + 2 * B3 ** 2 - B4 ** 2 - 5 * B1 - 5 *
     $   B2 - 21 * B3 + 7.*B4 + 100.
      G(1) = B1 ** 2 + B2 ** 2 + B3 ** 2 + B4 ** 2 + B1 - B2 + B3 - B4-8
      G(2) = B1 ** 2 + 2 * B2 ** 2 + B3 ** 2 + 2 * B4 ** 2 - B1- B4 - 10
      G(3) = 2 * B1 ** 2 + B2 ** 2 + B3 ** 2 + 2 * B1 - B2 - B4 - 5
      RETURN
      END
```

Appendix C

A Spring Design Problem : IGRAD = 0 (User-Supplied Gradients)

```
      SUBROUTINE SPRING
      IMPLICIT REAL*8(A–H,O–Z)
      COMMON /MFD1/ INF, NDV, NCON, NFV, IGRAD, EP, NAC, EPSPCT, EPSMIN,
     $      MAXACT, EPBD
      COMMON /MFD2/ DV(100), IACT(50), LINEAR(1000), F, G(1000),
     $ DF(100), AA(101,51)
      COMMON /MFD6/ DV0(100), F1, FM, G1(1000), D(100), DVL(100),
     $      DVU(100), BETA(1000)
      IF (INF .EQ. –1) THEN
C     SPRING PROBLEM — NDV = 3: NCON = 4
            .....IGRAD = 0 by default
       LINEAR(4) = 1 ..... Constraint #4 is linear
       DV(1) = 1
       DV(2) = 2
       DV(3) = 3
       DVL(1) = .05
       DVL(2) = .1
       DVL(3) = 1
       DVU(1) = 1
       DVU(2) = 5
       DVU(3) = 50
       RETURN
      END IF
      B1 = DV(1)
      B2 = DV(2)
      B3 = DV(3)
      IF (INF .EQ. 2) GOTO 701
            ..... INF = 1 ( supply F and G_j)
      NFV = NFV + 1
      F = (B3 + 2) * B1 ** 2 * B2
      G(1) = 1 – B2 ** 3 * B3 / 71875 / B1 ** 4
      G(2) = (4 * B2 ** 2 – B1 * B2) / 12566 / (B2 * B1 ** 3 – B1 ** 4)
      G(2) = G(2) + 1 / 5108 / B1 ** 2 – 1
      G(3) = 1 – 140.45 / B2 ** 2 / B3
      G(4) = (B1 + B2) / 1.5 – 1
      RETURN
..... Objective gradient in DF
701   DF(1) = (B3 + 2) * 2 * B1 * B2
      DF(2) = (B3 + 2) * B1 ** 2
      DF(3) =  B2 * B1 ** 2
      IF (NAC .EQ. 0)RETURN ... NAC = number of active constraints
      DO 100 II = 1, NAC
..... Constraint gradients placed in columns of AA- matrix using active set IACT
      IF (IACT(II) .EQ. 1) THEN
       AA(1, II) = 4 * B2 ** 3 * B3 / 71875 / B1 ** 5
       AA(2, II) = –3 * B2 ** 2 * B3 / 71875 / B1 ** 4
       AA(3, II) = –B2 ** 3 / 71875 / B1 ** 4
      END IF
```

```
      IF (IACT(II) .EQ. 2) THEN
        C1 = 12566 * (B2 * B1 ** 3 – B1 ** 4)
        C2 = 4 * B2 ** 2 – B1 * B2
        C3 = 5108 * B1 ** 2
        CC = –C1 * B2 – C2 * 12566 * (3 * B2 * B1 ** 2 – 4 * B1 ** 3)
        CC = CC / C1 ** 2
        CC = CC – 2 / 5108 / B1 ** 3
        AA(1, II) = CC
        CC = (C1 * (8 * B2 – B1) – C2 * 12566 * B1 ** 3) / C1 ** 2
        AA(2, II) = CC
        AA(3, II) = 0
      END IF
      IF (IACT(II) .EQ. 3) THEN
        AA(1, II) = –140.45 / B2 ** 2 / B3
        AA(2, II) = 2 * 140.45 * B1 / B2 ** 3 / B3
        AA(3, II) = 140.45 * B1 / B2 ** 2 / B3 ** 2
      END IF
      IF (IACT(II) .EQ. 4) THEN
        AA(1, II) = 1 / 1.5
        AA(2, II) = 1 / 1.5
        AA(3, II) = 0
      END IF
100   CONTINUE
      RETURN
      END
```

References

1. Zoutendyk, G.: Method of Feasible Directions. Van Nostrand, Princeton, NJ, 1960.
2. Vanderplaats, G.N.: Numerical Optimization Techniques for Engineering Design: With Applications. McGraw-Hill, New York, 1984.
3. Vanderplaats, G.N.: A Robust Feasible Directions Algorithm for Design Synthesis. Structures, Structural Dynamics and Materials Conference, 24th Proceedings, Pt. 1, AIAA, New York, 1983, pp. 392–399. (Also, AIAA Paper 83–0938, 1983.)
4. Klein, C.M.: Implementing Interior Point Methods in the Classroom. Int. J. Appl. Eng. Ed., vol. 6, no. 4, 1990, pp. 431–435.
5. Guptill, J.D., et al.: Comparative Evaluation Test Bed of Optimizers and Analyzers for the Design of Structures. NASA TM–4537, 1993.
6. Haftka, R.T.; and Gurdal, Z.: Elements of Structural Optimization. Kluwer Academic Publishers, Boston, MA, 1992.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1994	Technical Memorandum

4. TITLE AND SUBTITLE

An Optimization Program Based on the Method of Feasible Directions: Theory and Users Guide

5. FUNDING NUMBERS

WU-505-63-5B

6. AUTHOR(S)

Ashok D. Belegundu, Laszlo Berke, and Surya N. Patnaik

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E-8124

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM-4552

11. SUPPLEMENTARY NOTES

Ashok D.Belegundu, The Pennsylvania State University, Department of Mechanical Engineering, University Park, Pennsylvania 16802; Laszlo Berke, NASA Lewis Research Center; and Surya N. Patnaik, Ohio Aerospace Institute, 22800 Cedar Point Road, Brook Park, Ohio 44142. Responsible person, Laszlo Berke, (216) 433-5648.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 39

12b. DISTRIBUTION CODE

13. ABSTRACT (***Maximum 200 words***)

The theory and user instructions for an optimization code based on the method of feasible directions are presented. The code was written for wide distribution and ease of attachment to other simulation software. Although the theory of the method of feasible direction was developed in the 1960's, many considerations are involved in its actual implementation as a computer code. Included in the code are a number of features to improve robustness in optimization. The search direction is obtained by solving a quadratic program using an interior method based on Karmarkar's algorithm. The theory is discussed, focusing on the important and often overlooked role played by the various parameters guiding the iterations within the program. Also discussed is a robust approach for handling infeasible starting points. The code was validated by solving a variety of structural optimization test problems that have known solutions obtained by other optimization codes. It has been observed that this code is robust: it has solved a variety of problems from different starting points. However, the code is inefficient in that it takes considerable CPU time as compared with certain other available codes. Further work is required to improve its efficiency while retaining its robustness.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Optimization; Structures; Algorithms	15
	16. PRICE CODE: A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	