SPACE FRAMES - OPTIMIZATION STUDIES

VIBRATION ANALYSIS AND DESIGN OPTIMIZATION STUDIES OF SPACE FRAMES

III - OPTIMIZATION STUDIES

BY

V. GURUNATHAN

A Thesis
Submitted To The Faculty of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree
Master of Engineering

McMaster University

May, 1968

MASTER OF ENGINEERING (1968)
(Mechanical Engineering)

MCMASTER UNIVERSITY
Hamilton, Ontario.

TITLE : Vibration Analysis and Design Optimization Studies of Space Frames - Optimization Studies

AUTHOR : Viswanatha Gurunathan, B.E. (Madras University)
M.Tech. (I.I.T., Madras)

SUPERVISOR : Professor J. N. Siddall

NUMBER OF PAGES :

SCOPE AND CONTENT :

A general technique for the optimization of nonlinear functions including equality and inequality constraints has been developed and programmed for IBM 7040 Computer.

A detailed study was made of the possibility of optimizing a spatial structural model under static and dynamic constraints.

ACKNOWLEDGEMENTS

The author is deeply indebted to Professor J.N. Siddall and Dr. M.A. Dokainish for their invaluable guidance and assistance in the preparation of this thesis.

The author expresses his gratitude to the Department of Mechanical Engineering and the Canadian Armament Research and Development Establishment for the opportunity to participate in the project.

Further, the author wishes to express his appreciation to all who directly or indirectly offered help at various stages.

Finally, the assistance of Mrs. Kamala Gurunathan in typing the manuscript is thankfully acknowledged.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS	v
ABSTRACT	vi
INTRODUCTION	1
THE GENERAL CONSTRAINED OPTIMIZATION PROBLEM	12
AN ALTERNATE SEARCH ROUTINE	17
DESCRIPTION OF SEARCH TECHNIQUE	18
Description of General Code	25
Application of the Programme	30
STRUCTURAL OPTIMIZATION	34
Formulation of Constraints for Model	40
Optimization of Model	43
RESULTS AND CONCLUSIONS	47
FIGURES	50
APPENDICES	
I. Optimization Techniques	60
II. Computer Programmes	66

LIST OF ILLUSTRATIONS

FIGURES

1	Direct Search Hangs Up	50
2	Flow Chart For Alternate Search Routine	51
3	Exploratory And Pattern Moves	52
4	Structural Model	53
5	Terms in Matrix R	54
6	Sign Conventions	55

ABSTRACT

The optimization study of space frames has been considered in two aspects in this project work. The first was to develop a suitable optimization technique for a nonlinear programming problem including equality constraints, without any particular reference to structural optimization. The necessacity for the above requirement was due to the fact that almost all existing methods on optimization have some limitation. The second object of this study was to set up the necessary equations for the constraints on stress and on frequency for the structural model used, and then to use the developed technique to optimize the structural model for minimum weight.

A simple and effective strategy, which is a combination of direct search and linear approximate programming is believed to have been developed for optimization of simple nonlinear type equations.

The analysis of the space structure and the study of structural optimization revealed several difficulties inherent in the evaluation of constraining equations for the stresses and frequencies, which makes the optimization very difficult.

INTRODUCTION

This research programme has the general objective of establishing analytical techniques for analysis of indeterminate spatial frames and shells under dynamic loading, and the design optimization of these structures under the constraints of dynamic loading. Although techniques developed should have wide applicability, emphasis will be placed, for experimental and illustrative purposes, on structural configurations common to machine structures.

This present work relates to the third stage of this programme - the structural optimization problem with static and vibrational constraints. Related studies[39,40] will examine the static and dynamic analysis of the structure. The overall programme is in its early stages, the examination of a simple discrete space frame with generalized characterstics. The following discussion reviews the overall problem.

Design systhesis essentially is an evolutionary spiral process involving a complex feed back interrelating the fields of creativity, past experience and tools of analysis. The role of the designer is to optimize the value of a synthesis on the basis of some criteria through a balanced exploitation of the evergrowing information from all the three fields. The basic techniques and the criteria of evaluation themselves need refinement from time to time in the light of achievements in the foregoing areas.

The process has been marked with a rather slow progress in the field of mechanical engineering structures, mainly due to their complex nature. These have not received the intensive investigation that civil and aerospace engineering configurations have. Analysis of mechanical engineering structures has perhaps lagged behind because they are much more difficult to categorise than in the other fields where a few highly typical configurations can be recognised, modelled and studied in a concentrated way. In addition, the analytical tools available until lately have had their own limitations.

These methods can be broadly classified into two divisions,[1,2] -

(1) Methods based on exact solution of the differential equations describing the structure.

Apart from the difficulties in setting up and solving the equations subject often to awkward boundary conditions, in the case of complex structures the basic assumptions proved too restrictive for accurate solution.

(2) Approximate methods involving mathematical approximations

can be subclassified into -

(a) Those based on finite difference procedures. These are unsatisfactory in their formulation of boundary conditions and convergence characteristics, and

(b) Those which approximate the stress or displacement distribution by a series of analytical expressions and hence are unsuited for complex structures.

The classical analytical tools are thus incapable of providing an integrated approach even for structures of moderate complexity. Hence it is not surprising that the practical design of mechanical engineering structures has relied more on past practical experience supported by rough analytical checks wherever possible, rather than on the analytical tools.

The need for a tool well suited to complex configurations was most acute in the aircraft industry where the designer had to work within extremely narrow margins of practical expediency[3]. The extensive efforts over the years by numerous and often isolated workers culminated in the finite element approach which is a major breakthrough from the past.

Based on structural as against mathematical approximation, the method essentially seeks to idealise the structure into an assembly of a finite number of discrete elements connected at a finite number of points, and then proceeds to solve for the system response on an exact mathematical basis. It is the finite connectivity which permits a complex continuous structure to be analysed by a system of algebraic equations and forms the basis of the technique. Although earlier work was restricted to the field of aeronautical engineering, recently results of applications to non-aeronautical problems[4,5,6] and extensions to three dimensional discrete elements[7] have been reported.

It is realised that, although the finite element technique is still developing, it provides a unified approach to the analysis of any type of structural assembly, from any field and with any combination of one, two or three dimensional elements of different characteristics[4]. It thus provides a reliable anlytical tool which is a pre-requisite for design systhesis.

A rather limited amount of work appears to have been done on the general problem of elastic vibration of structures and the problem of optimization under vibrational constraints, although techniques for calculating the natural modes and frequencies of lumped mass spatial structures are fairly well established for essentially beam like aircraft structures, and to a lesser extent the rectangular frames of civil engineering. The significance of rotary inertia in spatial frames does not appear to have been studied. Archer[8,9] has provided two useful new papers in this field and has related it to the finite element stiffness matrix technique. Hurty[10] has developed a method for analysing complex structural systems that can be divided into interconnected components.

The concept of optimum design has registered a drastic change since the advent of high speed digital computers. Earlier, the magnitude of computation involved acted as a deterent and a feasible solution was accepted in lieu of the optimum. With computers to handle the arithmetic, systematic design synthesis has become a reality.

Very many general techniques of optimization appear in the literature that might be applied to structural optimization. Most promising are the Direct Search Method first suggested by Hooke and Jeeves and further developed by Flood and Leon[11], the Method of Successive Linear Approximation due to Griffith and Stewart[12], and the Random Method of Dickinson[13].

Minimization of weight, weight stiffness ratio, cost, volume for a homogeneous structure, etc. have been suggested as criteria for optimization of structures, but minimization of weight appears to have been accepted as the most satisfactory one even though the minimum weight design is not always the minimum cost design.

The optimization of a statically determinate truss subjected to single loading is a problem in analysis rather than synthsis. For strength design, member cross sections are proportioned to develop maximum allowable stress for the required failure mode. For optimum stiffness design based on minimization of weight per unit stiffness, stiffness being defined as the reciprocal of strain energy, the members should carry stresses proportional to the square root of the product of the modulus of elasticity and specific weight. The constant of proportionality is based on stiffness requirements[14].

For a given determinate truss under multiple load condition the problem essentially remains the same. All the member cross sections carry the maximum allowable stress, based on strength or stiffness design, at least under one load condition. The optimum design has come to be recognised as a fully stressed design.

In the case of indeterminate trusses, for a given configuration, applied loading and allowable stress, the cross sectional area of the members and hence the weight of the structure are functions of forces in the redundant members. Sved[15] has shown analytically that under single load conditions the minimum weight structure is always determinate.

Using the Lagrange multiplier technique, L.C. Schmidt[16] has shown that under alternative loads numerous fully stressed designs of an indeterminate truss exists. Due to the prohibitive nature of computations involved in arriving at the minimum weight he has suggested two complementary relaxation methods to arrive at a fully stressed design.

The beginning of the present decade marked a radical departure in the approach to structural optimization. It came to be accepted as a problem in mathematical programming with Schmit[17] as the pioneer. Utilising the joint force and displacement formulation of structural analysis as first proposed by Klein[18], he has optimized a fixed configuration three bar truss subject to three alternate loads. He treated it as a problem in nonlinear programming by adopting a modified steepest descent method designated as the method of alternate steps. On encountering an inequality constraint, which must be convex, the search moves along a constant weight plane in the feasible region until the constraint is again contacted. It then steps back halfway, and then continues to move along the steepest slope. On the basis of numerical results he concludes that in terms of design parameter space the minimum weight design need not be a fully stressed design lying at the apex of constraint hyperplanes.

Subsequently[19,20] in collaberation with Mallett and Kicher he extended the above to the problem of selecting a suitable configuration and material for the three bar truss. Various optimum designs were compiled by changing the material or configuration, one at a time in discrete steps. The best of all these design was chosen.

Dorn et al[21] have proposed a linear programming method which selects the optimum combination of configuration and member cross section from wide classes of admissible trusses defined by a given number of admissible joints connected in all possible ways by linear members. The optimization is based on a modified simplex method capable of handling large numbers of equations. The results provide an interesting study in the behaviour of optima due to change in load and the height-span ratio of the truss. The configuration remains the same for the load for a certain change in height-span ratio α, and then alters, as α continues to change. Thus a continuous spectrum is provided from which the value of α giving the absolute minimum weight truss and the configuration itself could be selected.

Best[22] has optimized a cantilever box beam by the steepest descent method. It has one unique feature. The partial derivatives of stress and deflections with respect to the design parameters are calculated by the finite difference approximation using the stiffness matrix, which must be inverted to obtain the deflections. To avoid the time consuming process of inversion at every step he adopts an interative scheme to obtain the deflections. Only the incremental stiffness matrix for a given change in design parameter is calculated which, in conjunction with the previously inverted stiffness matrix, rapidly converges to the required displacements on iterations. This feature is said to substantially reduce the calculation time. Constraints on stresses and deflections are handled by a version of the reduced gradient method. His solution is a maximum stress solution, and thus forced to be on a boundary.

The presentation of the structural synthesis as an unconstrained minimization problem by Schmit and Fox[23] is unique. It is based on the method of solving linear simultaneous equations by minimizing the sum of squares of the residuals to zero. This expression is set up for the equality constraints defining the stresses. To this is added penalty terms for violated inequality constraints, which are all simple upper and lower bounds. The actual quantity to be optimized, the weight, is treated as an inequality constraint, requiring that the weight be less than an arbitrarily defined draw down weight. The problem is now an unconstrained optimization problem solved by a gradient method. It is repeated using progressively lower draw-down weights until the optimization function cannot be made zero. This indicates that the draw-down weight is lower than the inherent minimum weight. The method thus actually requires a series of optimizations. It does not seem too applicable to complex problems, as the constraints must be expressed explicitly in order to set up the residuals. The implicit matrix form of equality constraints are ruled out.

Razani[24] has proposed an unconventional approach using an interative technique in which areas are changed by successive increments from an initial feasible solution so that each member is fully stressed in at least one of the several possible load conditions. This gives a feasible solution forced to be on a boundary. The true minimum may not be on a boundary if the stress is indeterminate.

The gradient projection technique has been successfully adopted by Brown and Alfredo[25] to optimize a portal frame and a two storey single bay frame. The search begins at a feasible starting point until constraints are encountered. At this point the constraint hypersurfaces are approximated by hyperplanes and the gradient of the objective function is projected on the line of intersection of these planes. After a move along the indicated direction a correction is indicated due to the nonlinearity of the constraint hypersurfaces. The authors have proposed the use of only one design parameter for a member as variable while the rest of the parameters for the same member are expressed as functions of the selected one. As moment of inertia of the members has a predominant effect on the behaviour of the structure, other parameters are expressed as functions of moment of inertia. Inspite of this simplification the procedure seems too involved for complex structures.

Young and Christiansen[14] have provided the first known optimal structural design technique using vibrational constraints using an iterative technique. Adjustment of the member area to achieve a fully stressed design simultaneously with the required resonant frequency characteristic is the main feature. An application toa pinjointed space truss is included.

This present work has two phases - the first is to develop a suitable optimization technique for a non-linear programming problem including equality constraints. The second is to set up the necessary equations for the constraints on stresses and frequency for the spatial structural model and then to see if the optimization routine developed, could be used to optimize the structural model for minimum weight, subject to the stress and frequency constraints.

THE GENERAL CONSTRAINED OPTIMIZATION PROBLEM

Definition Of The Problem

The general formulation of the optimization problem defines an optimization function to be maximized or minimized.

$$U(x_1, x_2, x_3, \ldots, x_n) = \text{maximum or minimum} \quad ..(1)$$

where U is a criterion variable such as cost, weight or capacity. The x's are independent variables such as dimensions, stresses, material properties and frequency. The device or system being optimized is constrained or defined by a set of inequalities or equalities, such as

$$\phi_i\,(x_1, x_2, \ldots, x_n) \geqslant 0, \quad i=1,m \quad \ldots \quad (2)$$

$$\psi_j\,(x_1, x_2, \ldots, x_n) = 0, \quad j=1,k \quad \ldots \quad (3)$$

In the case of structural problems, the function to be optimized is usually the weight of the structure and the constraint equations would be limitations on the geometry, stress, displacement and frequency.

Review Of Existing Methods

A number of methods have been developed to solve the constrained optimization problems. These include linear programming when both the functions to be minimized and the constraints are linear[26]. For a nonlinear optimization function and linear constraints the gradient projection method by Rosen[27] and the cutting plane method by Kelly[28] have been proposed. The largest body of material in the literature is concerned with gradient methods of minimization, where measurement of the slope of the function is used as an indication of the direction towards minimum[29,30]. Intuitively these methods have considerable appeal. But they are less satisfactory to work with, because of the computational effort involved in calculating the gradients at each step. Also the computational feasability of some of the methods referred to are not thorougly explored.

The direct search method is a sequential examination of trial solutions which are obtained by direct numerical functional evaluations. Each solution is compared with the best obtained up to that time, and there is a strategy for determining what the next trial solution will be. It is seen that the direct search procedure uses a numerical technique rather than an analytical one, and as such is very well adapted to any class of optimization problem. The repeated identical arithmetic operations with simple logic makes the problem extremely easy to solve by an electronic computer. Many direct search strategies have been proposed by a number of authors[31,11,32]. A detailed description of one direct search procedure is given in Appendix I.

Several very similar methods of successive linear approximation have been proposed. The earliest is called the method of approximate programming or MAP[12]. The linear programming algorithm is used repetitively in this differential technique, in such a manner that the solution of a linear problem converges to the solution of a nonlinear problem. Authors of this method claim that it has been very successful on nonlinear problems of fairly large orders. The procedure starts by linearizing the constraints and the objective function in the region about a known point, by expansion as a Taylor's series. The higher order terms other than linear ones are ignored. Then the problem is set up in such a way that it can be solved as a linear programming problem. The procedure is repeated with the new values of the variables as a starting point. A more detailed description is given in Appendix I. In practice, MAP is found to be a powerful but slow procedure.

Monte Carlo methods consist of formulating a game of chance which produces a random variable whose expected value is the solution of a certain problem. They can be applied to optimization problems involving a large number of factors where other methods could not be applied because the number of trials would be excessive. In this exploratory process, points are generated at random, certain points are picked over the whole range of variables, and the best result obtained is taken as optimum. One of these techniques, proposed by Dickinson[13], is described in Appendix I.

Constrained Optimization Problem

Although the constrained optimization problem has been handled by most of the methods discussed previously, the efficiency of each in handling constraints seems to vary widely. A constraint may be a condition on the individual parameters - like the lower and upper bounds on a design variable - or functions of the parameters. Before one writes the mathematical equations which leads to the computational algorithm, it is necessary to look into the requirements that the procedure is easy to program, guarantees convergence and presents no apparent problems with constraints. It is well to foresee the many alternate possibilities of expressing the problem. The inherent difficulty in handling the constraints is that they introduce discontinuities into the functions optimized. Some analysts have found it convenient to treat the constraints as absolute barriers. No bound variables or functions of them are allowed ever to transgress limits. The concept of the absolute barrier method is as follows. If a boundary violation occurs, the variables concerned with the violation are set back to their limit values so that the constraint is satisfied. This leads to the argument that many variables will reach their limits and stay there and the supposed solution may not be the optimum solution. However, when properly programmed and used by those who understand the nature of the particular problem at hand, this method has been demonstrated to work satisfactorily. It is also found that if the constraints are functions of the variables, this method becomes very difficult to program and time consuming to execute[36].

Among the other methods of handling constraints, the potentially applied restraint method seems promising. Corresponding to each constraint, there is a weighting factor W_j . If the function to be optimized is U, the variables are $x_i (i=1,n)$ and the constraints are $\psi_j = 0$ $(j=1,m)$ then, a pseudo optimum function U_o is created by [36]

$$U_o = U + \sum_{j=1}^{m} |\psi_j| \times W_j \qquad \ldots. \qquad (4)$$

If the weighting factors W_j are large, the constrainted items are indeed bound - not necessarily to the boundary limits; but the variables are artificially restrained from undergoing changes of significant magnitude. Boundary violations will occur if W_j is too small. The values of W_j will have to be reduced as the optimum point is approached.

It can be seen that the implementation of multivariable optimization problems with numerous constraints requires a reasonable degree of analysis before plunging into programming for a particular method of solution. There are no fast and sound rules which can apply with certainty to a new problem. It is essential that the influence of the various variables on the constraints and that of the constraints on the computation be first studied, before assigning all decisions to a mechanized programmed control. Typically four or five runs per job[36] on the computer may be necessary, for a human to review the progress and note whether or not an intervention is required - such as examining the boundary control procedure and making changes if it has not been satisfactorily executed. Thus in any meaningful constrained optimization problem, human intervention seems necessary.

AN ALTERNATE SEARCH TECHNIQUE FOR THE CONSTRAINED OPTIMIZATION PROBLEM

A generalized search technique, based on an alternation of a direct search code and linearization of the constraints and objective function, has been tried to provide a computationaly feasible method for the general constrained optimization problem discussed above. The direct search method of nonlinear optimization is quick and easy to formulate and operates very well when unconstrained. Although several versions of direct search have been proposed[31,11,32] all of them appear to have some difficulty when equality or inequality constraints have to be satisfied. The search usually gets hung up on the fences created by the constraints as shown in Figure 1. If we assume that the search has reached 'A' in a two variable problem, then the one-step-at-a-time nature of direct search will not permit us to move without violating the constraint and the search hangs up at that point. So it is necessary to turn the search parallel to the fence whenever one is encountered or constrain the search to a path if one must be followed. This is exactly what the alternate search technique, discussed below is programmed for. The program is written to treat difficult optimization problems involving as many as 20 variables, and has been successfully applied to several problems familiar in numerical analysis, formulated as peak finding problems. Among these problems were systems of linear equations, minimization of quadratic and cubic forms, and problems involving equality and inequality constraints. The results have been compared with those obtained by others.

Description Of The Search Technique

The alternate search technique consists essentially of two parts - a direct search in the region of the variables, including exploratory and pattern moves; and a linear approximation of the constraints and objective functions used when the direct search hangs up on a constraint or on a valley. The direct search itself consists of exploratory and pattern moves, which are explained in detail later. The exploratory moves indicate a direction, and the larger pattern moves allow us to move from point to point within the space. Whenever the set of exploratory and pattern moves fail to indicate a better result, the linear approximation routine replaces the direct search. The direct search may stall on a ridge (or valley) or on a constraint. The linear routine determines a direction along which the next pattern move in the direct search should be made. The direction is found by solving a linear programming problem with the linearized equations of the constraints and objective function at the point where the direct search has failed. In effect the exploratory search step of the direct search method is replaced by the linear approximation step. The direct search pattern move is carried on in the direction indicated by the linearized step. The entire cycle is repeated until two consecutive linear approximation routines indicate the same optimum value, which is taken as the optimum point. A general flow chart is given in Figure 2.

The optimization search process is initiated by picking up as a starting point an arbitary point - $x=(x_1, x_2, x_3, \ldots\ldots, x_k)$ inside the region of feasibility. We are assuming minimization is the optimization criterion.

It is assumed that the problem has k independent variables, n equality constraints and m inequality constraints. Symbolically the constraints may be written as

$$\phi_j \geq 0 \ (j=1,n) \ \text{Inequality constraints}$$
$$\psi_j = 0 \ (i=1,m) \ \text{Equality constraints}$$

Since the equality constraints must be satisfied precisely for the direct search an artificial objective function is created, by adding weighted functional values of the equality constraints to the true objective function $\bar{U}$. Thus,

$$\bar{U} = U + \sum_{i=1}^{m} |\psi_i| \times W_i \tag{5}$$

By doing so, a deviation of the optimization surface from the equality constraint paths is automatically penalized. Since the function U is to be minimized, the words success or failure have the following meaning when applied to the exploratory and pattern moves during the direct search. Beginning at some starting point, if the value of one or more independent variables during the search is changed, and if no inequality constraint violation occurs and if the value of the function at the new point is better than that at the starting point, then the move is called a success. Otherwise the move is a failure.

Direct Search Routine

The starting point in the space of feasible solutions is designated as the initial base point. To determine proper increments for each independent variable, an upper and lower value for each variable is intuitively assigned. If $R_i max$ and $R_i min$ are the estimated values for the upper and lower limits respectively of the ith variable, then the increment for that variable is defined as

$$\Delta x_i = \frac{R_i max - R_i min}{F} \qquad \ldots. \quad (6)$$

where F is an arbitrary number. The value of the objective function U is determined at the initial base point. An exploratory search is now started at this point. Each independent variable is incremented or decremented by the amount Δx calculated as above, while the remaining variables are kept constant. After each variable is changed, the inequality constraints are evaluated to check against constraint violation. If there is no violation, the objective function is evaluated at the new point. These are called exploratory moves and the new value of the variables is

$$x^1 = (x_1^1, x_2^1, x_3^1, \ldots., x_n^1)$$

The direction indicated by the exploratory move is an approximation to the negative gradient vector of the objective function;

and since this direction has indicated a better optimum value there is some reason for trying further in the same direction. A larger step size is now used, moving all the variables simultaneously in that direction. The pattern move and the sequence of steps for the pattern move is defined as follows

$$\delta x_i^{(J)} = H^J \delta x_i$$

where,

δx_i = $x_i^1 - x_i^o$ $(i=1,k)$

H = An acceleration factor greater than 1

J = An integer starting from zero and incremented by one for each successful pattern move.

After a pattern move the inequality constraints are checked for possible violations. Assuming there are none then the objective function is evaluated at the new point and this value is compared to the value at the initial base point. If the move is successful, the new point is designated as the base point. The value of J is increased by one which makes the step size larger, and another move is made in the same direction with this larger step size. Each time a successful pattern move is made a new base point is designated and the step size is increased by increasing J. Thus, once a pattern has been determined it is not revised, as long as the direction yields better values of the objective function and does not violate any constraints[11]. Once a large step (say when J=5) fails, the preceding base point (at which J=4) is kept and a new sequence is started from this point with J=0, in the same direction as the previous moves. This procedure is repeated until even the smallest step size (with J=0) fails. At this point a set of exploratory moves are made again to indicate a new direction for a pattern move. If a new direction can be found, pattern moves are made in this direction, in a manner described above. The procedure is explained clearly in Figure 3.

The search process thus consists of a series of pattern moves, of varying sizes. The computation for the series of pattern moves is far simpler than for the exploratory moves. Each set of exploratory moves may require up to 2k optimization function evalutions in the k variable problem, and with n constraints as many as 2nk constraint evaluations. The evaluation of the constraints may be the most time consuming portion of the search in a complicated problem. Keeping the number of exploratory searches to a minimum contributes to the efficiency of the search technique.

If, after a succession of successful pattern moves, a pattern move and the following exploratory move fails, then it could be expected that the region of search is in the neighbourhood of the optimum or hung up on a constraint or a valley. At this point, the linear approximation routine is effected.

Linear Approximation Routine

At the point where the direct search technique has failed, a new direction vector is found for the pattern move by solving the linear programming problem after linearizing the objective function and the constraints. The approach is identical to the method of approximate programming proposed by Griffith and Stewart[12] and is also explained in Appendix I. The linearization is carried out by expanding the functions about the base point by Taylor's series, and neglecting the terms above linear. By suitably assigning limits to the change in the variables, the linear programming results are prevented from making the linearization hopelessely invalid. Once a direction has been found by this routine, the direct search routine comes into effect and pattern moves are carried out in this direction just as before, after which the sequence of exploratory and pattern moves is again used. When the linear approximation routine fails to indicate a direction in which to continue the search, the search halts and it is assumed that the optimum value is reached.

Description of the General Code

The optimization technique described above has been programmed in FORTRAN IV of the IBM/7040 computer. The programme has the following principal parts.

I Main Programme

II Sub-routines

a. Subroutine MATRIX
b. Subroutine ORDER
c. Subroutine SIMPLE
d. Subroutine SEARCH
e. Subroutine REALU
f. Subroutine CONST
g. Subroutine ENEQ

The following variables and parameters are read in by the main programme.

NPROB	-	Number of problems that are to be solved in one run. The programme is designed for a maximum number of five problems to be solved in one run.
K	-	Number of real variables.
NUM,NUMR	-	Number of equality constraints.
NEQ	-	Number of inequality constraints.
NMAX	-	Maximum allowed number of iterations in a simplex cycle.
INDEXI	-	Indicatpr for phase I or II of the simplex cycle.
P,H	-	Acceleration constant in direct search, It has a value greater than 1.
F	-	Arbitrary constant used in finding search increment.

G	- Arbitrary (small) constant representing the smallest increment size in direct search.
TES	- Criterion for optimum

The following variable names represent the parameters used in the search code.

IMM	- 2K - Number of equations, representing constraints for the upper and lower limits for the linearized step size in successive linear approximation. This value is stored in memory of the computer, for repeated use.
IM	- IMM + NEQ
M	- Total number of constraints for the simplex solution.
Z(I),X(I)	- Variable name.
BB(I),B(I)	- Controlled step size for the variables during simplex operation.
STEPX(I)	- A small increment for each variable.
RMAX(I)	- Approximate maximum value a variable is likely to have.
RMIN(I)	- Approximate minimum value that a variable might have.
III(I),II(I)	- Subscript of the variables in basis.
WATE(I)	- Weighting factor for each equality constraint.
NCYCLE	- Counter for the number of simplex and direct search cycle.
UR	- Real optimum value.
U	- Pseudo optimum value.

A(I,J)	- Simplex Matrix.
C,S	- Co-efficients in the objective function.
DELX(I)	- Increments during direct search.
PSI	- Equality constraint function.
PHI	- Inequality constraint function.

Starting from an initial feasible solution, the procedure of optimization goes through the following subroutines.

Subroutine SEARCH

The smallest increment TEST(I) for each basic variable is generated by,

$$TEST(I) = \frac{RMAX(I) - RMIN(I)}{F} \quad \ldots (7)$$

Each variable is incremented or decremented by an amount TEST(I) and the best direction for a move is found. The best direction means that the inequality constraints are not violated and that it is likely that a better optimum can be obtained in that direction. Once a direction is established, the pattern move starts. The increments are given as follows.

$$X(I) = X(I) \pm DELX(I) * H^J \quad \ldots (8)$$

During each step, the constraints are checked for a violation by calling subroutine ENEQ, which evaluates the constraints for each X(I).

Once the direct search is not able to show any further improvements, the routine switches to the linear approximation procedure.

Subroutine MATRIX

This routine sets up the simplex matrix from the linearized quantities. At the point where the direct search has stopped, the constraints and the objective function are evaluated by calling the subroutine ENEQ, CONST and REALU. The values are represented by ϕ_j^o, ψ_j^o, and U^o. Now a small increment STEPX(I) is given to each variable one at a time, and the new values for the equality, inequality and objective functions ψ_j^1, ϕ_j^1 and U^1 are calculated at the new point. The partial derivatives of the equations are evaluated by the simplest numerical approximation given in Appendix I, and the entire matrix is set up with the slack variables included. To check against any of the B(I)s becoming negative, subroutine ORDER is called, to arrange the equations properly and include artifical variables if necessary.

Subroutine ORDER

This subroutine is called by the routine MATRIX, checks against any of the B(I)s becoming negative. If any B(I) becomes negative it means that a constraint is being violated.

This subroutine arranges the violated inequality constraint in such a manner that the violated constraints are included in Phase I of the simplex program.

Subroutine SIMPLE

This is a standard simplex routine with Phase I and Phase II. This subroutine is used only to find a new direction to proceed with, when the optimization procedure hangs up on a constraint or a valley.

Subroutine REALU

This routine calculates the true optimum value UR and the pseudo optimum value U, from the objective function. The relation between U and UR is given by

$$U = UR + |PSI(I)| * WATE(I) \qquad \ldots \quad (9)$$

Subroutine CONST

The expressions for the equality constraints are given in this routine and their values are computed whenever it is called. The expressions are denoted by the variable name PSI(I).

Subroutine ENEQ

This subroutine evaluates the inequality constraints.

The values are represented by PHI(I), (I=1,NEQ) and the subroutine is called by routines MATRIX, SEARCH and also the main programme.

Application Of The Programme

The programme has been successfully applied to solve several problems commonly used by workers in this field, as an instructive exercise and necessary testing phase in the development of the technique. The results are compared below, for six problems.

Glass And Cooper's Problem

Minimize $U = -\left[25.0 - (x_1-5)^2-(x_2-5)^2\right]^{1/2}$

Subject to $x_1^2 - 4x_2 \geqslant 0$

$$(x_2-6)^2 - 4(x_1-3) \geqslant 0$$

Since it is obvious that the optimum point will be at the intersection of the two constraints, both the constraints were treated as equality constraints.

	Author's method	Our method
Starting point :	$x_1=7,\ x_2=1$	$x_1=7,\ x_2=1$
Optimum point :	$x_1=4.0,\ x_2=3.999$	$x_1 = 3.99998$ $x_2 = 3.99985$
Optimum value :	$U = -4.796$	$U = -4.79580$
True Optimum point :	$x_1=4,\ x_2=4$	

J.E. Kelley's Problem

Minimize $U = x_1 - x_2$

Subject to $3x_1^2 - 2x_1x_2 + x_2^2 - 1 = 0$

	Author's method	Our method
Starting point :	$x_1 = 2, \ x_2 = 2$	$x_1 = 2, x_2 = 2$
Optimum point :	$x_1 = -0.073, x_2 = .929$	$x_1 = 0, x_2 = 1.0$
Optimum value :	$U = -1$	$U = -1$
True optimum point :	$x_1 = 0, \ x_2 = 1.0$	

Dickinson's Problem

Minimize $U = 1 + f_1 f_2$

Where, $f_1 = 11 - 6x_1 - 4x_2 + x_1^2 + 2x_2^2$

$f_2 = 17 - 8x_1 - 6x_2 + 2x_1^2 + x_2^2$

$x_1 \geqslant 0, \ x_2 \geqslant 0$

Starting point :	$x_1 = 5, x_2 = 2.0$	$x_1 = 5, x_2 = 2.0$
Optimum point :	$x_1 = 3.085, x_2 = .942$	$x_1 = 3.00019$ $x_2 = 0.99999$
Optimum value :	1.115	1.0
True optimum :	1.0, $x_1 = 3, \ x_2 = 1.0$	

Rosenbrook Problem

Minimize $U = 100(x_2 - x_1^2)^2 + (1-x_1)^2$

$$x_1 \geqslant 0, \; x_2 \geqslant 0$$

	Author's method	Our method
Starting point :	$x_1 = 0, \; x_2 = 0$	$x_1 = 0, \; x_2 = 0$
Optimum point :	$x_1 = 1.00100$ $x_2 = 1.0020$	$x_1 = 0.99946$ $x_2 = 0.99902$
Optimum value :	0	0

True optimum point : $x_1 = 1.0$ $x_2 = 1.0$

Quadratic Programming Problem By Leon

Minimize $U = 183-44x_1-42x_2+8x_1^2-12x_1x_2+17x_2^2$

Subject to $2x_1 + x_2 = 10$

$$x_1 \geqslant 0, \; x_2 \geqslant 0$$

	Author's method	Our method
Starting point:	$x_1 = 0, \; x_2 = 0$	$x_1 = 0, \; x_2 = 0$
Optimum point :	$x_1 = 3.69, \; x_2 = 2.304$	$x_1 = 3.802, \; x_2 = 2.395$
Optimum value :	20.868	19.002

True Optimum result: $x_1 = 3.8, \quad x_2 = 2.4$

$$U = 19.0$$

Fiacco And Mccormic Problem

Minimize $U = x_1^3 - 6x_1^2 + 11x_1 + x_3$

Subject to $x_1^2 + x_2^2 + x_3^2 = 4$

$x_1^2 + x_2^2 \leqslant x_3^2$

$x_3 \leqslant 5$

	Author's method	Our method
Starting point :	$x_1=0,\ x_2=0,\ x_3=0$	$x_1=0,\ x_2=0,\ x_3=0$
Optimum point :	$x_1=0,\ x_2=1.4142$ $x_3=1.4142$	$x_1=0.0,\ x_2=1.4142$ $x_3=1.41423$
	$U = 1.4142$	$U = 1.41423$

True optimum result: $x_1=0, \quad x_2=\sqrt{2}, \quad x_3=\sqrt{2}$

$$U = \sqrt{2}$$

The computer programme for the alternate search routine is given in Appendix 2.

STRUCTURAL OPTIMIZATION WITH VIBRATIONAL AND STRESS CONSTRAINTS

The optimization criterion in the optimum design of structures is usually minimum cost or weight. The minimum weight design is an arrangement of the structural elements where all the design requirements such as stresses, deflections and geometrical constraints are satisfied and the total weight of the structure is minimized. The procedure could be set up as a mathematical programming problem i.e. to determine $A_1, A_2, A_3, \ldots A_m$ in such a manner so as to minimize

$$U = \sum_{i=1}^{m} \rho_i L_i A_i \qquad \ldots\ldots (10)$$

A_1, A_2, ... A_m are the areas of cross section of the m members of the structure, ρ_i their densities and L_i their lengths. U represents the weight of the structure, subject to geometrical constraints, strength constraints, stability constraints, displacement constraints, frequency constraints and dynamic stress and displacement constraints. A rather limited amount of work seems to have been done on the general problem of optimization of spatial structures with vibrational constraints.

Probable Techniques For Structural Optimization

Very many techniques of optimization analysis appear in the literature and a brief reference to many of them has been made previously while discussing the entire project. It will be appropriate to review some of the unique techniques used in structural optimization at this stage.

Schmit and Mallet[19] used an unique approach to formulate the constraints while optimizing a simple pin jointed three bar planar truss. The parameters that identify the design were the material density for each member of the truss, the cross sectional area, and the angle representing the geometric layout of the truss. These parameters were designated by the quantities ρ_m, A_m and β_m respectively. The distinctively unique feature of this work was the treatment of the design parameter defining the material i.e. ρ_m as a continuously varying quantity. This is achieved by using the interpolated materials concept which contends that there exists a continuous spectrum of materials between existing materials. It is assumed herein that the pertinent mechanical properties may be expressed as continuous functions of weight density. The modulus of elasticity E, the yield stress σ_y and the coefficient of thermal expansion α of a representative class of structural alloys were plotted versus density and then curve fitted. The load conditions were taken to be independent of the design parameters and the optimization routine was designed to include five distinct load conditions.

It was stipulated that the tension yield stress was not to be exceeded in any load condition, that the compressive yield stress was not be exceeded and that the x and y displacement components of the nodal point where the three members join together were not to exceed specific limits. The stresses in the three truss members and the displacement components for the different loading conditions are designated as behaviour variables. The relation between the behaviour variables, the design parameters and the applied loads are readily formulated as

$$\left[C_{ij}\right] \left[B_{kj}\right] = \left[A_{ik}\right]$$

where C_{ij} is the configuration matrix, the elements of which contain the design parameters and constants; B_{kj} is the behaviour matrix, the elements of which represent the stresses in the members and displacement of the nodes; and A_{ik} is the applied load matrix, the elements of which represent the loads at the node in x and y directions and also the temperature variation in each member.

The following bounding constraints are imposed on the design parameters.

a. An upper and lower limit on ρ_m to avoid other than structural metals.

b. A lower limit on A_m to avoid negative areas.

c. Upper and lower limits on β_m to avoid a trivial multiplicity of design points for a single configuration.

The optimization technique was essentially a gradient method. On reaching , a constraint, the search moves along a constant weight plane and again the gradient method is used. The method is repeated until an optimum weight is obtained.

It could be easily seen that the optimization of the planar frame with pin jointed members does not have a very complicated form of constraining equations. The stresses in the members are uniform throughout their lengths. The size of the matrix for the behaviour variables is entirely dependent upon the number of loading conditions considered. The constraint evaluation is not difficult because of these reasons.

Razini[24] optimized a structure under multiple loading conditions on the basis of each member being fully stressed in at least one loading condition. If the analysis showed that a certain member is overstressed in a critical load condition, the design method increased the area of that member sufficiently to remove the overstress. It does the opposite if the member is understressed. The method of fully stressed design is thus an iterative process that usually converges to a final design. Starting with an initial design where the area of its members is given a matrix A^o , the forces in each member for each loading condition is found by using the combined method of analysis which leads to the following matrix form,

$$CY = P$$

where Y is the behaviour matrix containing the member forces and displacements. C is the configuration matrix and P is the applied load matrix. Matrix Y is found by inverting matrix C.

$$Y = C^{-1}P$$

The area of the ith member A_i^o, because of the redesign becomes A_i^1, or,

$$A_i^1 = \frac{F_i^o}{\sigma_i}$$

where, F_i^o is the ith member force for a particular loading condition and σ_i is the allowable stress for the member during that loading condition. The difference between the area of the successive designs is

$$\Delta A^o = A^1 - A^o$$

Because of this change in the member areas, the configuration matrix C and hence the behaviour matrix Y will be changed and a new set of critical areas are found by reanalysis or by using an equivalent force method[38]. This process of iteration converges within a reasonable number of cycles. The inversion of the matrix C during every iteration is eliminated in the equivalent force method.

Schmit and Fox[23] reported results of a new approach to structural synthesis which made it possible to find an optimum design without engaging in the evolution of a large number of intermediate trial designs. The problem was formulated so that optimum designs can be found by the application of any technique for seeking the unconstrained minimum of a function of many variables. For example, suppose that the equations representing the analysis of a structural system are

$$\begin{aligned} a_{11}\ \sigma_1 + a_{12}\ \sigma_2 &= C_1P_1 \\ a_{21}\ \sigma_1 + a_{22}\ \sigma_2 &= C_2P_1 \end{aligned} \qquad \ldots\ldots \quad (11)$$

and it desired to find σ_1 and σ_2 where a_{ij}, C_1 and P_1 are known. The value of σ_1 and σ_2 that satisfy equation (11) are those that make

$$\theta_1(\sigma_1, \sigma_2) = (a_{11}\ \sigma_1 + a_{12}\ \sigma_2 - C_1P_1)^2 + (a_{21}\ \sigma_1 + a_{22}\ \sigma_2 - C_2P_1)^2 = 0\,.$$

Therefore the problem reduces to the minimization of $\theta_1(\sigma_j)$ or to the finding of that stress σ_1 and σ_2 for which $\theta_1(\sigma_1, \sigma_2) = 0$. If a_{ij} are assumed to be design variables and θ_2 is a function of θ_1 but with the a_{ij} variable, then the problem could be modified such that $\theta_2(\sigma_1, \sigma_2, a_{ij}) \to 0$. Any solution of this problem will be a design, but it is obvious that the number of sets of values of a_{ij}, σ_j for which $\theta_2 = 0$ is infinite. If there are inequality constraints on the acceptable values of a_{ij} and the σ_j, penalty terms are added to $\theta_2(\sigma_j, a_{ij})$ for violating these constraints as follows.

$$\Theta_3(\sigma_j, a_{ij}) = \Theta_2 + \sum_{j=1}^{2}\left[(\sigma_j - LB_j)^2 + (UB_j - \sigma_j)^2 + \sum_{i=1}^{2}(a_{ij} - LB_{ij})^2 + (UB_{ij} - a_{ij})^2\right]$$

UB and LB are the upper and lower bounds respectively on a given variable. The weight of the structure is expressed as a function of the design variables and an additional constraint is incorporated such that

$$\psi = (W - W_o)^2 + \Theta_3$$

where W_o the draw-down weight, is a goal weight for a particular draw-down cycle. It is required that the weight of the structure should be less than the draw-down weight. The function is minimized using a gradient method of minimization. It is repeated using progressively lower draw down weights until the function cannot be made zero. At this point the value of W is taken as the optimum weight of the structure. Using this approach Schmit et al optimized a 3 bar pin jointed planar frame. This procedure obviously requires that the related functional expressions should be in the form of explicit equations of the variables. This method does not seem to handle constraints other than the simple types shown.

Formulation of Constraints for Structural Model

For the first stage of the project, Tiwari[39] and Raghava[40] examined a simple four bar frame without symmetry and with fixed member ends illustrated in Figure 4. The four bars are welded at the base to a half inch thick aluminium alloy plate at a 24" square spacing. The top ends of the bar are brought close together and welded to another half an inch thick aluminium plate at a square spacing of 2½". The formulation of the stress and frequency optimization constraints are obtained by the static and dynamic analysis of the model structure shown in Figure 4. It was decided to treat only the member areas as the variable from the optimization point of view, while the configuration is not changed. The constraint equations are given below as obtained from the analysis[39].

Stress Constraint

The theoretical analysis of the structure was done using the finite element - matrix approach[4]. For the optimization analysis, a relationship between the external loads and the resultant forces on the member ends was required. The external loads were to act at the apex of the structure. Hence only one node at the top plate was needed. The structure was therefore idealised into four flexible members integral with a short rigid element contributed by the top plate. The stresses S_i in the ith member are obtained by transforming the forces P_i at the free end of each element by a general transformation matrix R. Let S_i be a column vector of axial stress S_{xi} and transverse shear stresses S_{xyi} and S_{xzi}, where x,y,z refer to member co-ordinates. Then it is shown[39] that

$$\left\{ S_i \right\} = \left[R \right] \left[k_{22}^{i} \right] \left[T_i \right] \left[H_i^T \right] \left[K^{-1} \right] \left\{ F \right\} \quad \ldots (12)$$

where

$\left\{S_i\right\}$ = 3 x 1 matrix giving the three stresses in the ith leg of the 4 bar structural model

$\left[R\right]$ = 3 x 6 matrix relating the stress and loads in which all elements are known

$\left[k_{22}^i\right]$ = 6 x 6 element stiffness sub matrix in member co-ordinates in which all the elements are known

$\left[T_i\right]$ = 6 x 6 transformation matrix to system co-ordinates with all elements known

$\left[H\right]$ = 6 x 6 equilibrium matrix

$\left[K^{-1}\right]$ = 6 x 6 structural assembly matrix in which the elements are functions of unknown areas and the matrix is to be inverted each time before multiplying with other matrices

$\left\{F\right\}$ = 6 x 1 applied external load matrix

The exact nature of the terms in matrix R will depend on the member cross-section. The elements of this matrix for a hollow circular cross-section is given elsewhere.

Frequency Constraint

The dynamic analysis of the model structure gives the equation for the natural frequency of the structure[40] as,

$$-\omega^2 \left[m\right] \left\{x\right\} + \left[k\right] \left\{x\right\} = 0 \quad \ldots (13)$$

where,

$\left[m\right]$ = 54 x 54 mass matrix involving unknown member areas

$\left[k\right]$ = 54 x 54 structural stiffness matrix involving unknown member areas

$\left\{x\right\}$ = 54 x 1 eigenvectors.

The sizes of the matrices depends on the number of finite elements used, in this case three per leg. The limits for the cross sectional area and displacements could be treated as the other constraints of the problem. The objective function will be to minimize the weight of the structure and will include the lengths and cross sectional areas of the members.

Optimization of the Structural Model

The equations for the stresses and the natural frequency of the structural model, respectively shown in the form of equations (12) and (13), reveal their complex nature. These high order matrices have to be multiplied each time constraints must be evaluated. It is also necessary that the maximum stress in each member must be limited to a safe value for a particular loading condition. The exact nature of the elements in the transformation matrix R will depend on the member cross section. For hollow circular cross section it is given by,

$$\begin{bmatrix} \frac{1}{A} & \frac{-1\, R_o \operatorname{Cos}\theta}{I_z} & \frac{-1\, R_o \operatorname{Sin}\theta}{I_y} & 0 & \frac{R_o \operatorname{Sin}\theta}{I_y} & \frac{-R_o \operatorname{Cos}\theta}{I_z} \\ 0 & \frac{-MA_z \operatorname{Sin}\theta}{I_z t} & \frac{MA_y \operatorname{Sin}\theta}{I_y t} & \frac{-R_o \operatorname{Sin}\theta}{I_x} & 0 & 0 \\ 0 & \frac{MA_z \operatorname{Cos}\theta}{I_z t} & \frac{-MA_y \operatorname{Cos}\theta}{I_y t} & \frac{R_o \operatorname{Cos}\theta}{I_x} & 0 & 0 \end{bmatrix}$$

where the significance and the sign conventions for the elements in $\left[R\right]$ are given in Figures 5 and 6. $\left[R\right]$ defines the stresses only for previously selected locations along the length and across the cross section of each member. As such there is no way of predicting directly from these stress constraints, where the maximum stress occurs in a member of the structure. It seems necessary that for a particular loading condition the stresses in a number of cross sections along the length of a member must be analyzed before finding the maximum stress in that member, which is to be limited to a safe value during the optimization procedure. Moreover, the stress equation has a 6 x 6 matrix to be inverted each time the stresses are evaluated. The equation for the natural frequency of the model structure also does not seem less complex, in terms of the computing time necessary for the evaluation of the frequency.

It might be worthwhile to view the optimization of the structural model in light of the unique techniques found in the literature and which have already been discussed. In the planar pin jointed three bar truss problem of Schmit and Mallet[19] the stress and displacement constraints are of a simple form, and the size of matrices involved are of 5 x 5 or less. The stresses were uniform along the length in each member and hence the constraint evaluation did not pose much of a problem. There were no equality constraints which are to be satisfied precisely. We can generally assume that the number of constraint evaluations necessary before an optimum weight is reached to be the same in the gradient method of Schmit as it will be in the alternate search routine developed for this project. However, the gradient evaluation is more time consuming. From these observations, it is believed that the structural synthesis method of Schmit et al, will have the same difficulty in handling the optimization of our structural model.

In a second approach[23] Schmit et al used a technique based on the method of solving linear simultaneous equations by minimizing the sum of the square of the residuals to zero. It is easily seen that this method of solution requires that the constraints must be in explicit equation form. Hence the applicability of this approach to optimize the structural model, where we have the constraints in implicit complex matrix form is ruled out.

Razini's[24] method of fully stressed design is analysis-oriented. There is no apparent relationship between his concept and a minimum weight optimum design which can be set up as a mathematical programming problem. His analytical iterative approach does not make use of any conventional procedure of mathematical programming, as it was not the objective of his paper. However, he observes that for structures with a large number of members, the dimensionality of the problem becomes so large that the application of existing optimization methods becomes impractical. Under such circumstances, the iterative fully stressed design is helpful.

DISCUSSION OF RESULTS AND CONCLUSIONS

Alternate Search Routine

The alternate search routine for optimization of nonlinear problems has been developed and programmed for the IBM 7040 Computer. The method has been successfully applied to solve several problems familiar in optimization analysis. Among these problems were systems of linear equations and the minimization of quadratic and cubic forms, involving equality and inequality constraints. The program has been designed to treat problems involving as many as 20 variables. The results obtained have been compared with those obtained by others.

The general behaviour of the method in handling inequality constraints is of interest. If we are on an inequality constraint just after a linear approximation routine, the subsequent pattern move in the direction indicated by the linear approximation routine will inevitably go off feasibility quite badly. This is avoided by making the pattern move a constrained one. In other words, the variables are always checked against the violation of constraints at the time during the pattern or exploratory moves. If a violation occurs, the move causing the violation is abandoned, and a new direction for the move is determined either by an exploratory search, or by a linear approximate routine when the former fails. This approach is somewhat unsatisfactory, since it may degenerate into a pure linear successive approximation technique. However, the analysis of the results of the several simple problems indicates an initial and rapid movement by a series of exploratory and pattern moves until the first inequality constraint is reached, and then a series of successive linear steps is effected. But there is usually again a period of direct search pattern moves, ending at the optimum with two successive linear approximations. This is somewhat similar to the approach of Glass and Cooper[35].

Optimization Of The Structural Model

The extremely complex nature of the constraining equations makes the optimization of the structural model very difficult. The multiplication of the high order matrices each time a constraint must be evaluated is a very time consuming operation even for electronic computers. The predetermination of the location of maximum stress in a member is not possible, and this necessitates the evaluation of the stresses on a number of cross sections of the member for a particular loading condition. Obviously the whole matter is an enormously time consuming operation, considering the number of times the constraints have to be evaluated before the optimum point is reached. Although there is no way of predicting the number of steps of solution between the initial starting value and the final optimum in an optimization problem, experience of the previous workers could be taken as a guide. Schmit[23] assumed 2000 cycles as the criterion to abandon the optimization process of his three bar truss if there was no convergence. This value seems some what large, but might be necessary if the initial starting value is poor. Although no data is available on the computation time for the evaluation of the stress constraint, it is found that it takes at least 17 minutes of the IBM 7040 Computer time for one evaluation of the frequencies from equation 13 The magnitude of the problem could be realized if it is necessary to compute the stresses and frequencies more than about 100 times. The enormity of the problem cannot be lessened to any great extent, even after making allowance for such defects as inefficient programming. Hence no attempt was made to employ the alternate search technique to optimize the structural model.

It is necessary to point out here that the structural model is relatively a small one, and most real spacial structures will be much more complex. Optimization of such structures will be very difficult unless there is some way of eliminating the complexities associated with the static and dynamic constraints.

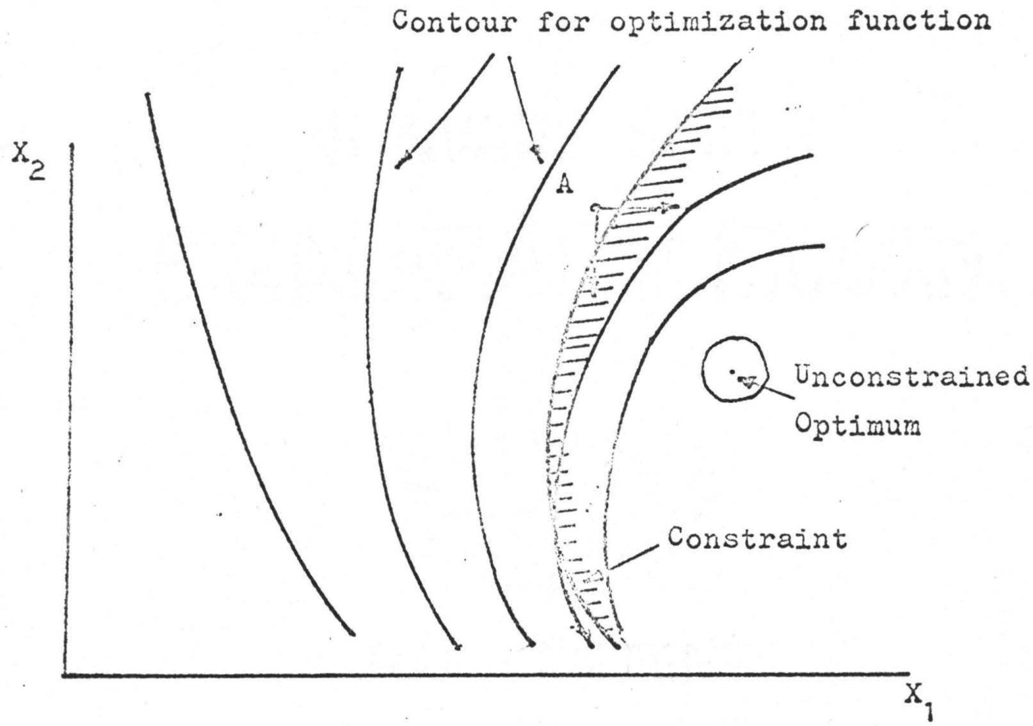

FIGURE 1 DIRECT SEARCH HANGS UP

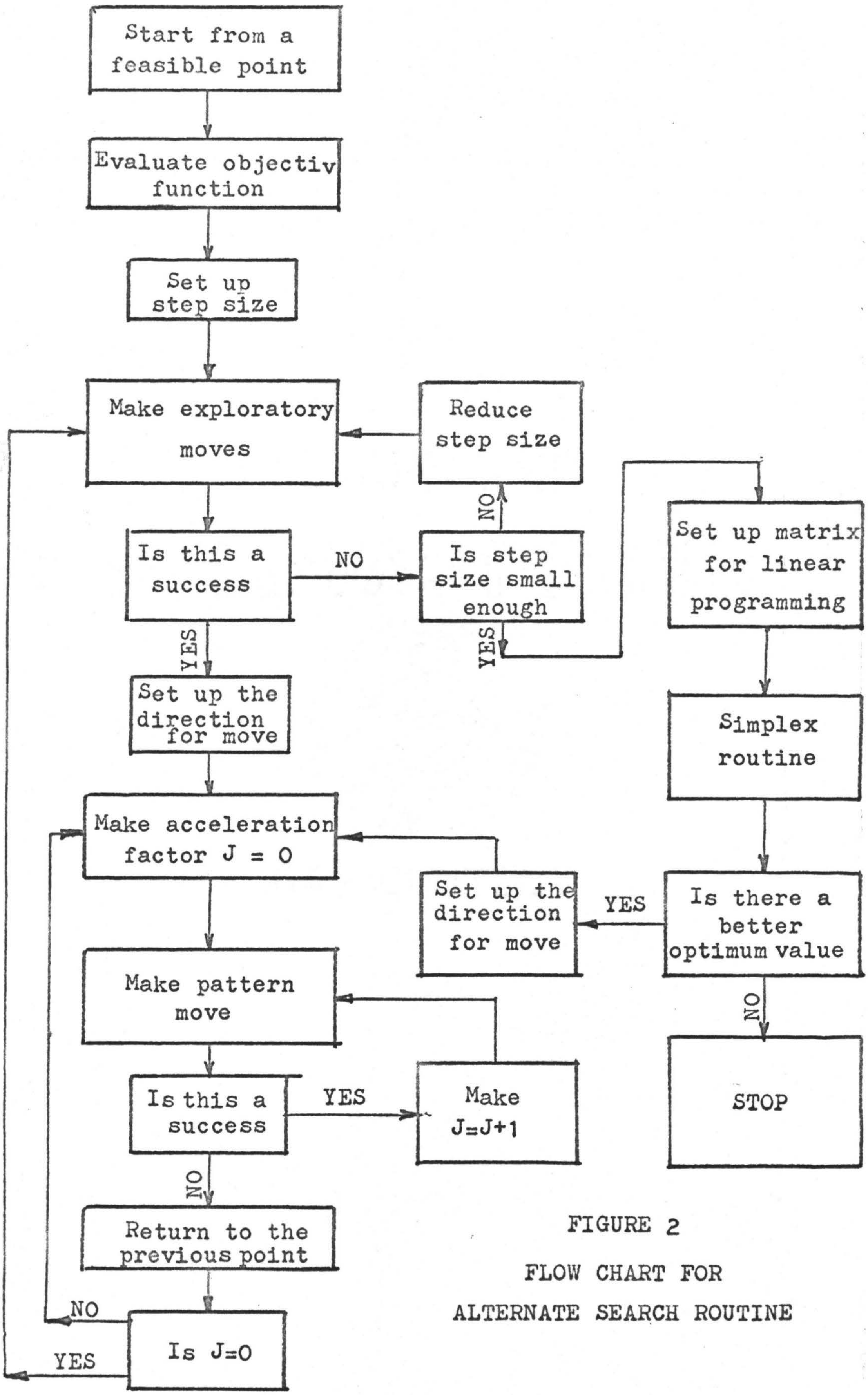

FIGURE 2

FLOW CHART FOR

ALTERNATE SEARCH ROUTINE

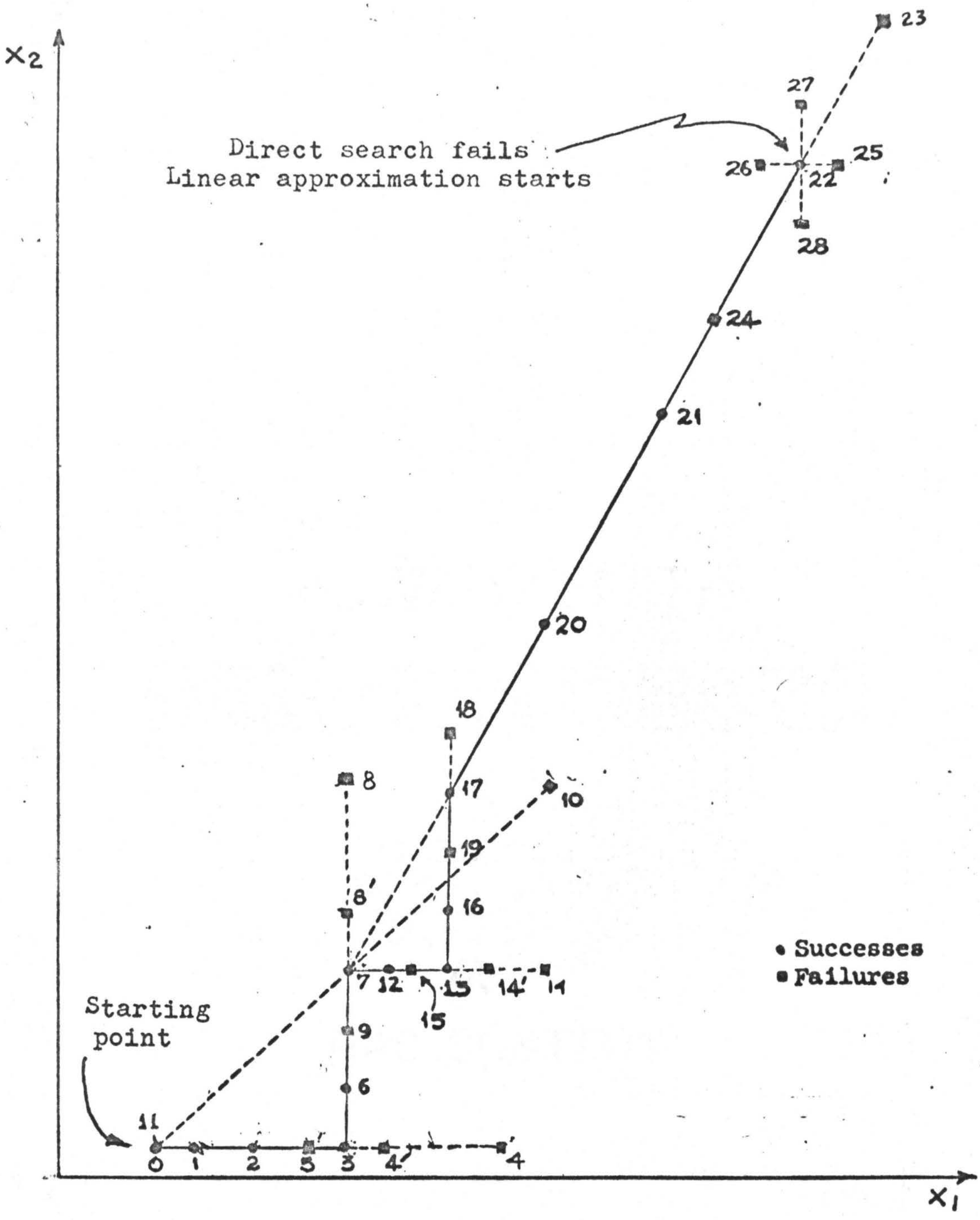

FIGURE 3 EXPLORATORY AND PATTERN MOVES

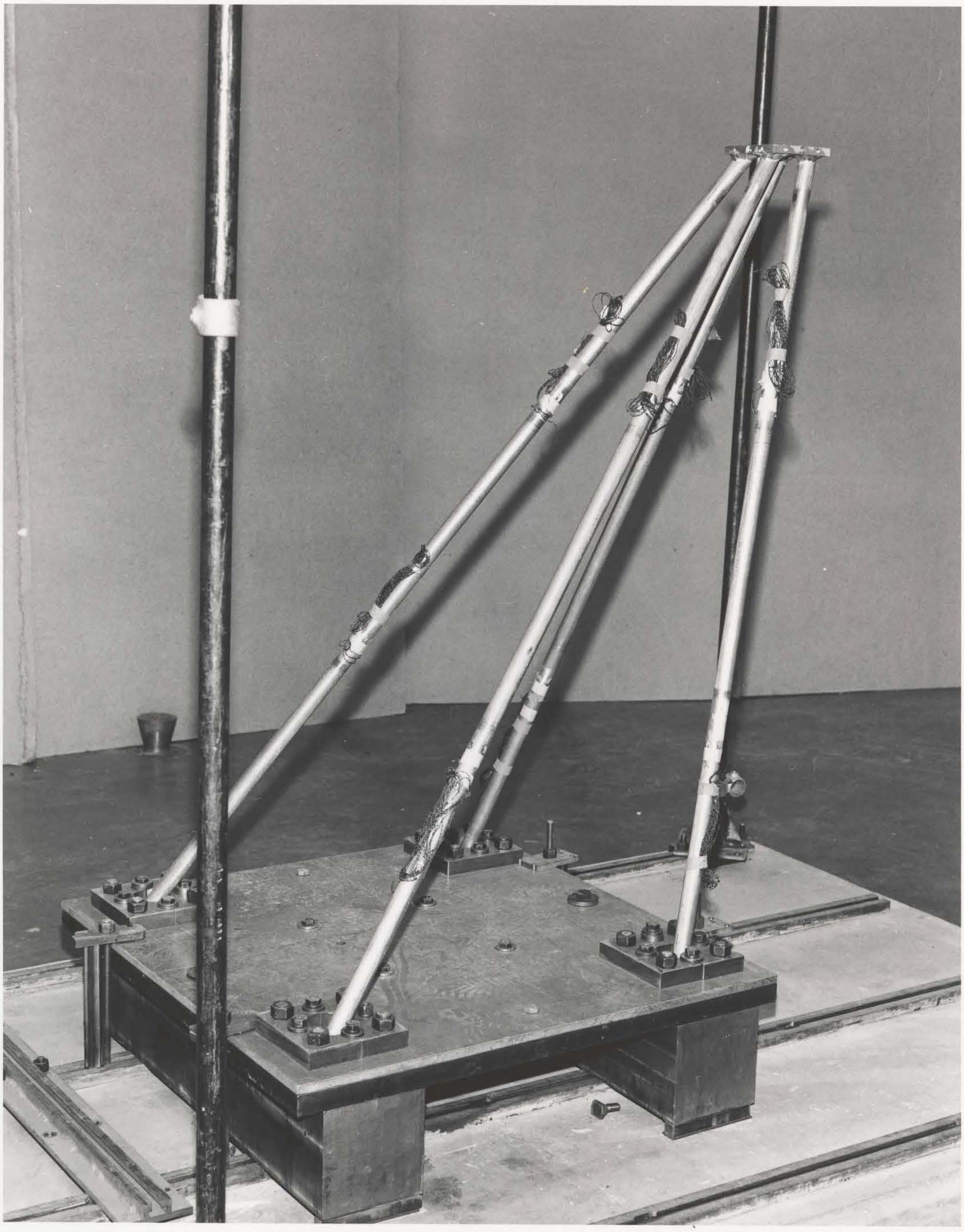

FIGURE 4 . STRUCTURAL MODEL

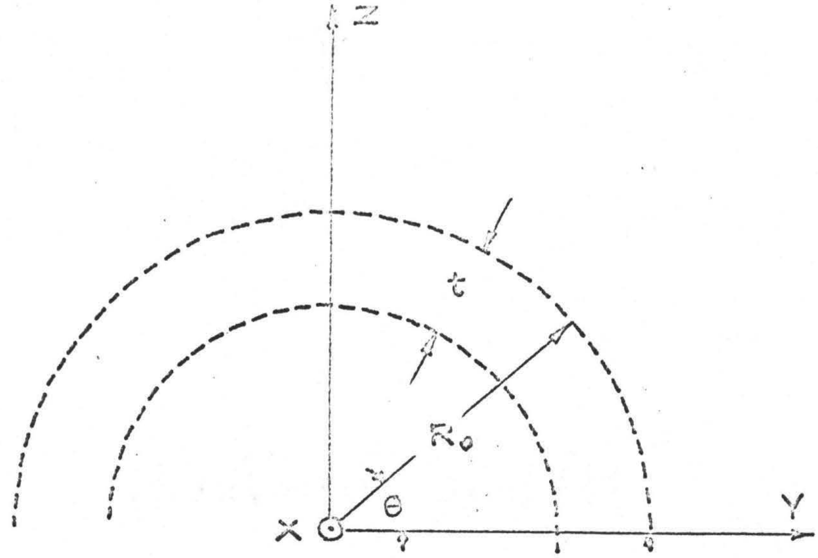

FIGURE 5.
TERMS IN MATRIX R

XYZ	...	MEMBER AXES
t	...	THICKNESS OF THE HOLLOW CYLINDRICAL ELEMENT
R_o	...	OUTER RADIUS
θ	...	ANGLE BETWEEN THE POSITION VECTOR OF ANY POINT AND Y AXIS
MA_Y	...	MOMENT OF AREA BETWEEN THE RADIUS VECTOR AND AXIS Y ABOUT THE Z AXIS

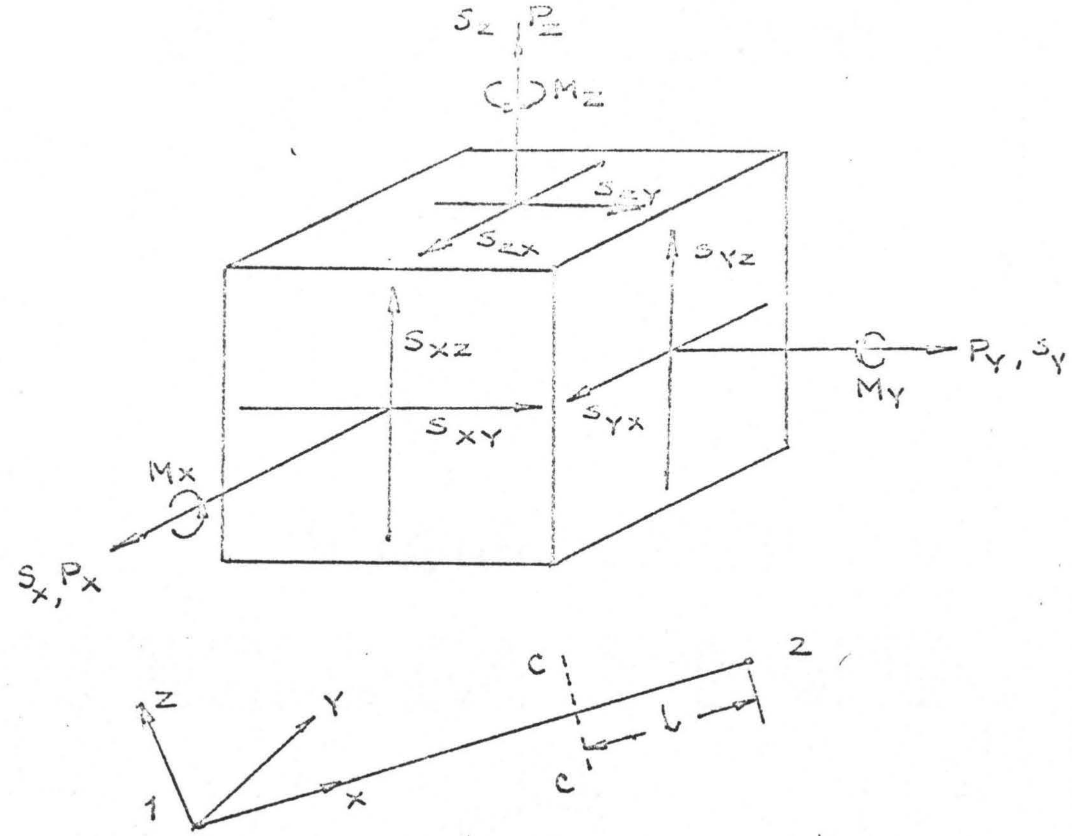

FIGURE 6 SIGN CONVENTIONS

P_x ETC	...	FORCES AT MEMBER END 2 IN X ETC DIRECTION
M_x ETC	...	CORRESPONDING MOMENTS AT MEMBER END 2
S_x ETC	...	DIRECT STRESSES ALONG COORDINATE AXES
S_{xY} ETC	...	TRANSVERSE STRESSES
XYZ	...	MEMBER COORDINATE AXES
1 2	...	MEMBER AXIS

REFERENCES

1. Przemieniecki, J.S. : "Matrix Structural Analysis of Substructures", AIAA Jour., Vol.1, No.1, Jan. 1963 pp. 138-147.

2. Argyris, J.H., S. Kelsy : Modern Fuselage Analysis and Elastic Aircraft, Butterworths, 1963.

3. Turner, M.J., R.W. Clough, H. C. Martin, L. J. Topp, "Stiffness and Deflection Analysis of Complex Structures", Jour. of Aeron. Sci., Vol. 23, No.9, Sept. 1956, pp. 805-824

4. Clough, R. W. : "Finite Element Method in Structural Mechanics", Stress Analysis, Wiley, 1966, pp. 85-119.

5. Zienkiewicz, O. C. : "Finite Element Procedures in the Solution of Plate and Shell Problems", Stress Analysis, Wiley, 1966, pp. 120-144.

6. Percy, J.H., T. H. H. Pian, S. Klein, D. R. Navratna : "Application of Matrix Displacement Method to Linear Elastic Analysis of Shells of Revolution", AIAA Jour., Vol. 3, No. 11, Nov. 1965, pp. 2138-2145.

7. Argyris, J. H. : "Matrix Analysis of Three Dimensional Elastic Media", AIAA Jour., Vol. 3, No. 1, Jan. 1965, pp 45-51.

8. Archer, J. S. : "Consistent Matrix Formulation for Structural Analysis Using Finite Element Techniques", AIAA Jour. Vol. 3, No. 10, Oct. 1965, pp. 161-178.

9. Archer, J. S. : "Consistent Mass Matrix for Distributed Mass Systems", Jour. Str. Div., Proc. ASCE, Vol. 89, St. 4, Pt. I, August 1963, pp. 161-178.

10. Hurty, W. C. : "Dynamic Analysis of Structural Systems Using Component Modes", AIAA Jour., Vol. 3, No. 4, April 1965, pp. 678.

11. Flood, M. M. and A. Leon: "A Generalised Direct Search Code for Optimization", Mental Health Research Institute, University of Michigan, Preprints 109, and 129, May 1963 and June 1964.

12. Griffith, R. E. and R. A. Stewart,: "A Nonlinear Programming Technique for the Optimization of Continuous Processing Systems", Management Science, Vol. 7, 1961, pp. 379.

13. Gallagher, P. J.: "Mop-1, An Optimizing Routine for the I.B.M. 650", Can. G. E. Civilian Atomic Power Dept. Report No. R60CAP35, 1960.

14. Young, J. W. and H. N. Christiansen: "Synthesis of Space Truss Based on Dynamic Criteria", Jour. Str. Div., Proc. ASCE, ST. 6, Dec. 1966, pp. 425-441.

15. Sved, G.: "The Minimum Weight of Certain Redundant Structures", Austr. Jour. of App. Sci., Vol. 5, 1954, pp. 1-9.

16. Schmidt, L. C.: "Fully stressed Design of Elastic Redundant Trusses Under Alternate Load Systems", Austr. Jour. of App. Sci., Vol. 9, 1958, pp. 337-348.

17. Schmit, L. A. Jr.: "Structural Design By Systematic Synthesis", Proc. ASCE, 2nd Conference on Electr. Comput., Pitts, Pa., Sept. 8-9, 1960, pp. 105-132.

18. Klein, B.: "A simple Method of Matrix Structural Analysis", Jour. Aeron. Sci., Jan. 1957, pp. 39-46.

19. Schmit, L. A. Jr., and R. H. Mallett: "Design Synthesis in Multidimensional Space with Automated Material Selection", Research Report, Engg. Design Centre, Case Inst. of Tech. Clev., Ohio, Aug. 1962.

20. Schmit, L. A. and T. P. Kicher: "Synthesis of Material and Configuration Selection", Jour. Str. Div., Proc. ASCE, ST 3, June 1963, pp. 79-102.

21. Dorn, W. S., R. E. Gomory, H. J. Greenberg: "Automatic Design of Optimal Structures", Jour. de Mecanique, Vol. 3, No. 1, March 1964, pp. 25-52.

22. Best, G. C.: "Completely Automated Weight Minimization Method for High Speed Digital Computers", Jour. Aircraft, Vol. 9, No. 3, May-June 1964, pp. 129-133.

23. Schmit, L. A., R. L. Fox: "An Integrated Approach to Structural Synthesis and Analysis", AIAA Jour., Vol. 3, No. 6, June 1964, pp. 1104-1112.

24. Razani, R.: "Behaviour of Fully Stressed Design of Structures and It's Relationship to Minimum Weight Design", AIAA Jour., Vol. 3, No. 12, Dec. 1965, pp. 2262-2268.

25. Brown, D. M., H. S. Alfredo.: "Structural Optimization by Nonlinear Programming", Jour. Str. Div. Proc. ASCE, ST 6, Dec. 1966, pp. 319-340.

26. Danzig, G. B.: "Linear Programming Extensions", Prinction University Press 1963.

27. Rosen, J. B.: "The Gradient Projection Method for Nonlinear Programming, Part I and Part II., J.Soc. Industries. App. Math. Vol.8, No. 1, March 1960, Vol. 9, No. 4, December 1961.

28. Kelley, J. E. : "The Cutting Plane Method for Solving Conves Programs"., J. Soc. Indust. App. Math., Vol. 8, No. 4., 1960.

29. Dickinson, J. R. : "The Use of Gradient Methods in Optimization"., Canadian General Electric Report., September 28, 1959.

30. Arrow, K. J.: " A Gradient Method for Approximating Saddle Points and Constrained Maxima"., The Rand Corporation, Santa Monica, California.

31. Hooke, R., Jerves.T. A., : " Direct Search " Solution of Numerical and Statistical Problems J. Assoc. Court., Mach. 8, 119, 1961.

32. Brooks, S. H. : "A Discussion of Random Methods for Seeking Maxima". Oper. Res. 6., 1958.

33. Leon, A. : "A Classified Bibliography on Optimization"., Recent advances in optimization by A. Lavi and T. P. Vogl, John Wiley and Sons, New York - 1965.

34. Carroll C. W. : "The created response surface technique for optimizing non-linear restrained system", Operations Research, Volume. 9, 1961.

35. Glass, S. and Cooper, L. : "Sequential Search", "A method of solving constrained optimization problem", J. Ass. Company. Mach. Vol. 12, No. 1, January 1965.

36. Gray, D. S. : "Boundary conditions in optimization problem", "Recent advances in optimization technique" - A. Lavi and T. P. Vogl, John Wiley and sons , New York 1965.

37. Gallagher, P. J. : "Mop-1, An optimizing routine for IBM 650 Canadian G. E. Civilian atomic power dept. report R60CAP35, 1960.

38. Best, G. C. : "Completely automatic weight minimization method for high speed digital computers", J. Aircraft 1, 129 - 133, 1964.

39. Tiwari, S. K. : "Vibration analysis and design optimization studies of space frames", Master's degree thesis, Mech. Eng. McMaster University, May. 1968.

40. Raghava, R. S. : "Vibration analysis and design optimization studies of space frames", Master's degree , - Thesis to be submitted., Mech. Eng., McMaster University, 1968

APPENDIX I

A Direct Search Code

The authors of this direct search code[11] claim that this method has been successfully applied to many of the peak finding problems of linear and nonlinear equations with complicated and also ill-behaved functions, that cannot effectively be handled by other methods.

The search has two main phases: first a one variable-at-a-time search and then a pattern move. The one variable at a time search gives the most promising direction for the pattern move. The optimization process is initiated by picking up a arbitrary starting point, R . Then the independent variables are changed one at a time in an order selected initially. Let x_i be the variable first under study. This is first incremented by an amount equal to Δ_i, holding the other variables at their initial value. If $F(x_1, x_2, \ldots, x_i+\Delta_i \ldots x_m) > F(x_1 \ldots x_i \ldots x_m)$ there is some reason for trying further in the same direction. Now the step size is increased to $\lambda_i \Delta_i, (\lambda_i > 1)$ and if a better functional value is obtained, a new step length of $\lambda_i^2 \Delta_i$ is tried. The Search is kept going until no further improvement is obtained. If a step size $\lambda_i^{n+1} \Delta_i$ was unsuccessful, we return to the point $\lambda_i^n \Delta_i$ and a new sequence with initial step size equal to Δ_i is started at this point, following the same scheme as before. If an increment in the positive direction is not successful, the whole operation is tried in the negative direction. Finally we reach a point where no improvement is obtained by moving variable x_i either Δ_i or $-\Delta_i$. This is a best point for x_i temporarily. In the same way all the variables are tried and the best temporary functional value F' is found, after moving all variables x_i to x_i^1.

Pattern Move

If F is better than F', there again is a reason to believe that moving in the direction $F \rightarrow F'$ will give a better result. So a pattern move is made in that direction, incrementing the co-ordinates of the point x_i^1 by an amount proportional to the change experienced for co-ordinates in going from F to F'. This rate of change will be greater than 1. The pattern move will be $x_i'' = (x_i' - x_i)\Delta P$. If the functional value F'' after the pattern move happens to be better than F', a new pattern move with step length $\lambda P.\Delta P$, $(\Delta P > 1)$ is taken in the same direction. The process is continued with step lengths $\lambda^2 P.\Delta P$, $\lambda^3 P.\Delta P$ etc. until no further improvement is achieved. As done previously, a new sequence of pattern move with initial step size equal to ΔP is started after a failure to improve, and if a positive direction is not a success, the negative direction is tried. Finally we reach a point F''' at which no improvement is reached by a pattern move in either direction.

If F is better than F , a new set of 'one variable at a time' search is attempted and the process is kept going until no better points are found. The end of the process will always be the starting point of a new 'one variable at a time' search.

Dickinson's Random Strategy[37]

An upper limit (upper x_i) and a lower limit (lower x_i) are specified for each variable x_i. Then we calculate a certain number of optimum values for points selected at random by using the expression

$$x_i = \text{Lower } x_i + r\,(\text{upper } x_i - \text{lower } x_i)$$

r is a random number between 0 and 1. It is different for each setting of each independent variable. From the optimum values, a best few (a certain number) are selected and examined to determine a new and reduced range for each variable. A second group of optimum values are evaluated using the new upper and lower limits of the independent variables. As before a certain number of best values are selected, and these, together with the previous best values are examined to determine a new improved range for each individual independent variables. The whole process is repeated as often as is necessary to obtain a sufficiently small range for the variables.

Method of Approximate Programming[12]

The method of approximate programming (MAP), is a differential technique which utilized the linear programming algorithm repetitively in such a way that the solution of a linear problem converges to the solution of a nonlinear problem. This method has given substantial success in the petroleum industry where the authors have solved problems with 7 nonlinear variables, 80 linear variables involving 45 constraints out of which 10 were nonlinear. The method is not rigorous and gives good results for well behaved functions. We have as before the functions.

$$\left.\begin{array}{lll} U = U(x_1,x_2,\ldots,x_n) \text{ Minimize} & & \\ \psi_j(x_1,x_2,\ldots,x_n) = d_j & & j = 1,m \\ \phi_k(x_1,x_2,\ldots,x_n) \quad b_k & & k = 1,1 \end{array}\right\} \ldots \text{(A)}$$

The constraints may be linear and nonlinear. We start by linearizing in the region about the point x^o, by expansion as a Taylor's series and ignoring higher orders other than linear. Equation (A) become,

$$U = U(x_1^o,\ldots,x_n^o) + \sum_{i=1}^{n} (x_i - x_i^o) \frac{\partial U(x_1^o,\ldots,x_n^o)}{\partial x_i}$$

$$\psi_j(x_1^o,\ldots,x_n^o) + \sum_{i=1}^{n} (x_i - x_i^o) \frac{\partial \psi_j(x_1^o,\ldots,x_n^o)}{\partial x_i} = d_j$$

$$\phi_k(x_1^o,\ldots,x_n^o) + \sum_{i=1}^{n} (x_i - x_i^o) \frac{\partial \phi_k(x_1^o,\ldots,x_n^o)}{\partial x_i} \leqslant b_k$$

Now let,

$$\frac{\partial U(x_i^o, \ldots, x_n^o)}{\partial x_i} = c_i \text{ a constant}$$

$$U^o = U(x_i^o, \ldots, x_n^o), \qquad \psi_i^o = \psi_i(x_i^o, \ldots, x_n^o)$$

$$u_{ji} = \frac{\partial \psi_j(x_i^o, \ldots, x_n^o)}{\partial x_i}, \quad \phi_k^o = \phi_k(x_i^o, \ldots, x_n^o)$$

$$v_{ki} = \frac{\partial \phi_k(x_1^o, \ldots, x_n^o)}{\partial x_i}$$

$$\delta x_i = x_i - x_i^o$$

From the above equations we can write

$$U - U^o = \sum_{i=1}^{n} c_i \, \delta x_i \qquad \text{minimum}$$

$$\sum_{i=1}^{n} u_{ij} \, \delta x_i = d_j - \psi_j^o$$

$$\sum_{i=1}^{n} v_{ki} \, \delta x_i \leqslant b_k - \phi_k^o$$

The problem has been set up in such a way that it can be solved by a linear programming problem. However, since δx_i are not always positive, we let $\delta x_i^+ = \delta x_i$ when $\delta x_i \geqslant 0$ and

$$\delta x_i^- = -\delta x_i \text{ when } \delta x_i \leqslant 0$$

Thus we introduce a standard substitution that makes δx_i^+ and δx_i^- positive quantities all the time during linear programming. If the above system of equations is solved and if we get $(\delta x_i^+ - \delta x_i^-)$ 0 for every $i = 1, 2, \ldots, n$, we have solved the problem. Before starting the linear programming, we also limit the changes in x_i to a small amount, to prevent the linearization from becoming invalid, i.e.,

$$|\delta x_i| \leqslant m_i$$

The value of m_i is chosen by trial or by intution.

APPENDIX II

ALTERNATE SEARCH ROUTINE

```
$JOB            003723 V GURUNATHAN      100    010    030
$IBJOB          DECK
$IBFTC MAIN
      DIMENSION Z(20),STEPX(20),X(40),S(40),DELX(20), A(40,40),B(40),
     1BB(40),C(40),II(40),III(40),V(20),RMAX(20),RMIN(20),XUP(20),
     2XLO(20),PSI(20)
      COMMON NPR,WATE(10)
C
C     READ IN AND STORE THE PROBLEM VARIABLES IN COMPUTER MEMORY
      READ (5,300) NPROB
      NPR = 0
   15 NPR=NPR+1
      WRITE(6,100) NPR
      READ(5,200)K,NUM,NEQ,NMAX,INDEXI
      READ(5,201)P,F,G,TES
      IMM=2*K
      IM=IMM+NEQ
      IN=IM+IMM
      M = IM+NUM
      READ(5,202)(Z(I),I=1,K)
      READ(5,203)(BB(I),I=1,IMM)
      READ(5,205)(STEPX(I),I=1,K)
      READ(5,206)( RMAX(I),I=1,K),(RMIN(I),I=1,K)
      READ(5,204)(WATE(I),I=1,NUM)
      DO 5 I=1,M
    5 III(I)=IMM+I
C
C     TRANSFER INITIAL VALUES FROM STORAGE TO WORKING LOCATIONS
C
      NCYCLE = 0
    1 N = IN+NUM
      MM=IMM
      NUMR =NUM
      INDEX = INDEXI
C
C     START DIRECT SEARCH ROUTINE
C
      CALL SEARCH (Z,K,P, RMAX,RMIN,F,G,DELX,NEQ,NUMR,NCYCLE)
C
      DO 2 I=1,M
    2 II(I)=III(I)
C
C     SET UP THE MATRIX FOR LINEAR APPROXIMATION
C
      CALL MATRIX(A,B,BB,Z,STEPX,C,NUMR,N,M,MM,X,S,II,K,NEQ)
C
C     CALCULATE INITIAL OPTIMUM VALUE
C
      CALL REALU(UR,Z,UI)
      WRITE(6,110) UI
      NCYCLE = NCYCLE + 1
      WRITE(6,208) NCYCLE
C
C     START LINEAR APPROXIMATION ROUTINE
C
C      SIMPLEX OPERATION
```

```
C
      CALL SIMPLE (A,B,C,NUMR,N,M,MM,INDEX,X,NMAX,II,S)
C
C     CALCULATE NEW VALUES FOR BASIC VARIABLES
C
      DO 31 I=1,K
      Z(I) = Z(I) +X(2*I-1)-X(2*I)
      DELX(I) = X(2*I-1) -X(2*I)
   31 CONTINUE
      WRITE(6,104) (Z(I),I=1,K)
C
C     CHECK FOR FINAL OPTIMUM VALUE
C
      CALL REALU(UR,Z,UP)
      CALL CONST(PSI,Z)
      WRITE(6,209)(PSI(J),J=1,NUM)
      WRITE(6,111) UP
      IF(ABS(UI-UP).LT.TES ) GO TO 1000
C
      GO TO 1
 1000 WRITE(6,107)
      WRITE(6,108) UP,UR
      WRITE(6,109) (Z(I),I=1,K)
      IF(NPR.LT.NPROB) GO TO 15
  100 FORMAT(///,2X,12HPROBLEM NO. ,I3,//)
  104 FORMAT(/,2X,8HV MATRIX,/,(2X,8F12.5))
  107 FORMAT(/,2X,19HFINAL OPTIMUM VALUE)
  108 FORMAT(/,2X,18HARTIFICIAL OPTIMUM,2X,F15.5,2X,12HREAL OPTIMUM,
     12X,F15.5)
  109 FORMAT(/,2X,15HVARIABLES VALUE,/,(2X,8F12.5))
  110 FORMAT(/,2X,21HINITIAL OPTIMUM VALUE,2X,F15.5)
  111 FORMAT(/,2X,27HOPTIMUM VALUE AFTER SIMPLEX,2X,F15.5)
  200 FORMAT(8I3)
  201 FORMAT(5F10.4)
  202 FORMAT(8F10.5)
  203 FORMAT(8F10.5)
  204 FORMAT(5F10.4)
  205 FORMAT(8F10.5)
  206 FORMAT(8F10.5)
  208 FORMAT(/,2X,13HSIMPLEX CYCLE,2X,I4,/)
  209 FORMAT(/,2X,9HPSI VALUE,/,2X,8F12.4)
  300 FORMAT(I2)
      STOP
      END
$IBFTC SUB 1
      SUBROUTINE MATRIX(A,B,BB,Z,STEPX,C,NUMR,N,M,MM,X,S,II,K,NEQ)
      DIMENSION Z(20),STEPX(20),X(40),PSI(20),PHI(50),PSIN(20),PHIN(20)
     1A(40,40),B(40),BB(40),C(40),II(40),S(40)
      COMMON NPR,WATE(10)
C
C     SET UP INCREMENTS AND THE MATRIX
C
C     SET UP COEFFICIENTS OF OBJECTIVE FUNCTION
C
C
      DO 5 I=1,M
      DO 5 J=1,N
    5 A(I,J) = 0.0
```

```
      CALL REALU(U ,Z,UK)
      IF(NUMR.NE.0) CALL CONST(PSI ,Z)
      CALL ENEQ (PHI,Z)
      DO 10 I =1,K
      Z(I) = Z(I) + STEPX(I)
      CALL REALU(UN,Z,UK)
      IJ = 2*I-1
      S(IJ) = (UN-U)/STEPX(I)
      S(IJ+1) = -S(IJ)
      IF(NUMR.NE.0) CALL CONST(PSIN,Z)
      CALL ENEQ (PHIN,Z)
      DO 9 J=1,NEQ
      JI = J+MM
      A(JI,IJ) =-(PHIN(J)-PHI(J))/STEPX(I)
      A(JI,IJ+1) = -A(JI,IJ)
    9 CONTINUE
      IF(NUMR.EQ.0) GO TO 11
      DO 6 J=1,NUMR
      JI =J+MM+NEQ
      A(JI,IJ)=(PSIN(J)-PSI(J))/STEPX(I)
      A(JI,IJ+1)  = -A(JI,IJ)
    6 CONTINUE
   11 CONTINUE
      Z(I) = Z(I) - STEPX(I)
   10 CONTINUE
C
C     SET UP EQUATIONS FOR UPPER AND LOWER LIMITS
C
      MMK =    MM-1
      J = 0
      DO 12 I=1,MMK,2
       J = J+1
      JJ = 2*J-1
      A(I,JJ) =1.0
      A(I+1,JJ  ) =-1.0
      A(I+1,JJ+1) = 1.0
   12 A(I,JJ+1) =-1.0
C
C     SET UP B(I)'S
C
      DO 20 I=1,MM
   20 B(I) = BB(I)
      MP=MM+1
      MEQ = MM+NEQ
      DO 19 I=MP,MEQ
      J = I-MM
      IF(ABS(PHI(J)).LE.0.001)PHI(J) = 0.0
   19 B(I) = PHI(J)
      IF(NUMR.EQ.0) GO TO 16
      MEQI=MEQ+1
      DO 21 I=MEQI,M
      J=I-MEQ
   21 B(I) = - PSI (J)
   16 CONTINUE
C
C     SET UP SLACK VARIABLES
C
      DO 22 I=1,M
```

6063

```
      DO 15 J=MP,N
      S( J) = 0.0
   15 A(I,J) =0.0
      MI=MM+I
   22 A(I,MI) = 1.0
C
C
C
C     CHECK FOR NEGATIVE VALUES OF B AND REARRANGE IF NECESSARY
C
      CALL ORDER (A,B,NUMR,N,M,MM,K,II)
C
C     SET INITIAL FEASIBLE BASIS
C
      DO 30 I =1,MM
   30 X(I) = 0.0
      DO 35 I=1,M
      MMI =MM+I
   35 X(MMI) = B(I)
      RETURN
      END
$IBFTC SUB 2
      SUBROUTINE ORDER (A,B,NNN,N,M,MM,K,LL)
      DIMENSION A(40,40),B(40),KL(20),LL(20)
      COMMON NPR,WATE(10)
C
C      ROWS WITH SLACKS CHECKED
C
      NK = 0
      KN = M-NNN
      DO 21 I = 1,KN
      IF(B(I).GT.(-1.0E-06)) GO TO 21
C
C     STORE NEGATIVE B ROWS
C
      NK =NK+1
       KL(NK) =I
  21  CONTINUE
C
C     IF ALL B'S ARE POSITIVE CHECK THE ROWS WITHARTIFICIAL VARIABLES
C
      IF(NK.EQ.0) GO TO 25
      ML=KN-NK+1
      ND=0
      DO 22 I=ML,KN
      ND=ND+1
C
C     CHECK IF INTERCHANGE OF ROWS IS NECESSARY
C
      DO23 J=1,NK
   23 IF(KL(J).EQ.I) GO TO 24
C
C     INTERCHANGE ROWS AND ALTER THE SIGNS OF A'S AND B'S
C
      IE = KL(ND)
      DO 26 JJ=1,MM
      TEMP=A(I,JJ)
      A(I,JJ) = -A(IE,JJ)
```

```
   26  A(IE,JJ) = TEMP
       TEMP=B(I)
       B(I) =-B  (IE)
       B(IE) = TEMP
       GO TO 27
C
C      INTERCHANGE OF ROWS NOT NECESSARY, CHANGE SIGNS OF A'S AND B'S
C
   24  DO 29 JK=1,MM
   29  A(I,JK) =-A(I,JK)
       B(I)=-B(I)
C
C      SHIFT THE NOW NEGATIVE SLACKVARIABLEOUT OF BASIS
C
    27 MD=MM+ND
       A(I,MD)=-1.0
   22  CONTINUE
       IF(NNN.EQ.0) GO TO 40
C
C      CHECK THE ROWS WITH ARTIFICIAL VARIABLES
C
   25  MP=KN+1
       DO 31 I=MP,M
       IF(B(I).GE.(-1.0E-06)) GO TO 32
C
C      B IS NEGATIVE, ALTER THE SIGNS OF B AND ALL A'S EXCEPT THE
C      ARTIFICIAL ONES
C
       DO33 J=1,MM
    33 A(I,J)=-A(I,J)
       B(I)=-B(I)
       GO TO 31
   32  IF(B(I).LT.0.0) B(I)=0.0
   31  CONTINUE
    40 CONTINUE
C
C      CHECK IF CHANGE IN BASIS IS REQUIRED
C
       IF(NK.EQ.0) GO TO 50
C
C      CHANGE THE BASIS
C
       DO 35 I=1,M
       MMI=MM+I
       A(I,MMI) =0.0
       DO 36 J=1,NK
       NJ=N+J
   36  A(I,NJ) =0.0
       MIL=MMI+NK
       LL(I)=MIL
        A(I,MIL) =1.0
    35 CONTINUE
       N=N+NK
       NNN=NNN+NK
       MM = MM+NK
    50 CONTINUE
       RETURN
       END
```

16065

```
$IBFTC SUB 3
      SUBROUTINE SIMPLE(A,B,C,NN,N,M,MM,INDEX,X,NMAX,II,S)
       DIMENSION S(40)
      DIMENSION A(40,40),B(40),C(40),II(50),X(40)
      COMMON NPR,WATE(10)
C
C
C     PHASE 1 OR 2 OF LINEAR PROGRAMING STANDARD SIMPLEX
      NCYCLE = 1
C     INDEX =0 FOR PHASE 2 INDEX =1 FOR PHASE 1
      IF (INDEX.NE.1) GO TO 8
C
C     CALCULATION OF ALLC(J) FOR VARIABLES NOT IN BASIS
C
      MM=N-M
      MMM=M+1-NN
      DO 5 J=1,MM
      C(J)=0.
      DO 5 I=MMM,M
    5 C(J)= C(J)-A(I,J)
C
C     SET C(J) = 1.E10 FOR VARIABLES IN BASIS
C
      MA=MM+1
      DO4 J=MA,N
    4 C(J)=1.E10
C
C     CALCULATE INITIAL U0
C
      U0=0.
      DO 6 I=MMM,M
    6 U0=U0+B(I)
       GO TO 9
   8    MB=M+1
      DO 12 J=1,N
   12 C(J) =S(J)
      U0=0.0
C     SELECT SMALL C(J) WHICH IS C(L)
C
    9 SMALL=C(1)
      L=1
      DO 10 I=2,N
       IF ( C(I).GE.SMALL) GO TO 10
       SMALL=C(I)
      L=I
   10 CONTINUE
C
C     TESTING FOR OPTIMUM  NOTE ALLOWANCE FOR ROUND OFF ERROR
      IF(C(L)+1.E-5.GE.0.) GO TO 100
C
C     TESTING FOR FINITE OPTIMUM   ALLOWANCE FOR ROUND OFF ERROR
C
      DO 15 I=1,M
       IF(A(I,L).GT.1.E-5) GO TO 16
   15 CONTINUE
      WRITE(6,210)
       GO TO 101
C
```

```
C     SELECT SMALLEST RATIO FOR WHICH A(I,L) GT.0. GIVING EQN.(LL)
C     IN WHICH VARIABLE IS DROPPED
C
  16   SMALL = +1.0E+10
      LL=1
      DO 18 I=1,M
       IF (A(I,L).LE.1.E-5) GO TO18
      IF(B(I)/A(I,L).GT.SMALL) GO TO 18
       SMALL=B(I)/A(I,L)
      LL=I
   18 CONTINUE
C
C     BRINGING C(K) BACK TO 0 BEFORE CONVERTING TO NEW CANNONICAL FORM
      K = II(LL)
      C(K)=0.
C
C      CONVERTING TO NEW CANONICAL FORM
C
      B(LL)=B(LL)/A(LL,L)
      U0=U0+B(LL)*C(L)
      DO 30 J=1,N
      IF(J.EQ.L) GO TO 30
      A(LL,J)=A(LL,J)/A(LL,L)
      C(J)=C(J)-A(LL,J)*C(L)
   30 CONTINUE
      A(LL,L)=1.
      DO 33 I=1,M
      IF(I.EQ.LL) GO TO 33
      Y=A(I,L)
      B(I)= B(I)-B(LL)*A(I,L)
      DO 31 J=1,N
   31 A(I,J)=A(I,J)-A(LL,J)*Y
   33 CONTINUE
C
C     SWITCH BASIS TAGS ON LL EQN.
C
      C(L) =1.E10
      KK=II(LL)
      II(LL)=L
C
C     SETTING OLD VARIABLE IN BASIS =0
      X(KK)=0.
C
C      RECORD NEW VALUES OF X IN MEMORY. VARIABLES NOT IN BASIS ARE
C     ALREADY 0 IN MEMORY
C
      DO 40 I=1,M
      K=II(I)
   40 X(K)=B(I)
C
C     ITERATION COMMAND
C
      NCYCLE = NCYCLE + 1
      IF(NCYCLE.EQ.NMAX) GO TO 110
      GO TO 9
C
C      OUTPUT
  100 CONTINUE
```

```
      IF(INDEX.NE.1) GO TO 101
C
C     CALCULATION OF CANONICAL FORM OF OPT. EQN. FOT INITIAL FEASIBLE BA
 102  N = N-NN
      MC=M+1
      DO 94 J=MC,N
   94 S(J) = 0.0
      DO 95 J = 1,N
   95 C(J) = S(J)
      U0=0.
      DO 90 I=1,M
      K=II(I)
      Q=C(K)
      U0=U0+B(I)*Q
      DO 90 J=1,N
   90 C(J)=C(J)-A(I,J)*Q
      INDEX = 0
      DO 91 I=1,M
      K = II(I)
   91 C(K) = 1.E10
        GO TO 9
  101 RETURN
  110 WRITE(6,211) NCYCLE
 111  STOP
  200 FORMAT(2X,4HU0= ,E11.5)
  201 FORMAT(2X,8HA MATRIX,/,(1X,10F11.5))
  202 FORMAT(2X,22HVARIABLES IN BASIS ARE,/,(2X,30I3) )
  206 FORMAT(2X,28HPHASE II OF SIMPLEX SOLUTION,//)
  208 FORMAT(2X,8HC MATRIX,/,(2X,8E13.5))
  210 FORMAT(2X,17HNO FINITE OPTIMUM)
  211 FORMAT(2X,30HPROCESS DID NOT CONVERGE AFTER,2X,E12.5,2X,6HCYCLES)
      END
$IBFTC SUB 4
      SUBROUTINE SEARCH (X,N,H, RMAX,RMIN,F,G,DELX,NEQ,NUMR,NCY)
      DIMENSION X(20),RMAX(20),RMIN(20),TEST(20),
     1DELX(20),XO(20),PHI(20)
      COMMON NPR,WATE(10)
C
C
C     GENERATE DELX(I) AND TEST(I)
      DO 20 I = 1,N
      TEST(I) =(RMAX(I)-RMIN(I))/F
   20 CONTINUE
      M = 0
      CALL REALU(UO,X,UR)
      WRITE (6,11) M, UO, ( X(I), I=1,N )
C
C     MAKE SEARCH BY PATTERN MOVE
C
      J = 0
      IF(NCY.EQ.0) GO TO 6
    1 DO 2 I = 1,N
    2 X(I) =X(I) + DELX(I)*H**J
      CALL ENEQ(PHI,X)
      DO 3 JJ = 1,NEQ
    3 IF(PHI(JJ).LT.0.0) GO TO 5
      CALL REALU(U ,X,UR)
      IF(U.GT.UO) GO TO 5
```

```
      UO = U
      J = J+1
      M = M + 1
      GO TO 1
C
C     REDUCE STEP SIZE AND MAKE PATTERN MOVE
C
    5 DO 4 I = 1,N
    4 X(I) =X(I) - DELX(I)*H**J
      IF(J.EQ.0) GO TO 6
      J = 0
      GO TO 1
    6 CONTINUE
C
C     PATTERN MOVE FAILS, MAKE EXPLORATORY SEARCH
C
      DO 7 I = 1,N
    7  XO(I) = X(I)
       DO 9 I=1,N
      X(I) = X(I)+TEST(I)
      CALL ENEQ(PHI,X)
      DO 10 JJ=1,NEQ
   10 IF(PHI(JJ).LT.0.0) GO TO 12
      CALL REALU(U ,X,UR)
      IF(U.GT.UO) GO TO 12
      UO = U
      GO TO 9
   12 X(I) = X(I)-2.*TEST(I)
      CALL ENEQ(PHI,X)
      DO 13 JJ=1,NEQ
   13 IF(PHI(JJ).LT.0.0) GO TO 14
      CALL REALU(U ,X,UR)
      IF(U.GT.UO) GO TO 14
      UO = U
      GO TO 9
   14 X(I) = X(I) +TEST(I)
    9 CONTINUE
      NN = 0
C
C     FIND THE DIRECTION FOR PATTERN MOVE
C
       DO15 I=1,N
      DELX(I) =X(I)-XO(I)
      IF(ABS(DELX(I)).LT.G) NN=NN+1
      IF(NN.EQ.N) GO TO 17
   15 CONTINUE
C
C     MAKE SEARCH BY PATTERN MOVE
C
      GO TO 1
   17 CONTINUE
C
C     REDUCE STEP SIZE AND MAKE EXPLORATORY MOVE
C
      DO 18 I=1,N
      TEST(I) =TEST(I)/4.
C
C     DIRECT SEARCH FAILS
```

```
C
      IF(ABS(TEST(I)).LT.G) GO TO 19
   18 CONTINUE
      GO TO 6
   19 CONTINUE
      CALL ENEQ(PHI,X)
       WRITE(6,8)M,UO,(X(I),I=1,N)
      WRITE(6,106)                    (PHI(I),I=1,NEQ)
    8 FORMAT(2X,19HRESULT AFTER SEARCH,/,2X,I5,E20.5,/,(2X,6E20.5))
  106 FORMAT(/,2X,(/,2X,9F10.4))
   11 FORMAT ( 1X, I5, E20.5, / ( 1X, 6E20.5 ) )
      RETURN
      END
$IBFTC SUB 5
      SUBROUTINE REALU(U,X,UR)
      DIMENSION X(20),PSI(20)
      COMMON NPR,WATE(10)
C
C     EXPRESSIONS FOR OBJECTIVE FUNCTION
C
      GO TO (10,20,30,40,50,60),NPR
   10 CONTINUE
C
C     ROSENBROCK'S PROBLEM
C
      NUMR = 0
      UR=100.0*(X(2)-X(1)**2)**2+(1.0-X(1))**2
      GO TO 100
   20 CONTINUE
C
C     LEON'S PROBLEM
C
      NUMR = 1
      UR=183.0-44.0*X(1)-42.0*X(2)+8.0*X(1)**2-12.0*X(1)*X(2)+17.0*X(2)
     1**2
      GO TO 100
   30 CONTINUE
C
C     FIACCO AND MCCORMIC'S PROBLEM
C
      NUMR = 1
      UR=X(1)**3-6.0*X(1)**2+11.0*X(1)+X(3)
      GO TO 100
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
  100 CONTINUE
      CALL CONST(PSI,X)
      U = UR
      DO 1 I=1,NUMR
      U=U +ABS(PSI(I))*WATE(I)
    1 CONTINUE
      RETURN
      END
$IBFTC SUB 6
      SUBROUTINE CONST(PSI,X)
      DIMENSION X(20),PSI(20)
      COMMON NPR
```

```
C
C      EXPRESSIONS FOR EQUALITY CONSTRAINTS
C
       GO TO (10,20,30,40,50,60),NPR
   10  CONTINUE
C
C      ROSENBROCK'S PROBLEM
C
       PSI(1) = 0.0
       GO TO 100
   20  CONTINUE
C
C      LEON'S PROBLEM
C
       PSI(1) = 2.0*X(1)+X(2)-10.0
       GO TO 100
   30  CONTINUE
C
C      FIACCO AND MCCORMIC'S PROBLEM
C
       PSI(1) = X(1)**2+X(2)**2+X(3)**2-4.0
       GO TO 100
   40  CONTINUE
   50  CONTINUE
   60  CONTINUE
  100  CONTINUE
       RETURN
       END
$IBFTC SUB 7
       SUBROUTINE ENEQ(PHI,X)
       DIMENSION X(20),PHI(20)
       COMMON NPR
C
C      EXPRESSIONS FOR INEQUALITY CONSTRAINTS
C
       GO TO (10,20,30,40,50,60),NPR
   10  CONTINUE
C
C      ROSENBROCK'S PROBLEM
C
       PHI(1)=X(1)
       PHI(2)=X(2)
       GO TO 100
   20  CONTINUE
C
C      LEON'S PROBLEM
C
       PHI(1)=X(1)
       PHI(2)=X(2)
       GO TO 100
   30  CONTINUE
C
C      FIACCO AND MCCORMIC'S PROBLEM
C
       PHI(1)=X(1)
       PHI(2)=X(2)
       PHI(3)=X(3)
       PHI(4) = 5.0-X(3)
```

```
      PHI(5) = X(3)**2-X(1)**2-X(2)**2
      GO TO 100
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
  100 CONTINUE
      RETURN
      END
$ENTRY
 3
  2  0  2 99  1
1.5       40.0      0.000001  0.00001
0.0       0.0
0.0001    0.0001    0.0001    0.0001
0.00001   0.00001
4.0       4.0       0.0       0.0

  2  1  2 99  1
1.5       40.0      0.000001  0.002
0.0       0.0
0.01      0.01      0.01      0.01
0.00001   0.00001
4.0       4.0       0.0       0.0
10.0
  3  1  5 99  1
1.5       40.0      0.000001  0.00001
0.0       0.0       0.0
0.05      0.05      0.10      0.10      0.05      0.05
0.00001   0.00001   0.00001
4.0       4.0       4.0       0.0       0.0       0.0
10.0
$IBSYS
```

SUBROUTINES FOR THE OBJECTIVE FUNCTIONS AND CONSTRAINT EQUATIONS FOR THREE SOLVED PROBLEMS

```
$IBFTC SUB 5
      SUBROUTINE REALU(U,X,UR)
      DIMENSION X(20),PSI(20)
      COMMON NPR,WATE(10)
      GO TO (10,20,30,40,50,60),NPR
   10 CONTINUE
C
C     GLASS AND COOPER'S PROBLEM
C
      NUMR=2
      UR=-SQRT (25.0-(X(1)-5.0)**2-(X(2)-5.0)**2)
      GO TO 100
   20 CONTINUE
C
C     J.E. KELLEY'S PROBLEM
C
      NUMR = 1
      UR=X(1)-X(2)
      GO TO 100
   30 CONTINUE
C
C     DICKINSON'S PROBLEM
C
      F1=11.0-6.0*X(1)-4.0*X(2)+X(1)*X(1)+2.0*X(2)*X(2)
      F2=17.0-8.0*X(1)-6.0*X(2)+2.0*X(1)*X(1)+X(2)*X(2)
      UR=1.0+F1*F2
      GO TO 100
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
  100  CONTINUE
      CALL CONST(PSI,X)
      U = UR
      DO 1 I=1,NUMR
      U=U +ABS(PSI(I))*WATE(I)
    1 CONTINUE
      RETURN
      END
$IBFTC SUB 6
      SUBROUTINE CONST(PSI,X)
      DIMENSION X(20),PSI(20)
      COMMON NPR
      GO TO (10,20,30,40,50,60),NPR
   10 CONTINUE
C
C     GLASS AND COOPER'S PROBLEM
C
      PSI(1)=X(1)*X(1)-X(2)*4.0
      PSI(2)=(X(2)-6.0)**2-4.0*(X(1)-3.0)
      GO TO 100
```

```
   20 CONTINUE
C
C      J.E. KELLEY'S PROBLEM
C
       PSI(1)=3.0*X(1)*X(1)-2.*X(1)*X(2)+X(2)*X(2)-1.0
       GO TO 100
   30 CONTINUE
       PSI(1)=0.0
       GO TO 100
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
  100  CONTINUE
       RETURN
       END
$IBFTC SUB 7
       SUBROUTINE ENEQ(PHI,X)
       DIMENSION X(20),PHI(20)
       COMMON NPR
       GO TO (10,20,30,40,50,60),NPR
   10 CONTINUE
C
C      GLASS AND COOPER'S PROBLEM
C
       PHI(1)=1.0
       PHI(2)=1.0
       GO TO 100
   20 CONTINUE
C
C      J.E. KELLEY'S PROBLEM
C
       PHI(1)=X(1)
       PHI(2)=X(2)
       GO TO 100
   30 CONTINUE
C
C      DICKINSON'S PROBLEM
C
       PHI(1)=X(1)
       PHI(2)=X(2)
       GO TO 100
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
  100  CONTINUE
       RETURN
       END
C      DATA FOR THE THREE PROBLEMS
$ENTRY
 3
  2   2   2 99   1
1.5        40.0        0.000001  0.0001
7.0        1.0
1.0        1.0         0.5       0.5
0.00001    0.00001
4.0        4.0         0.0       0.0
10.0       10.0
```

```
  2   1   2 99   1
1.5          40.0         0.000001  0.0011
1.0          2.0
0.05         0.05         0.1       0.1
0.00001      0.00001
4.0          4.0          0.0       0.0
10.0
  2   0   2 99   1
1.5          40.0         0.000001  0.0001
5.0          2.0
0.0001       0.0001       0.0001    0.0001
0.00001      0.00001
4.0          4.0          0.0       0.0

$IBSYS
```