Structural Optimization

A Thesis

Presented to

The Faculty of Graduate School

University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

---

by

Hui-Ru Shih

Dr. Eric Sandgren Thesis Supervisor

December, 1985

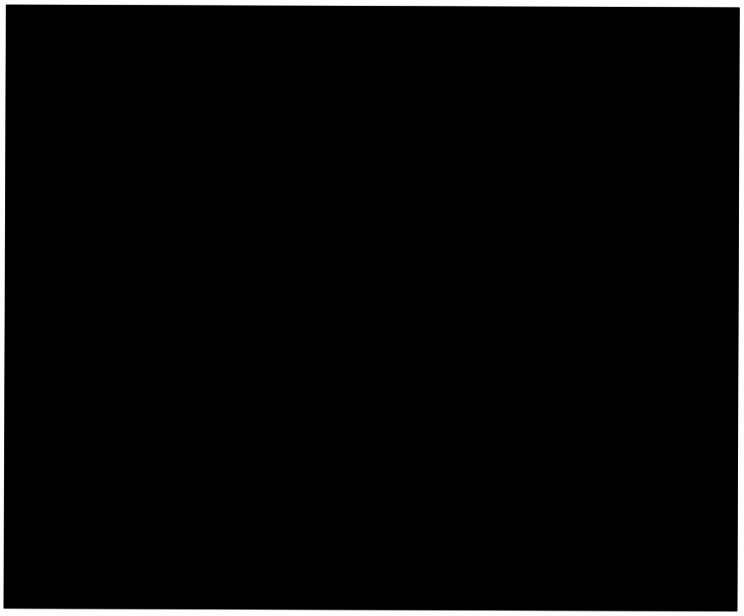
The undersigned, appointed by the Dean of the Graduate Faculty, have examined a thesis entitled

Structural Optimization

presented by Hui-Ru Shih

a candidate for the degree of Master of Science

and hereby certify that in their opinion it is worthy of acceptance.



CONTENTS

1. INTRODUCTION

Structural optimization may be defined in various ways. One definition could be the design and construction of a quality structure at the lowest cost. This definition often turns out to be inappropriate due to both the difficulty of giving a precise definition to cost and the dependence of the cost function on highly variable factors. In common practice the weight or volume of the structure is taken as the objective function. Therefore, in all further developments the objective of structural optimization will be to minimize the weight while maintaining high reliability and rigidity without failure. To ensure that failure will not occur, the stresses in any part of the structure and certain displacements should remain within specified limits. Also, no buckling should occur in any part of the structure, and in many cases the fundamental natural frequency of the structure must be considered, in order to avoid a resonance problem. Judging from this, in order to discuss the theory of structural optimization it is necessary to be able to analyse any structural configuration for these considerations. The finite element method will be used to obtain stresses, displacements, natural frequencies, and any other response quantities.

The history of structural optimization is long and the

literature is extensive. The main classification based on the optimization method can be described as either Mathematical programming methods or Optimality criteria approaches [1]. The fully stressed design is the earliest optimality criteria approach used in structural design [2]. By definition the fully stressed design is the one in which the maximum stress in each of the elements reaches its limiting value at least under one loading condition. This criterion is valid if the structure is statically determinate but is only approximately true for indeterminate structures.

Historically, the application of mathematical programming methods to structural optimization was initiated by Schmit [3]. Moses [4] reviewed the rapid development of the mathematical programming approach to structural design optimization. He divides these applications into three categories : (1) element design, (2) system design, and (3) discrete decisions. Some of the early mathematical programming tools that were applied to the structural design optimization problem include : the sequential unconstrained minimization technique, based on an interior penalty function formulation, and the sequential linear programming approach, particularly the cutting plane method. Moses concludes that the application of mathematical programming methods to the optimum design of structural elements and simple components is a well established and profitable technique. He indicates that a major consideration in structural systems optimization will be to reduce

the number of analyses.

In Ref. [5], Haug and Arora used a design sensitivity analysis coupled with a gradient projection method for the optimal design of trusses and frames. They also use a substructuring concept to deal with large structures. The objective function was to minimize the structural weight. The constraints imposed limitations on the design variables and on quantities describing the structural response (behavior constraints) such as stresses and displacements under multiple static loading cases, natural frequencies and critical buckling loads.

TRUOPT, [6], [7], is applicable to general space truss structures. It uses approximation concepts and the generalized reduced gradient method.

In Refs. [8] through [10], structural optimization with material selection, large and continuum systems, and optimization of geometry and topology are discussed.

Now, we wish to develop an efficient algorithm based on combining a finite element analysis method and a nonlinear programming method for structural optimization problems. This algorithm can be simultaneously applied to general structures, such as trusses, frames, etc, and use both sizing and geometric variables. However, a conventional nonlinear programming approach

was found to require many finite element analyses and resulted in an extremely long computer run time. Because of this major cost of analysis, approximation concepts and approximate analysis methods are used to improve the efficiency of the optimization process. The approximation concepts are used to replace an original nonlinear and implicit structural optimization problem with a sequence of explicit and high-quality approximate problems.

2. ILLUSTRATIVE EXAMPLES

A simple structural optimization problem that will be used to illustrate basic ideas and formulation is the three-bar truss. The first step in optimizing a structure is to idealize the structure as an assemblage of finite elements. The three-bar truss structure is modeled by three elements, as shown in Fig. 2.1. The design objective is to choose the element radii (the design variables), so that the truss is as light as possible, while satisfying constraints on stress, buckling, deflection, natural frequency, and member size.

The objective function F is the volume of the structure

$$F(X)=\sum_{i=1}^{3} L_i(X_i \cdot X_i)\pi \qquad (2.1)$$

Subject to $G_j(X)\geq 0 \quad j=1,2,....m \qquad (2.2)$

and $X_i^l \leq X_i \leq X_i^u \quad i=1,2,3 \qquad (2.3)$

where

G=stress, displacement, and buckling constraints

X=design variables (radius of each member)

X_i^l =lower bound for variable i

X_i^u =upper bound for variable i

L_i =specific length of i-th element

Design data for this truss is given in Table 2.1.

This is a constrained nonlinear optimization problem. The functions G are not explicitly given. Instead, for given values on the variables X, we shall assume that the constraint values are to be obtained by finite element method calculation. Since such calculations may be comparatively expensive, it is necessary to

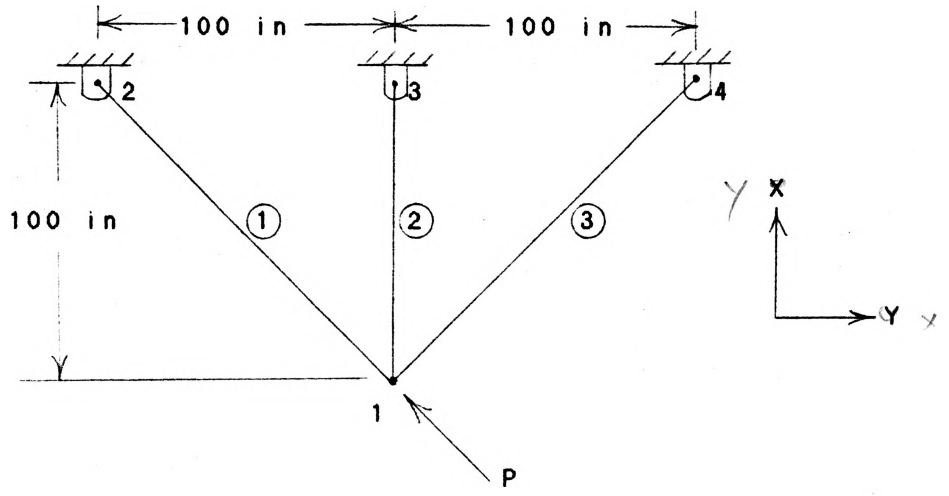


Figure 2.1 Three-bar truss

Table 2.1 Design Data for three-bar truss

Modulus of elasticity=$2.07 \cdot 10^4$ ksi
Material density =0.15 lb/in^3
Lower limit on radius=0.15 in
Upper limit on radius=None
Displacement limit at node
1 in x direction=±0.15 in
in y direction=±0.06 in
Stress limit=±25 ksi
Lower bound on frequency=13.316 HZ
Number of loading conditions=1

Load Data

Node Number	Direction of Load, lbf		
	F_x	F_y	F_z
1	-28300	28300	0.0

find methods which require as few function evalutions as possible in solving this problem.

It is, for example, hardly realistic to solve this problem by a 'conventional' method such as the generalized reduced gradient (GRG) method (nonlinear programming method, using forward difference gradient calculations and a normal line search procedure). The number of function evalutions required by this method is empirically known to grow rapidly when the size of the problem grows. Even a modestly sized problem could require several hundred constraint evaluations. This implies several hundred finite element method calculations which would be unacceptable.

In next sections, Eqs. 2.1 and 2.2 will be processed by a 'conventional' method, followed by an approximation method, with a comparison of the results.

2-1 Nonlinear Programming Method

Central to any optimization scheme are two elements - the optimizer and the analysis processor. The analysis processor accepts values of the design variables X, and produces the constraint values. For the structural optimization problem, the analysis processor essentially provides a structural analysis of a given configuration and finds the stresses, deflections, and natural frequencies. The optimizer provides the input to the

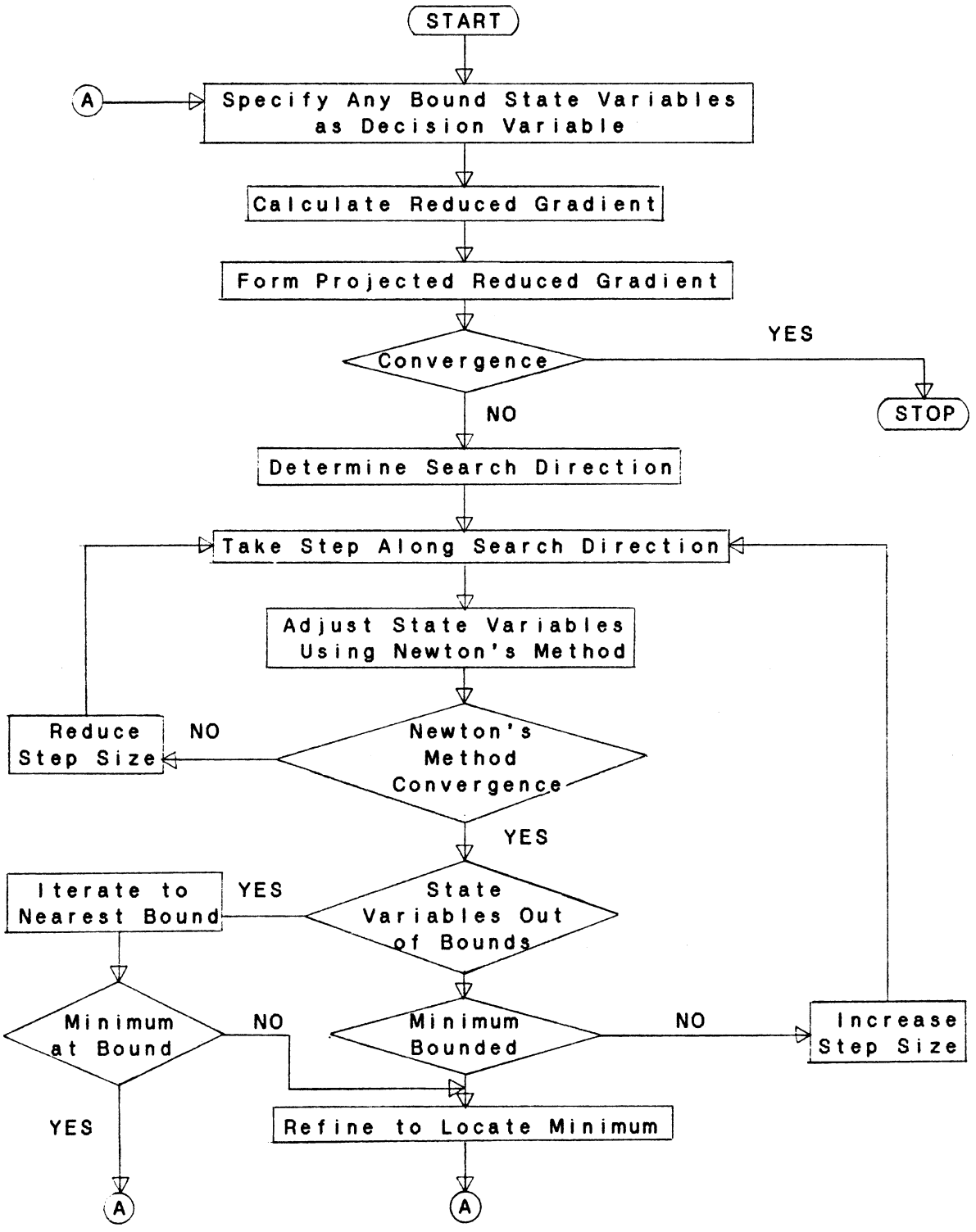


Figure 2.2 Flowchart of GRG

analysis processor and based on the information returned from the analysis processor, determines what changes should be made in the variables to both reduce the objective function and to remain in the feasible region.

2-1-1 Optimization Algorithm

In Eqs. 2.1, 2.2, and 2.3, either the objective function F or constraints G(X) may be nonlinear functions of the design variables X. These equations are solved by the generalized reduced gradient code, OPT [12]. The flowchart of the GRG method is shown in Figure 2.2, and a detailed description of the algorithm will be given in this subsection.

Consider the constrained nonlinear programming problem:

$$\text{Minimize } F(X);\ [X]=[x_1, x_2, \ldots\ldots, x_N]^T \qquad (2.4)$$

$$\text{Subject to } H_l(X)=0 \quad l=1,2,\ldots,L \qquad (2.5)$$

$$G_k(X)\geq 0 \quad k=1,2,\ldots,K \qquad (2.6)$$

$$X^l \leq X \leq X^u \qquad (2.7)$$

where

X= a column vector of design variables

N= total number of design variables

F(X)= the objective function

$H_l(X)$= L equality constraint functions

$G_k(X)$= K inequality constraint functions

X^l, X^u = Lower and upper bounds on the design variables

With no loss of generality, this problem can be transformed into the following set of equations which are handled by the reduced gradient method

$$\text{Minimize } F(X);\ X=[x_1, x_2, \ldots\ldots, x_N]^T \qquad (2.8)$$

$$\text{Subject to } H_m(X)=0 \quad m=1,2,\ldots\ldots,M \qquad (2.9)$$

$$X^l \le X \le X^u \qquad (2.10)$$

The inequality constraints of Eq. 2.6 are included as equality constraints by using the following transformation

$$H_k(X) = G_k(X) - S_k = 0$$

$$0 \le S_k \le \infty \qquad k=1,2,\ldots,K \qquad (2.11)$$

where

S_k = nonnegative slack variables which are added to the original set of design variables.

The parameter N represents the total number of original design variables plus the number of slack variables, and the parameter M represents the total number of equality and inequality constraints. Let's divide the design vector [X] into decision and state variables [Z,Y]

$[Z] = [z_1, z_2, \ldots\ldots, z_Q]^T$ decision variables

$[Y] = [y_1, y_2, \ldots\ldots, z_M]^T$ state variables

$Q = N-M$

The decision variables are independent, but the state variables are slaves to the decision variables and are used only to maintain constraint satisfaction.

In order to derive the reduced gradient vector ,let's examine the first variation of F(X) and H(X)

$$dF=\nabla_z F(X)dz+\nabla_y F(X)dy \qquad (2.12)$$

$$dH=\nabla_z H(X)dz+\nabla_y H(X)dy=0 \qquad (2.13)$$

From Eq. 2.13 we can get

$$dy=-\nabla_y^{-1}H_y\cdot\nabla_z H_z dz \qquad (2.14)$$

Substituting Eq. 2.14 into Eq. 2.12 will yield

$$\nabla_r F(X) = \nabla_z F(X)-\nabla_y F(X)\cdot\nabla_y^{-1}H(X)\cdot\nabla_z H(X) \qquad (2.15)$$

The vector $\nabla_r F(X)$ as defined by Eq. 2.15 is called the reduced gradient.

A minimum of the unconstrained nonlinear problem occurs when the elements of the gradient vanish at the optimum point. Similarly, a minimum of the constrained nonlinear problem occurs when the elements of the projected reduced gradient vanish. Now, by calculating the projected reduced gradient, and the norm of $\nabla_r F(X)$ ($\|\nabla_r F(X)\| = \|\nabla_r F^2(x_i)\|$), we can see if is within some tolerance for convergence,

$$\|\nabla_r F(X)\| \leq \epsilon$$

If it is within this tolerance, then a constrained relative minimum has been obtained which can be shown to be a Kuhn Tucker

point [13]. If it is not, the algorithm continues by determining a new search direction, P_z and P_y, for the decision and state variables respectively. P_z is determined using $\nabla_r F(X)$ by any gradient based unconstrained search technique while an approximation for P_y is determined from the following equation.

$$P_y = \nabla_y^{-1} H(X) \cdot \nabla_z H(X) \cdot P_z \qquad (2.16)$$

A line search is then performed to locate a local minimum of $F(X)$ along P_z and P_y. From a starting point $[Z,Y]$, we move to a new point $[Z^1,Y^1]$ according to the step prescription

$$Z^1 = Z + \alpha P_z$$

$$Y^1 = Y + \alpha P_y$$

where

α = step length parameter

In Eq. 2.13, we constructed linear approximations for the problem constraints. However, while the linearizations are good enough to determine local descent directions, they are not good enough to actually calculate feasible minimum points. This is due to the fact that the first-order approximation can be represented by a plane tangent to the exact constraint surface. For nonlinear constraints, P_z and P_y may not be a feasible direction. Hence when nonlinearities arise in the constraint functions, the point $[Z^1,Y^1]$ is likely to be infeasible. Holding the decision variables Z^1 constant, the state variables Y^1 are adjusted to obtain a feasible point, $[Z^1,Y^1]$. This step is equivalent to the solution

of the set of nonlinear equations

$$H(Z^1, Y^1) = 0$$

This step terminates at some point $X^1 = [Z^1, Y^1]$ where $F(X^1) < F(X)$

If, as a result of the line search , any element of Y^1 is equal to either of its bounds, then a repartitioning of the design variables is carried out by choosing some appropriate decision variable Z_r to take its place in the state variable set and assigning Y_s to the decision variable set.

After redefining $X=[Z,Y]$ to $X^1=[Z^1,Y^1]$, the above steps are repeated.

2-1-2 Analysis Processor

As was discussed earlier, a finite element code is used as the analysis routine by the optimization program. Given a set of design variables, this routine must return stresses, displacements, and frequencies when requested by the optimizer. Here, the finite element package NISA (Numerically Intergrated Elements for System Analysis) is used [11].

The NISA program incorporates a variety of isoparametric elements in its element library. The element relations are

developed using displacement models. The basic idea of the finite element technique is to express the displacement or deformation of a body in terms of known functions and displacements at predetermined points on the body. These points are selected as a systematic network of grid points called nodal points and their displacements are called nodal displacements. Therefore, in finite element techniques the displacements at all points of system are known once the nodal displacements have been determined. The nodal displacements are calculated from a system of algebraic equilibrium equations for the entire structure.

Let {U} be a vector of independent generalized displacements and {P} a cooresponding vector of equivalent nodal forces for the entire structure. Then the equilibrium equation in term of displacements for the entire structure is given as

$$[K]\{U\} = \{P\} \qquad (2.17)$$

where [K] is a structural stiffness matrix that relates nodal displacements and nodal forces. If the stiffness matrix for each element of the structure is known, then the stiffness matrix for the entire structure may be obtained by the assemblage of the element matrices.

Rearranging Eq. 2.17 we have

$$\{U\} = [K]^{-1} \{P\} \qquad (2.18)$$

For structural analysis, the quantities so obtained will be nodal displacements. Stresses for each element of the system can be calculated using these computed displacements.

The basic relations for an element can be expressed as follows:

$\{u\}$ displacement field for a finite element

$\{\delta\}$ nodal displacement vector

$\{\epsilon\}$ strain vector

$\{\sigma\}$ stress vector

$[E]$ elasticity matrix

$[N]$ shape function matrix

$[K_e]$ element stiffness matrix

$[B]$ relates the nodal displacement vector and the strain vector, that is obtained from differentiation of terms in matrix $[N]$

then

$$\{u\} = [N] \{\delta\}$$

$$\{\epsilon\} = [B] \{\delta\}$$

$$\{\sigma\} = [E] \{\epsilon\}$$

$$[K_e] = \int_V [B]^T [D] [B] \, dV$$

The NISA code can analyze a wide spectrum of problems encountered in the engineering mechanics field, like static

analysis, dynamic analysis ,etc.

The finite element model created for the static analysis can be used for a dynamic analysis with minor modification and vice versa.

Input data in NISA is in a free format. The input data deck for the static analysis of this three-bar truss is shown as follows:

```
STATIC
*A1
0 STRUCTURAL OPTIMIZATION TEST PROBLEM (3 BAR TRUSS)
*B1
1,3 $ 0,0,3,0,5
*B2
3,4,4,2,6
*C1
1,12,1
*D1
   1.600000,     0.200000,     0.200000,     0.400000
0
0,0,0,0,1,0,     0.713517,0.
   0.070686      0.000398      0.000398      0.000795
0
0,0,0,0,1,0,     0.150000,0.
   0.070686      0.000398      0.000398      0.000795
0
0,0,0,0,1,0,     0.150000,0.
*E1
1,2 $ 1,1,1
1,3 $ 1,1,2
1,4 $ 1,1,3
*F1
1,0 $ 0.,-100.,0.
2,0 $ -100.,0.,0.
3,0 $ 0.,0.,0.
4,0 $ 100.,0.,0.
*H1
EX,1,0,2.07E7
NUXY,1,0,0.3
```

```
DENS,1,0,.15
*I3
1 $ 0,0,3,0 $$ 1
*I5
1,1,0,NOD
1,3,1,ELE
*J1
2,UX,0,4,1,UX,UZ,ROTX,ROTY
1,UZ,0 $ ROTX,ROTY
*M1
1,FX,-28300.
1,FY,28300.
*Z1
```

The first line is the analysis ID. The group *B1 is used to control selective loading, analysis logic and the reading of other data groups. The data group *B2 defines the total number of elements, total number of nodes, etc. *C1 selects the elements from the NISA element library. In the *D1 data group, we assign quantities of cross sectional properties. *E1 data cards define the element connectiveties. The data groups *F1, *H1, *J1, and *M1 define the nodal coordinates, material properties, boundary conditions,and concentrated nodal forces respectively.

2-1-3 Coupling Finite Element And Optimization

In order that NISA or any finite element code can become an integral part of the optimization procedure, OPT and NISA must interact with one another. To do so, it is necessary to link NISA with OPT. When structures (like trusses, frames, etc) are analyzed using existing finite element programs, we first prepare the required structural data, as shown in the previeous subsection.

Next we input this data into the computer and let the program run. Then, the finite element method program computes all quantities necessary for the design phase directly following the step of structural analysis. With the conventional use of finite element programs pre- and postprocessors have been introduced, in order to save valuable time in data preparation and evalution [8], [14], [15]. The optimizer communicates with the analysis processor through a pre-processor and the analysis processor supplies the analysis information to the optimizer through a post-processor. The function of the pre-processor is to convert the variables of the optimization process to a set of input parameters written in the format required by the analysis processor. For structural optimization, these parameters are the structural member sizes such as cross-sectional area, moment of inertia, and nodal point coordinate data. These quantities are the actual physical design variables and are seldom in a one-to-one direct equivalence relationship with the variables of the optimization process. Thus, in a typical application, the conversion within the pre-processor is not only limited to format changes but also must include such commonly used techniques as variable linking, conversion from radius information to cross-sectional area,etc. The function of the post-processor is to compute the constraints, and their gradients, if required, and to provide them in the format required by the optimizer. To do so, the post-processor extracts the pertinent behavior variables from the analysis output and compares them with the allowable value of these quantities in the equations

specific to the design problem at hand.

The subroutine NISOPT was written to generate the input data for a NISA analysis, and to drive NISA. In the optimization process with OPT, in order to find the minimum, the design variables must be updated before each evaluation. NISOPT must alter the data for NISA corresponding to design variables which are being changed. For the example under consideration, the design variables are the element radii (the size of the transverse dimensions of the elements). Thus in the input data, only the data group *D1 must be changed. The *D1 group defines the cross sectional properties (cross-sectional area, moment of inertia). The other data groups remain fixed. In the subroutine NISOPT, we create a new data file which is constructed from the data file which has been generated previously. When reading the *D1 data group, the new variables are used instead of the old ones. This data file is then used to drive the NISA program. If the geometry of the structure is also to be optimized, The data group *F1 will be modified in addition to the *D1 group.

Each time NISA is run, it creates an output file. But this output file has a fixed format, so it can't be accessed by the optimizer. We use a subroutine called FILEH to condense NISA's output file to a uniformly formated file that can be called by subroutine CONST which calculates the constraint functions.

The output file from subroutine FILEH for the three-bar truss example is shown in Table 2.2.

Table 2.2 Output File From FILEH

```
       1          1
-0.25454E+5 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1
-0.23880E+0 0.73063E-2 0.00000E+1 0.00000E+1 0.00000E+1-0.12571E-2
       1          2
-0.25453E+5 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1
 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1-0.12524E-2
       2          1
-0.14071E+4 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1
-0.23880E+0 0.73063E-2 0.00000E+1 0.00000E+1 0.00000E+1-0.12571E-2
       2          3
-0.15124E+4 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1
 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1-0.29535E-2
       3          1
 0.23922E+5 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1
-0.23880E+0 0.73063E-2 0.00000E+1 0.00000E+1 0.00000E+1-0.12571E-2
       3          4
 0.23921E+5 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1
 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1 0.00000E+1-0.12767E-2
```

The first line defines the element and node number. The second line contains the normal and shear stresses. The third line contains displacements in the X, Y, and Z-directions and rotations about the X, Y, and Z-directions respectively.

During each iteration of the GRG method, finite element solutions are required every time the constraint subroutine 'CONST' is called. After each analysis is complete, subroutine CONST must retrieve the results from FILEH which serve to constrain the design. The flowchart of subroutine "CONST" is given

in Figure 2.3.

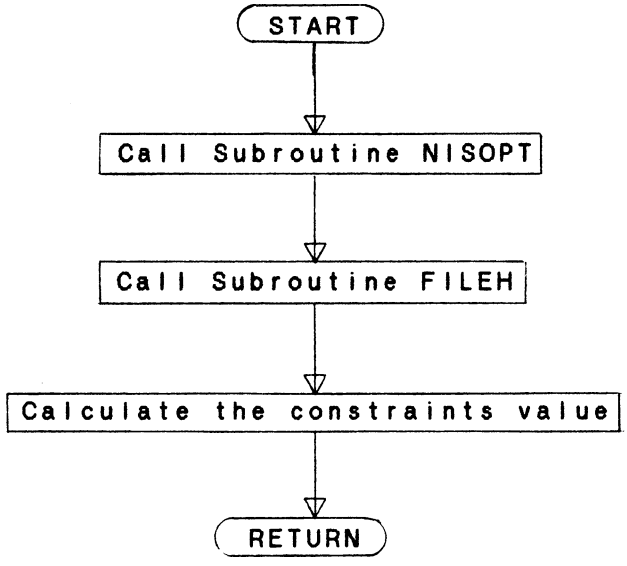


Figure 2.3 Flowchart of subroutine CONST

2-1-4 Numerical Results

The optimization problem of the three-bar truss is mathematically formulated as follows:

$$\text{Min} \quad F(X)=\sum_{j=1}^{3} L_j \cdot (X_j \cdot X_j)\pi$$

Subject to

$$d_i^a - |d_i(X)| \geq 0 \qquad i=1,\ldots\ldots,m_d$$

$$\sigma_i^a - |\sigma_i(X)| \geq 0 \qquad i=1,\ldots\ldots,m\sigma$$

$$b_i(X) \geq 0 \qquad i=1,\ldots\ldots,m_b$$

$$\beta-\beta_o \geq 0$$

$$X_j^l \leq X_j \leq X_j^u \qquad j=1,2,3$$

where

X=design variables

F(X)= the volume of the structure

$d_i(X)$= the displacement of a given node in a given direction under a given load condition.

$\sigma_i(X)$= the relevant stress in a given element under a given load condition.

$b_i(X)\geq 0$ a buckling constraint in a given element under a given load condition.

X_j^l, X_j^u = explicit lower and upper bounds on the variables.

d_i^a, σ_i^a = largest permitted values of $d_i(X)$ and $\sigma_i(X)$.

β_o = specified lower bound on eigenvalue related to the natural frequency.

Optimum results for this example are given in Table 2.3. For the solution considering stress constraints only, the optimum volume is 239.338 in^3, and three constraints are active at the optimum design. These are the stress constraint for element 1, and minimum-size constraints for elements 2 and 3. For the case with stress and displacement constraints, the optimum volume is 273.626 in^3. The displacement constraint at node 1 and the minimum-size constraint for element 3, are active at the optimum. For the case with stress, displacement, and buckling constraints, the optimum volume is 1482.59 in^3. The buckling constraints for all three elements are active at the optimum for this case. When all the constraints (stress, displacement, buckling, and natural

frequency) are imposed, the optimum volume is 1507.36 in^3. The set of active constraints include buckling in elements 1 and 2, and the frequency.

Table 2.3 Optimum three-bar truss

Element Numbers	Optimum radius, in			
	I	II	III	IV
1	0.70731	0.75928	1.44664	1.44125
2	0.15000	0.15448	0.81000	0.81652
3	0.15000	0.15000	0.88336	0.91286
At optimum Volume, in^3	239.338	273.626	1482.59	1507.36

where
I : with only stress constraints
II : with stress and displacement constraints
III : with stress, displacement, and buckling constraints
IV : with all constraints

Table 2.4 Design Data for three-bar truss (circular tube)

Lower limit on mean radius=0.15 in
Lower limit on thickness=0.075 in
Upper limit on mean radius=1.1 in
Upper limit on thickness=0.8 in

The next example is also a three-bar truss, but the cross sectional area was allowed to become a circular tube instead of a solid bar. Therefore, the design variables are the cross-sectional dimensions of each element (radius and thickness). The design data and optimum results for this example are given in Tables 2.4, 2.5 and 2.6.

The design variable numbers 1, 2, and 3 are radii (mean radii) of element numbers 1, 2, and 3, respectively. The design variable numbers 4, 5, and 6 are the thickness of element numbers 1, 2, and 3.

Because the stiffness matrix of the truss is only related to the area of the element, the optimum volume for the circular tube and solid circular cross section is same. The design, however, is different in that the optimal variables are different.

Table 2.5 Results for three-bar truss (circular tube)

	Optimum design, in	
Design Variable	Stress constraints	Stress and displacement constraints
1	0.931080	0.804796
2	0.150000	0.159251
3	0.150000	0.150000
4	0.268659	0.358150
5	0.075000	0.075000
6	0.075000	0.075074
At optimum Volume, in^3	239.3376	273.6318

Table 2.6 Critical Constraints at Optimum

With stress constraints: Lower limit on design variable numbers 2, 3, 5, and 6; and stress constraints for element 1.
With stress and displacement constraints: Displacement in the X and Y-direction at node 1; lower limit on design variable numbers 2 and 5.

2-2 Approximation Techniques

As was discussed in the previous section, we consider the design task to be automated structural design generated by combining a finite element structural analysis method with the GRG nonlinear programming method. However, this approach can create difficulties which result from too many analyses which result in a long computing run time. In order to reduce the number of detailed analyses required during the optimization, while still maintaining the salient features of the design problem, what we term approximation techniques will be used.

2-2-1 Active Constraint Logic

In structural optimization problems, there are a great many constraints that must be considered. For example, when designing an aircraft wing, the analysis model is a finite-element idealization composed of hundreds of elements. Therefore, if we require that the stress in each element never exceeds specified limits, it is clear that hundreds of constraints must be included in the optimization. However, a large percentage of these constraints may never become active during the design optimization process and so could be excluded from the constraint set. The concept of constraint deletion is to identify constraints which are far from being active and simply omit these from consideration, at least for a portion of the design process. We

can't, in general, identify nonactive constraints at the beginning of the optimization and delete them entirely because they may become active as the design progresses. Therefore, we will only temporarily delete constraints from consideration, allowing them to be included later if necessary.

At every iteration, we may delete any constraints whose value is more than some cutoff value g_o , then , proceed with the optimization including only the retained constraints. If we are using approximation techniques (discussed in next subsection), we must update the set of retained constraints after each approximate optimization. From this discussion, we see that constraint deletion can conserve computer storage, reduce computational effort, and enhance the practical use of optimization techniques.

2-2-2 Linearized Constraints

In the previous subsection, we considered ways of improving the design optimization procedure using constraint deletion. Whenever an analysis was called for, a complete, detailed finite element analysis was performed. These evaluations are generally very expensive. Here we wish to look at ways in which the analysis itself can be simplified. Linearizing the constraints through the use of approximation creates a simplified analysis model of the original complicated analysis problem. These approximations are processed using OPT, instead of original problem.

Following the approximation concept described above, the constraints given by Eq. 2.2 are approximated by introducing a first-order Taylor series expansion.

The expansion takes the form of

$$G_j = G_j(x^t) + \nabla G_j(x^t)(x - x^t) \qquad (2.19)$$

at the specified design point X^t.

Using the explicit form of the objective function and the now explicit form of the constraint functions which result from the Taylor series approximation, the function, constraints, and gradient evaluations required during the optimization process are both simple and fast. Because the first-order Taylor series expansion is used to approximate the original behavior constraints, during the optimization process, all the constraints are linear. Thus, linear constraints will always be satisfied automatically, since a linear approximation to a linear constraint will be the constraint itself. The Newton iteration which maintains feasibility for nonlinear constraints will not be required.

The subroutine 'CONST' now contains the linear functions. The original subroutine 'CONST', as shown in Fig. 2.3, changes name to

'FINIT', which is used to obtain the first and second terms for the right-hand side of Eq. 2.19.

Now, the approximate method is used to design the three-bar truss, shown in Fig. 2.1. The following four problems were solved:

Case 1: Stress constrained

Case 2: Stress and displacement constrained

Case 3: Stress, displacement and buckling constrained

Case 4: All constrained (stress, displacement, buckling and natural frequency.)

The iteration histories for these four problems are summarized in Tables 2.7, 2.8, 2.9 and 2.10. Constraint deletion was not used in this example.

It should be recongized that the linearization is in most instances a very gross approximation, and must be used with great caution [12]. One way to ensure the approximation is adequate is

Table 2.7 Iteration history for Case 1 problem

Stage	X_1	X_2	X_3	Volume (in^3)
1	0.8000	0.5000	0.5000	473.9560
2	0.6743	0.1500	0.1500	219.0564
3	0.7070	0.1500	0.1500	239.1693
4	0.7073	0.1500	0.1500	239.3542

Table 2.8 Iteration history for Case 2 problem

Stage	X_1	X_2	X_3	Volume (in^3)
1	0.80000	0.50000	0.50000	473.9560
2	0.75597	0.17352	0.15000	273.3644
3	0.75920	0.15193	0.15019	273.3448
4	0.75929	0.15433	0.15000	273.6201
5	0.75927	0.15451	0.15003	273.6254

Table 2.9 Iteration history for Case 3 problem

Stage	X_1	X_2	X_3	Volume (in^3)
1	2.00000	2.00000	2.00000	4810.939
2	1.61272	1.53053	1.54318	2949.482
3	1.43723	1.20551	1.23686	2053.978
4	1.42575	0.99071	1.04101	1692.961
5	1.44135	0.86584	0.93046	1543.172
6	1.44595	0.81754	0.88947	1490.379
7	1.44662	0.81022	0.88350	1482.792
8	1.44663	0.81006	0.88337	1482.632

Table 2.10 Iteration history for Case 4 problem

Stage	X_1	X_2	X_3	Volume (in^3)
1	2.00000	2.00000	2.00000	4810.939
2	1.61272	1.53053	1.54318	2949.482
3	1.43723	1.20551	1.23686	2053.977
4	1.42575	0.99071	1.04101	1692.960
5	1.44135	0.86584	0.93046	1543.171
6	1.44477	0.82139	0.91013	1507.370
7	1.44499	0.81662	0.91280	1507.360

to impose limits on the allowable increments in the variables. That is, for each subproblem constructed around a base point X^t, we impose the bounds

$$-\delta_i \leq \bar{X}_i^t - X_i^t \leq \delta_i \qquad i=1,\ldots,N \qquad (2.20)$$

where δ_i is some suitably chosen positive step-size parameter.

To solve a problem we first choose a starting point X^t and linearize the constraints in the neighborhood of X^t. We solve the approximate problem of Eqs. 2.1, 2.19, and 2.20 and redefine X^t as the optimum solution to the preceding subproblem. If either a worse rather than an improved value of objective function is obtained or the actual constraints are further outside the feasible region than its predecessor, we may reduce the value of the bounds δ's, and continue. For computational efficiency it is desirable to choose large values for these bounds, so that the imposed limits will not slow convergence to the vicinity of the optimum. Convergence is assumed if the change in objective function for two successive approximate subproblems is smaller than a prescribed value and the nonlinear constraints are satisfied within a prescribed tolerance.

The approximation concept consists of the following fundamental procedures:

A) Construct a first-order Taylor series expansion of the response quantities around the current design point.

B) Create an approximate optimization problem by substituting the expansion in (A) into the behavioural constraints.

C) Select pontentially critical constraints in order to temporarily remove all unnecessary constraints.

D) Solve the approximate problem with OPT.

E) Repeat steps (A) to (D) until convergence is attained.

The comparison between the results of the conventional nonlinear programming method and approximate method are shown in Tables 2.11, 2.12, 2.13 and 2.14.

Table 2.11 With only stress constraints

Element Number	Final radius, in	
	Non.	Appro.
1	0.707312	0.707338
2	0.150000	0.150000
3	0.150000	0.150000
Volume, in^3	239.3397	239.3542
Detailed structural analysis number	71	12

Table 2.12 With stress and displacement constraints

Element Number	Final radius, in	
	Non.	Appro.
1	0.759276	0.759266
2	0.154485	0.154508
3	0.150000	0.150029
Volume, in^3	273.6263	273.6254
Detailed structural analysis number	232	16

Table 2.13 With stress, displacement, and buckling constraints

Element Number	Final radius, in	
	Non.	Appro.
1	1.446638	1.446632
2	0.809995	0.810059
3	0.883357	0.883376
Volume, in^3	1482.592	1482.632
Detailed structural analysis number	125	28

Table 2.14 With all constraints

Element Number	Final radius, in	
	Non.	Appro.
1	1.444999	1.444991
2	0.816524	0.816617
3	0.912858	0.912804
Volume, in^3	1507.366	1507.360
Detailed structural analysis number	180	24

where

Non. : nonlinear programming method
Appro. : approximate method

From the above Tables, we can see that the approximate method provides similar results but greatly improves the efficiency of the optimization process. It can be noted that the savings in CPU time due to the use of approximate methods is 83.1 percent, in Table 2.11, 93.1 percent, in Table 2.12, 77.6 percent, in Table 2.13, and 86.6 percent, in Table 2.14.

Because a sensitivity analysis capability is not currently available in NISA, even this method proves to be expensive computationally. This is especially true when the number of design variables is large. This is due to the fact that the derivative information must be obtained by a forward difference approach. This means running the finite element analysis for the nominal value of a design variable and repeating the run for a slightly perturbed value of that design variable. We will discuss how to calculate such a derivative (also called sensitivity) directly later. So far, however, such algorithms seem to have limitations and may not be suitable for general finite element programs.

2-3 Multiple loading conditions

In reality, a mechanical system must operate over a range of loading conditions. For example, applied loads on a structure may act over a range of positions and angular orientations. At every point where stress and displacement constraints are imposed, these constraints must be satisfied for every load condition. Therefore, it is necessary to develop an optimization technique which will allow multiple load cases to be applied to the structure.

To demonstrate the implementation of muliple load cases, we

Table 2.15 Load conditions for three-bar truss

Load Condition	Node Number	Load component (lbf) in direction		
		F_x	F_y	F_z
1	1	-28300	28300	0.
2	1	-15315.85	36975.73	0.
3	1	0	40022.24	0.
4	1	15315.85	36975.73	0.
5	1	28300	28300	0.

will again consider the three-bar truss problem. For the truss shown in Fig. 2.1, we will determine the radius of each element in order to minimize the volume of the structure, under constraints that must be satisfied for every load condition. Assume that five load conditions, as shown in table 2.15, are applied to the truss. When each of these loads is applied to the truss, the truss will be subject to different stresses, displacements and any other responce quantities (except natural frequency). Because the truss must function under each specified load case without failing , it is necessary to choose the worst case to constrain the design. Thus the i-th constraint $G_i \geq 0$ may be expressed as

$$G_i = \min \{G_{i,r}(X)\} \geq 0, \quad r=1 \text{ to NLC} \qquad (2.21)$$

where NLC is the number of loading conditions. Then 'active constraint logic' can be used in order to eliminate redundant constraints.

For each of these load cases applied to the truss, only the

magnititude and direction of the loads or points of application are different. The element dimensions and boundary conditions are the same. When a call is made to the finite element program (NISA), the decomposed stiffness matrix from the first load case can be reused for the other load cases, so that significant recomputation can be avoided. This is accomplished by using the restart capability of NISA.

Table 2.16 Optimum Design for Three-Bar Truss under multiple loading conditions

Design Variable	With Only Stress Constraints		With Stress and Displacement Constraints	
	Starting Values	Final Values	Starting Values	Final Values
1	1.0000	0.8384	1.0000	0.8977
2	0.8000	0.6953	0.8000	0.6916
3	1.0000	0.8343	1.0000	0.8974
4	0.5000	0.2378	0.5000	0.2856
5	0.4000	0.1647	0.4000	0.1515
6	0.5000	0.2363	0.5000	0.2858
Volume, in^3	1089.64	424.5356	1089.64	521.5534

Table 2.16 shows the results of the optimization of the three-bar truss problem under five load conditions. The results show that with only stress constraints, the three stress constraints for elements 1, 2, and 3 become tight for load cases 1, 3, and 5 respectively. For the case with stress and displacement constraints, the displacement constraint in the X-direction is tight for load case 1.

The displacements at node 1 are restricted to be within ±0.12 inch both in the X and Y-direction. The design variables are the same as those listed in Table 2.4. Design variable numbers 1, 2, and 3 are mean radii, and design variable numbers 4, 5, and 6 are the thicknesses, for elements 1, 2, and 3 respectively.

On comparing the optimum volume obtained for this three-bar truss under multiple loading conditions with those under the single loading condition, it is evident that the former is much greater than the latter. This is not unexpected as the design in the former case has to be such that it can function under each of the five assumed loading cases, while under the single loading condition, only one load needs to be considered.

2-4 Optimization of Geometry

In the above discussion, the geometry of the structure is held fixed, i.e. the coordinates of the nodes are not allowed to change. What remains to be optimized are the sizes of the transverse dimensions of the elements, e.g. cross sectional areas of the bars and the thickness of membrane plates. In this section the optimal design of the geometry of the structure is discussed, i.e. it is not assumed that the geometry of the structure is fixed. Thus, the design variables consist both of element sizes and geometric variables. Geometric variables may represent the

coordinates of nodes, or some arbitrary curve in space along which the nodes may be placed.

Using the finite element method of analysis, the optimal design problem of geometry and element sizes can be formulated as follows [see Eqs. 2.1 through 2.3] : Find the member sizes X, and node coordinates C, such that

$$\text{objective function } F(X,C) \text{ -----> min} \qquad (2.22)$$

subject to

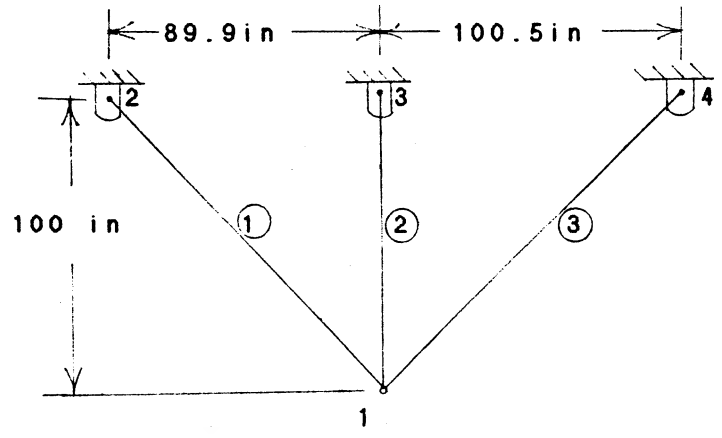
$$G_j(X,C) \geq 0 \quad j=1,2,\ldots\ldots,m \qquad (2.23)$$

$$X^l \leq X \leq X^u \qquad (2.24)$$

$$C^l \leq C \leq C^u \qquad (2.25)$$

Eqs. 2.24 and 2.25 are bounds on the design variables X and C. Eq. 2.23 represent stress, displacement, etc, constraints. The problem given by Eqs. 2.22 through 2.25 can be solved by the approximate method using a procedure similar to that discussed in section 2-2 for problems with only member sizes as variables. However, in the present case derivatives must be computed with respect to both X and C.

Consider the three-bar truss shown in Fig. 2.1. The geometrical variables are the coordinates of nodes 2, and 4 in the X direction. The element size variables are the radius of each element.



$X_1 = 0.71438$ in
$X_2 = 0.15000$ in
$X_3 = 0.16493$ in

$V = 234.784$ in^3

Figure 2.4 Optimal layout

The optimal layout and the corresponding minimum volume are shown in Fig. 2.4. In order to investigate the effect of changes in geometry, a comparision is made between the results of the fixed geometry and variable geometry problem in Table 2.17.

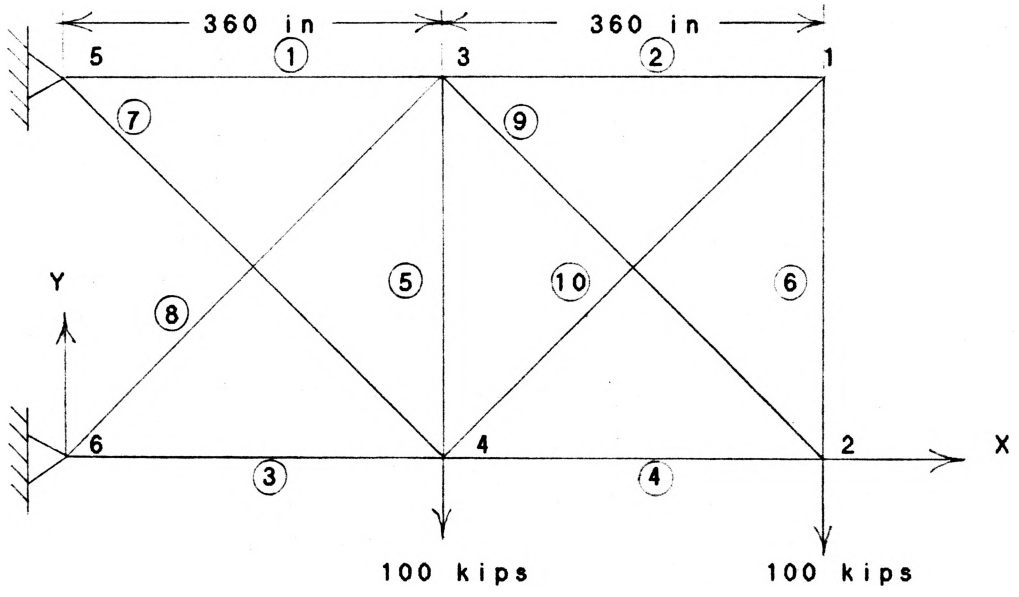
Table 2.17 Comparsion of results, three-bar truss

Case	Minimum Volume, in^3
Fixed Geometry	239.3379
Variable Geometry	234.7842

The variable geometry problem simultaneously considered the size of the transverse dimensions of the elements and the node coordinates, so it must get an equal or better result than that from the fixed geometry problem.

2-5 Ten-Member Cantilever Truss

A ten-member cantilever truss is shown in Figure 2.5. This problem has been used in the literature [5,7,8] to compare various techniques of optimal design. The problem is to find the cross-sectional area of each member of the truss in order to minimize its weight, subject to stress, displacement, and member size constraints. Design data for this problem is given in Figure 2.5.



Design Data :

Modulus of elasticity = 10^4 ksi
Material density = 0.1 lb/in^3
Lower limit on area = 0.1 in^2
Displacement limit = 2.0 in
Stress limit = 25 ksi

Figure 2.5 Ten-member Cantilever Truss

Table 2.18 presents a comparison of the optimum solution obtained using the algorithm of this chapter with results available in literature. The optimum weight found is 5056.473 lb. The downward displacement constraints at nodes 1 and 2, the stress constraint for member 5 and mimimum-size constraints on members 2, 5, and 10 are active at the optimum.

Table 2.18 Comparison of final designs for ten-bar truss

Member No.	Final cross-sectional area (in^2)				
	Ref. 5	Ref. 7	Ref. 8	Ref. 8	NISOPT
1	30.031	30.463	30.730	30.521	30.352
2	0.100	0.100	0.100	0.100	0.100
3	23.274	23.848	23.941	23.200	23.031
4	15.286	15.133	14.733	15.223	15.203
5	0.100	0.100	0.100	0.100	0.100
6	0.557	0.515	0.100	0.551	0.570
7	7.468	7.436	8.341	7.457	7.282
8	21.198	21.204	20.951	21.036	20.636
9	21.618	21.082	20.836	21.528	22.259
10	0.100	0.100	0.100	0.100	0.100
Weight	5061.6	5062.2	5076.7	5060.8	5056.5

From the comparison made with the other available programs in this example, it is shown that NISOPT provides the least weight design.

3. PRACTICAL EXAMPLE

We now want to turn our attention to the application of the formulation described in Chapter 2 to the solution of an actual seat frame structural design problem. This is a space frame problem. The vehicle seat frame considered herein is assumed to have a preassigned geometric configuration and is constructed of a fixed structural material. This structure, shown in Figures 3.2-3.5, is idealized using 71 beam elements and 60 nodes. The total number of degrees of freedom is 351. Design variable linking is employed, which limits specified groups of finite elements to be the same type with identical design variables.

The following grouping of elements, as shown in Table 3.1, where elements of each group are required to have the same cross-sectional area, is used to maintain symmetry in the structure.

Table 3.1 The element group of seat frame

Group No.	Element Number
1	1,2
2	3-8,46-51
3	9-15,39-45
4	22-29
5	52-57,62-67
6	58-61
7	68-71
8	16-21,30-38

In this example both hollow circular and rectangular sections are considered. The hollow sections are useful due to their efficiency, especially when weight and appearance become important design considerations, or when weight savings may be a substantial economical consideration.

The element groups of 1, 6, and 8 have rectangular cross sections and the others have cross circular sections. In the optimization all element cross-sectional dimensions including widths, heights, mean radii and thicknesses are considered to be design variables, which are 19 in number. Design data for this frame is given in Table 3.2.

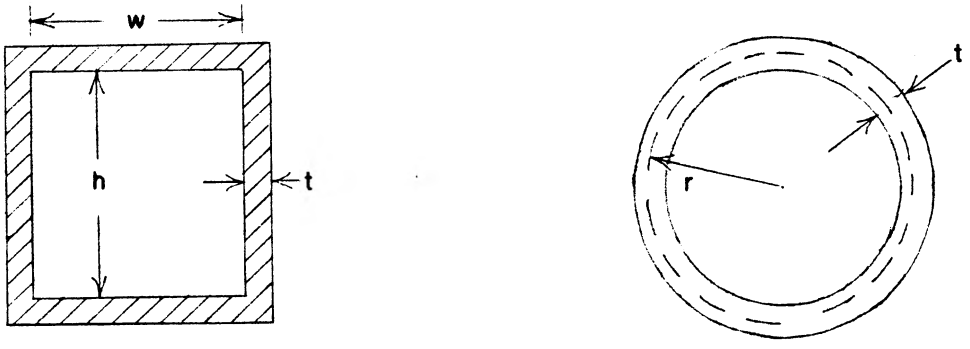


Figure 3.1 Rectangular and circular cross sections

The objective function is taken as the total volume of the frame which is expressed as

$$F = \sum_{i=1}^{NG} V_i$$

For circular section $V_i = 2\pi \cdot L_i \cdot R_i \cdot t_i$

For rectangular section $V_i = L_i \cdot ((H_i + t_i)(W_i + t_i) - H_i \cdot W_i)$

where

L_i = length of the i-th element group

t_i = thickness of the elements in the i-th group

R_i = mean radius of the elements in the i-th group (circular)

H_i = inner height of the elements of the i-th group (rectangular)

W_i = inner width of the elements in the i-th group (rectangular)

NG= total number of element groups

Table 3.2 Design Data for Seat Frame

Modulus of elasticity $=2 \cdot 10^4$ ksi
Poisson's ratio = 0.3
Lower limit on mean radius =0.0866 in
Lower limit on thickness =0.0472 in
Displacement limit at node
48 in X-direction = 0.7874 in
Stress limit = 25 ksi

The frame is designed for two cases :

Case 1 : The structure is optimized under stress constraints. Loads of 326.9 and 138.8 LBF in the positive X-direction and negative Z-direction are applied at node 48.

Case 2 : The frame is optimized under stress and displacement constraints. There are four loading conditions. The first condition uses the same loads as in case 1. The second loading condition is to have a 1349 LBF normal force and 3147.3 LBF tangential force

distributed over the elements of group 2. The third condition is a load of 285 LBF in the positive X-direction applied at node 48. In the fourth condition, loads of 270 LBF in the positive X-direction and 135 LBF in negative Z-direction are applied at node 46.

The elements of a seat frame are usually subjected to an axial force, bending moment, shear force, and/or torsional moment. Thus, we need to consider the effect of combined stresses in formulating the stress constraint. Compared with the normal stress, the shear stress is generally smaller. In this example, we will neglect the shear stress, and only consider the normal stress. In general, the normal stress is a maximum at the outer edges of the beam element. In this problem all the normal stresses are calculated at the outer edges.

Table 3.3 gives the results for this example for Case 1, and Fig. 3.6 shows the variation of the cost function with respect to the iteration number. The set of active constraints at the optimum includes : minimum element thickness for element groups 1-7, minimum element mean radius for element group 4, and stress limits on elements 32, 39, and 66. It is also noted here that many thickness variables are at their lower bounds at the optimum. In many practical design problems it may be possible to fix the element thickness. This will result in further computational

savings.

Table 3.3 Results for Seat Frame Structure

Group Number	Optimum design, in			
	Mean Rad.	Height	Width	Thickness
1		0.2880	2.4534	0.0472
2	0.3333			0.0472
3	0.7166			0.0472
4	0.0866			0.0472
5	0.6291			0.0472
6		2.6869	0.7211	0.0472
7	0.2290			0.0472
8		0.8596	0.8596	0.0688
At optimum Volume, in^3		38.1322		

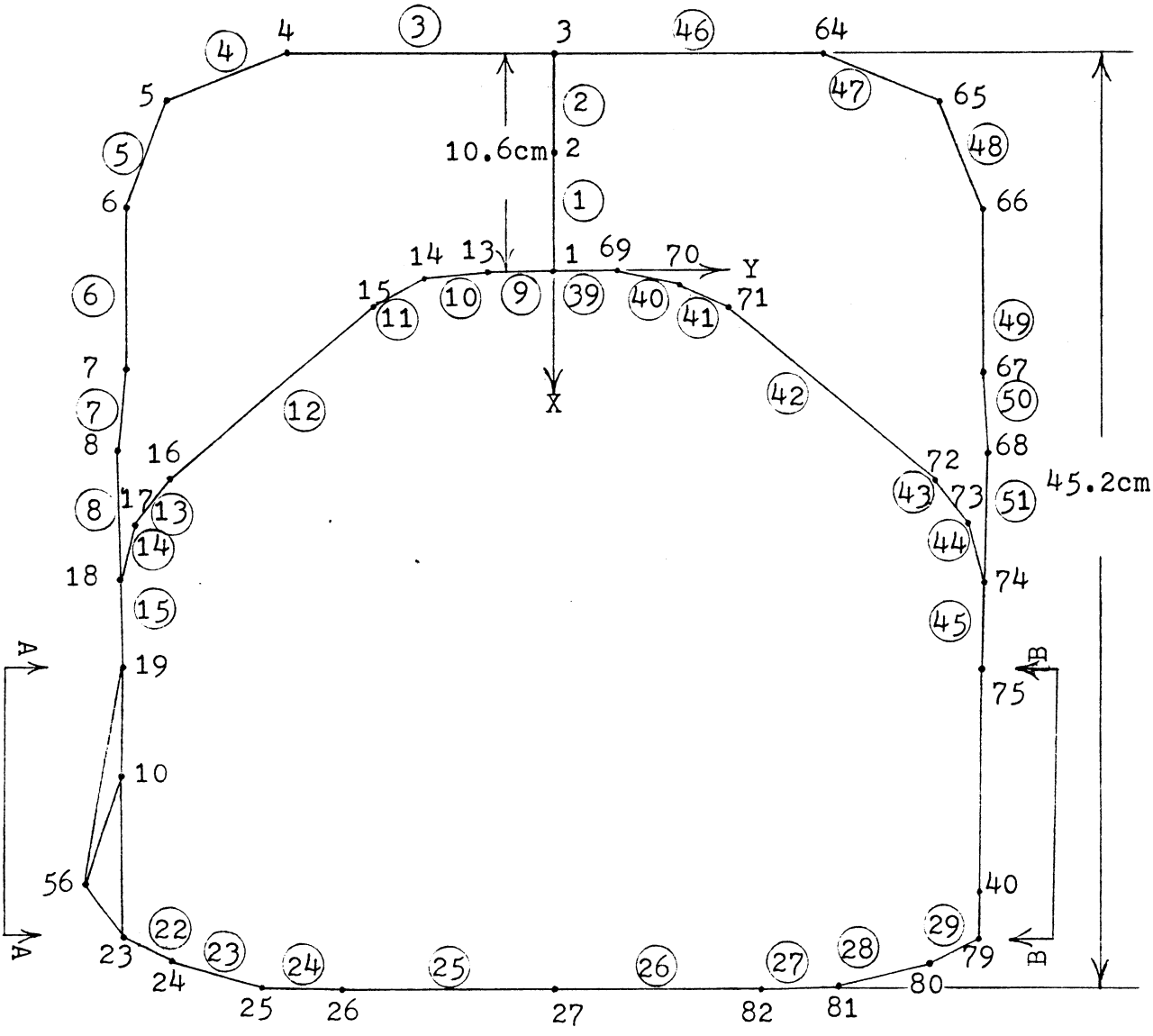
Table 3.4 Results for Seat Frame Structure

Group Number	Optimum design, in			
	Mean Rad.	Height	Width	Thickness
1		0.4102	2.5146	0.0472
2	0.6940			0.0494
3	0.7079			0.0472
4	0.0895			0.0472
5	0.5717			0.0472
6		2.7509	0.7824	0.0472
7	0.2902			0.0472
8		0.7983	0.7983	0.0802
At optimum Volume, in^3		44.0073		

For Case 2, the optimum volume is 44.0073 in^3. The stress constraints on elements 46 (for load condition 2), 66 (for load condition 1) and 32 (for load condition 4), and the minimum thickness constraint on groups 1 and 3-7 are active at the

optimum. Optimum results for this case are given in Table 3.4. Figure 3.7 shows the iteration history for this example.

As can be seen from Figures 3.6 and 3.7, the method possesses suitable convergence properties. In this example, approximation techniques worked well. In this chapter we only considered the seat frame with a given geometry. However, it is easy to extend to geometric optimization, and even to include material selection.



Top View

Figure 3-2

Front View

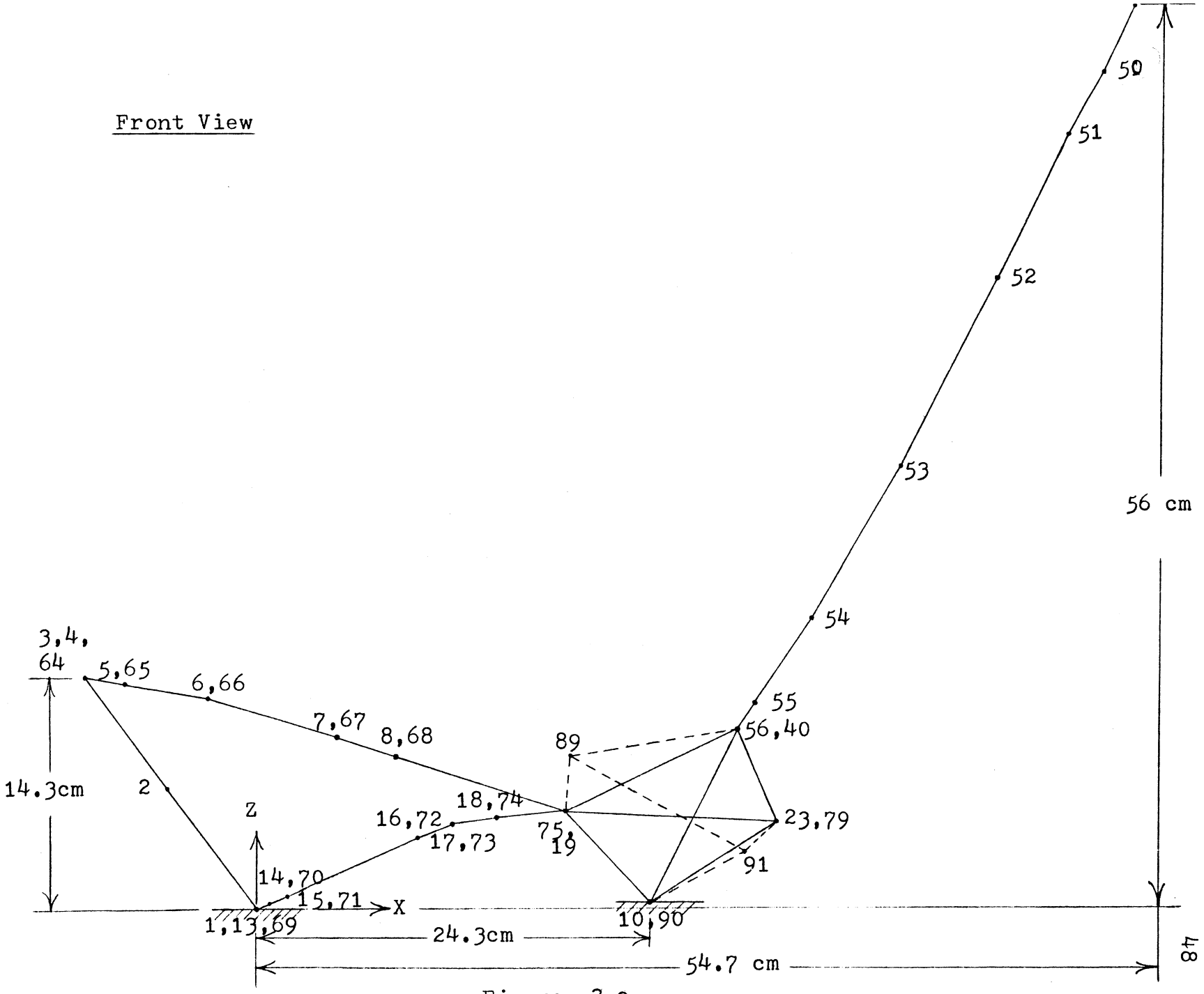
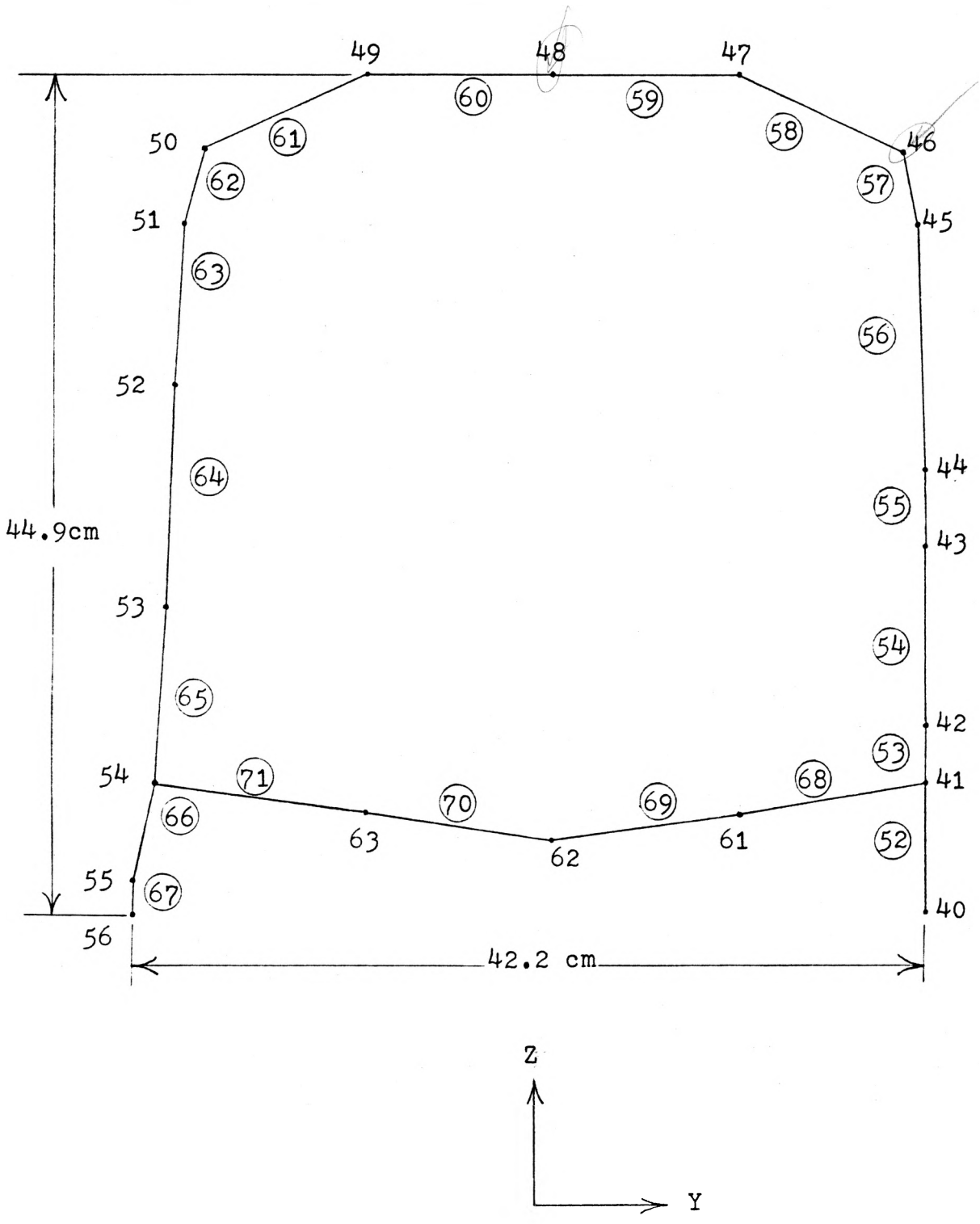
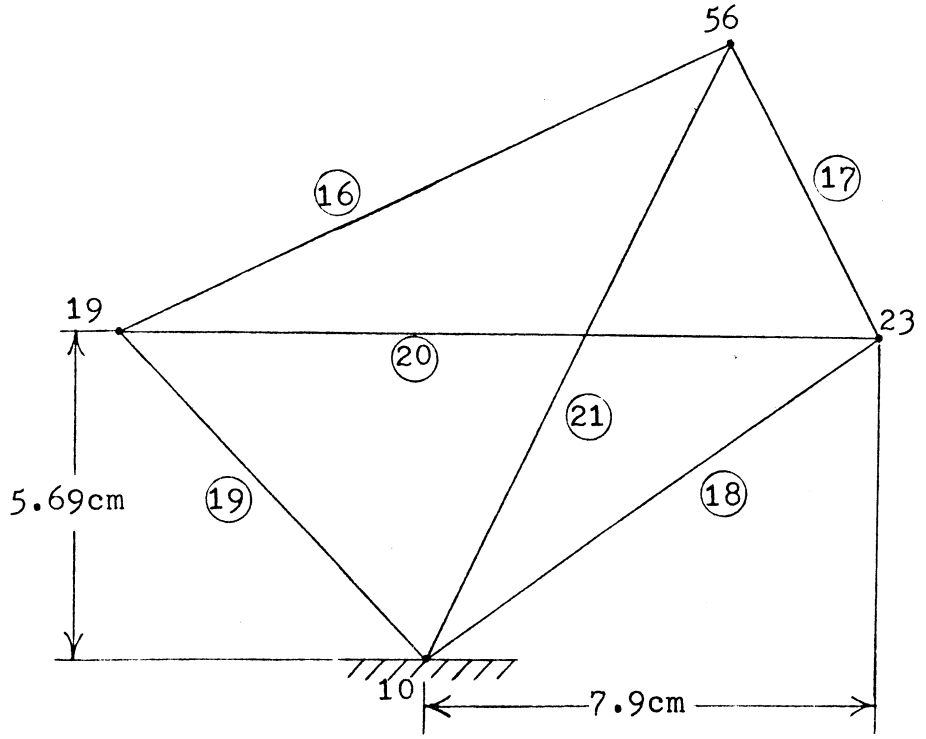


Figure 3-3

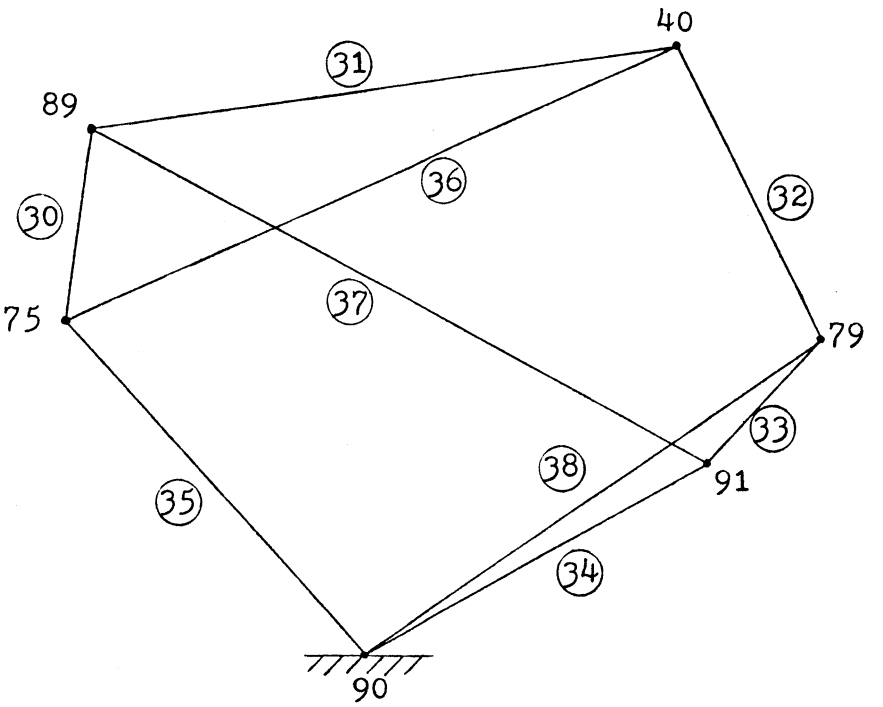


Side View

Figure 3-4



A-A View



B-B View

Figure 3-5

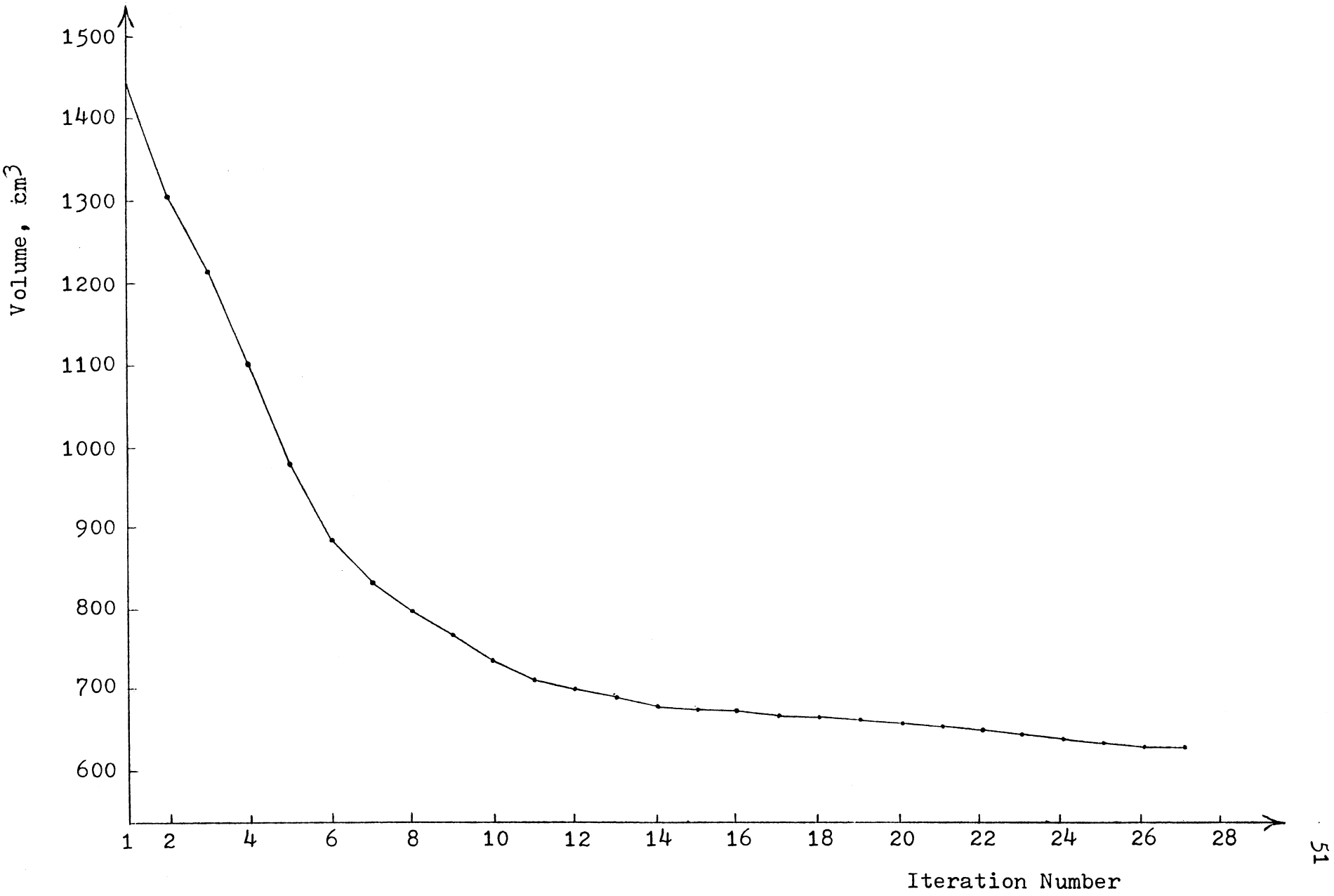

Volume, cm3
1500
1400
1300
1200
1100
1000
900
800
700
600
1
2
4
6
8
10
12
14
16
18
20
22
24
26
28
Iteration Number

Figure 3-6

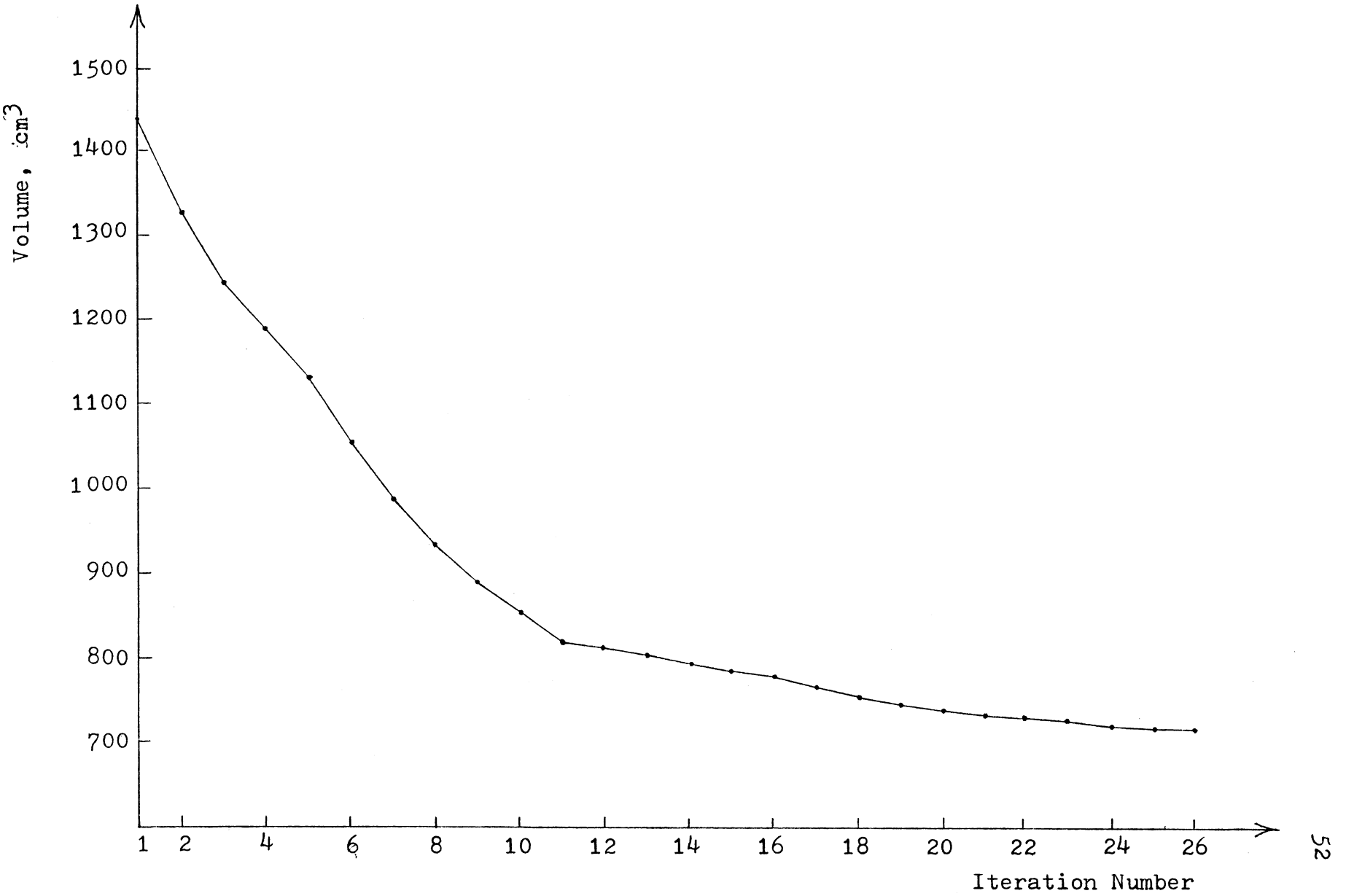

Volume, cm^3
1500
1400
1300
1200
1100
1000
900
800
700
1
2
4
6
8
10
12
14
16
18
20
22
24
26
Iteration Number

Figure 3-7

4. ENHANCEMENTS TO THE STRUCTURAL OPTIMIZATION PROCEDURE

4-1 Response Gradients

In chapters 2 and 3, we used an approximate method to solve the problems. Because a sensitivity analysis is not currently available in NISA, this method is not very efficient. Now, we want to consider some techniques which would make the derivative computations more efficient.

4-1-1 Displacement constraints

The gradient of a constraint for the nodal displacement of a structural system can be obtained by taking the partial derivative of the equilibrium equation with respect to the design variables.

$$[K]\{U\} = \{P\} \qquad (4.1)$$

Differentiating Equation 4.1 with respect to the design variable j, we get

$$\frac{d[K]}{dX_j}\{U\} + [K]\frac{d\{U\}}{dX_j} = \frac{d\{P\}}{dX_j} \qquad (4.2)$$

The gradient of {U} becomes

$$\frac{d\{U\}}{dX_j} = [K]^{-1}\left(\frac{d\{P\}}{dX_j} - \frac{d[K]}{dX_j}\{U\}\right) \qquad (4.3)$$

We will assume that {P} is independent of X, so that Equation 4.3 reduces to

$$\frac{d\{U\}}{dX_j} = -[K]^{-1}\frac{d[K]}{dX_j}\{U\} \qquad (4.4)$$

4-1-2 Stress Constraints

Let p and u represent composite vectors of element forces and displacements respectively, and k is the composite stiffness matrix for all elements of the structure. Then the equilibrium equation for all elements is compactly written in matrix form as

$$[\bar{p}]=[k]\{\bar{u}\} \qquad (4.5)$$

Now, a compatibility relationship between {u} and {U} is expressed as

$$\{\bar{u}\}=[A]\{U\} \qquad (4.6)$$

where [A] is a Boolean matrix.

Therefore

$$\frac{d\{\bar{u}\}}{dX_j} = [A]\frac{d\{U\}}{dX_j} \qquad (4.7)$$

From $\{\bar{u}\}$, we can find $\{u\}_i$, where $\{u\}_i$ is a vector of displacements for i-th element in the global coordinate system. We

now transform from a global to a local system, hence

$$\{u\}_l = [R]_l \{u\}_l \qquad (4.8)$$

where $[R]_l$ is a transformation matrix.

From chapter 2 we have shown

$$\{\epsilon\} = [B] \{u\}_l \qquad (4.9)$$

where $\{\epsilon\}$ is a vector of strains and $[B]$ is a matrix that is obtained from the differentiation of the terms in the shape function matrix. A truss element only has axial strain, hence, the matrix $[B]$ is just

$$[B] = \frac{1}{L_l} [-1 \quad 0 \quad 1 \quad 0] \qquad (4.10)$$

To express the element stress, one needs the stress-strain relationship

$$\{\sigma\} = [E] \{\epsilon\} \qquad (4.11)$$

where $\{\sigma\}$ is a vector of stresses and $[E]$ is matrix of elastic constants.

Rearranging Eqs. 4.5 through 4.11, the element stresses can be written as explicit functions of nodal displacements. So $\frac{d\{\sigma\}}{dX_j}$ is also an explicit function of $\frac{d\{U\}}{dX_j}$.

4-1-3 Frequency constraint

The frequency gradient can be obtained by taking the partial derivative of the characteristic equation with respect to the design variables

$$[K]\{y_i\} = w_i^2 [M]\{y_i\} \qquad (4.12)$$

where [M] is the mass matrix , w is the natural frequency in rad. per sec., and $\{y_i\}$ is the vibration mode. The mode is assumed to be normalized so that

$$\{y_i\}^T [M]\{y_i\} = [I] \qquad (4.13)$$

Differentiating Eq. 4.12 with respect to the design variables, gives us

$$([K]-w_i^2[M])\frac{d\{y_i\}}{dX} - \frac{d(w_i^2)}{dX}[M]\{y_i\} = -\left(\frac{d[K]}{dX} - w_i^2\frac{d[M]}{dX}\right)\{y_i\} \qquad (4.14)$$

If only the derivatives with respect to frequency are required the calculation is much simpler. Premultiplying Eq. 4.14 by $\{y_i\}^T$, we get

$$\frac{d(w_i^2)}{dX} = \{y_i\}^T \left(\frac{d[K]}{dX} - w_i^2\frac{d[M]}{dX}\right)\{y_i\}$$

The formulation for gradient computations given here is intended to present only the basic concepts. In pratice, computational efficiency may be dramatically improved by careful analysis of the matrix operations involved in relation to the number of constraints for which gradients must be computed.

Although gradient computations require considerable effort, this must be coded only once for computer solution. It can then be applied to any structure modeled by the elements for which the gradient information is available. With this information available as part of the analysis, major efficiency gains in the optimization are achieved. Thus, when developing a general purpose structural optimization program, the effort necessary to evaluate gradients is well worthwhile.

4-2 Approximate Methods

In the optimization of a structure it is often necessary to employ approximate reanalyses for intermediate designs, which requires less computational effort. In chapters 2 and 3, the Taylor series expansion has been used. There are other approximate methods which are discussed in the literature.

In Ref. [6], the reduced basis method has been used to perform approximate structural analysis. The concept of stress-ratio design (fully-stressed design) may also be incorporated in

the program. This feature generally gives a good starting point for the optimization method [5,6,8,9]. Substructuring techniques are an effective way of analyzing large and complex structures [5,9].

A combination of approximate methods may lead to better results. Therefore it is necessary to investigate methods which can use other approximation methods, in addition to the Taylor series expansion. This may reduce the number of structural analyses requiring the use of a finite element method.

5. CONCLUSION

Numerical optimization is a powerful tool for structural design. All members of the structure are sized for the most mass efficient way of carrying the prescribed loads. If the optimization of geometry is also considered, the best configuration of the structure can be simultaneously determined. This is clearly a major improvement over the trial and error design practice for determining the most weight efficient design that will adequately carry all the loads.

In the previous chapters, we have presented in detail an efficient automated minimum weight design procedure for the solution of the structural optimization problem. The examples of the three-bar and ten-bar trusses have indicated that the linear sequence of the explicit approximate method which is used to replace the original problem, is a powerful and a rather general approach to structural optimization. The approximate methods include temporary deletion of noncritical constraints, design variable linking and Taylor series expansions for response variables in terms of design variables. The subroutines NISOPT and FILEH which are used to link NISA with OPT play an extremely important role in the programming procedure. In order to shorten the computer run time , it is necessary to include the sensitivity

analysis in the finite element code. Although the sensitivity analysis is presently unavailable in NISA, it is hoped that it will be completed in the near future and, thus, increase the overall efficiency. All the examples included in this thesis considered only truss and frame structures. Extensions can be made to include plate, shell and solid elements which would enhance the potential applications of the method.

REFERENCES

[1] Vipperla B. Venkayya, "Structural Optimization: A Review And Some Recommendations", International Journal For Numerical Methods in Engineering, 203-228, 1978.

[2] L. Berke and V. B. Venkayya, "Review of Optimality Criteria Approaches to Structural Optimization", Structural Optimization Sysposium (Ed. L. A. Schmit), ASME Winter Annual Meeting, 1974 (AMD7).

[3] L. A. Schmit, "Structural Engineering Applications of Mathematical Programming Techniques," Symposium on Structural Optimization, AGARD Conf. Proceed. No. 36, 1969.

[4] Fred Moses, "Mathematical Programming Methods For Structural Optimization", Structural Optimization Sysposium (Ed. L. A. Schmit), ASME Winter Annual Meeting, 1974 (AMD7).

[5] Edward J. Haug and Jasbir S. Arora, "Applied Optimal Design, Mechanical and Structural Systems", 1979.

[6] S. H. Wang and K. M. Ragsdell, "TRUOPT users manual", 1984.

[7] Seung J. Lee, "Applications of TRUOPT", 1984.

[8] A. J. Morris, "Foundations of Structural Optimization : A Unified Approach", Wiley-Interscience, 1982.

[9] Garret N. Vanderplaats, "Numerical Optimization Techniques for Engineering Design with Applications", McGraw-Hill Book Co., 1984.

[10] Uri Kirsch, "Optimum Structural Design, Concepts, Methods, and Applications", McGraw-Hill Book Co. 1981.

[11] "NISA User's Manual, Vol. 1", Engineering Mechanics Research Co., 1983.

[12] G. A. Gabriele and K. M. Ragsdell, "OPT Users Manual".

[13] G. V. Reklaitis, A. Ravindran and K. M. Ragsdell, "Engineering Optimization, Methods and Applications", Wiley-Interscience, 1983.

[14] James A. Bennett and Mark F. Nelson, "An Optimization Capability for Automotive Structures", General Motors Research Laboratories, 1979.

[15] Dietrich Hartmann, "Computer Aided Structural Optimization, Enbedding of the Finite Element Method into Numerical Nonlinear Optimization", ASME, 1983.

Appendix A : Description of Subroutines

NISOPT : This subroutine provides the input data for NISA, the finite element package, and to drive NISA.

FINIT : Calculate the original and accurate constraint values, and delete the unnecessary constraints.

FILEH : Condense NISA's output file to a uniformly formated file that can be called by subroutine FINIT.

PARTGR : Calculate numerical first partial derivatives of the original constraints with respect to the design variables using forward differences.

CONST : Calculate the approximate constraints which construct a first-order Taylor series expansion of the original and accurate constraints.

Appendix B : Submitting A Program to OPT

To submit a program to OPT we must supply three basic elements, (1) a function subprogram defining the objective function , (2) a subroutine defining the constraints, and (3) a calling program.

1. The Objective Function Subprogram, F(X)

The objective function subprogram returns the value of objective function given design variables X. The required format is as follows,

```
FUNCTION F(X)
DIMENSION X(1)
COMMON /ONE/ NF,NC
NF=NF+1
.
.
F=value of objective function
RETURN
END
```

The objective function in the structural optimization is the

volume or weight of the structure.

2. The Constraint Subroutine, CONST

The constraint subroutine returns the values of constraints given design variables X. The required format is as follows,

```
SUBROUTINE CONST(X,CON)
DIMENSION X(1),CON(1)
COMMON /ONE/ NF,NC
NC=NC+1
CON(1)=1st equality constraint
CON(2)=2nd equality constraint
.          .          .          .
.          .          .          .
CON(L)=Lth equality constraint
CON(L+1)=1st inequality constraint
.          .          .          .
.          .          .          .
CON(L+K)=Kth inequality constraint
RETURN
END
```

3. The Calling Program

The calling program serves two purposes, (1) initialization

of program parameters and (2) the calling of OPT.

The following arrays must be dimensioned in the calling program with the values specified

```
DIMENSION X(N),XMAX(N),XMIN(N)
COMMON /VARB/ D(NSIZE)
```

where N is the number of design variables, and NSIZE is used for array D in this program to provide them with enough storage area.

The following COMMON statement is also required.

```
COMMON/PARI/CRIT,EPS,IPR,MAXM,IDATA,NE,NI,LBD,NCON,EPSLS,EPSBD
```

The following program parameters must be specified before the call to OPT

N = number of design variables

NE = number of equality constraints

NI = number of inequality constraints

IPR = output control parameter

MAXM = maximum number of reduced gradient stages

CRIT = convergence criteria

EPSLS = line search stopping criteria

EPSBD = active constraint region

EPS = differencing parameter used in the numerical determination of first derivatives.

X(I) = starting values of design variables

XMAX(I)= upper bounds of design variables

XMIN(I)= lower bounds of design variables

After specifying the above program parameters the call to OPT can take place. The required format is,

```
CALL OPT(X,XMAX,XMIN,N)
```

Appendix C : Listing of Subroutines

1. SUBROUTINE CONST

```
      SUBROUTINE CONST(X,CON)
      DIMENSION X(1),CON(15)
      COMMON /PROX/ PAGR(15,6),COEF(15),N
      COMMON /XQQ/ XQ(6),NW,DEC
      COMMON /PARI/ CRIT,EPS,IPR,MAXM,IDATA,NE,NI,LBD,NCON,EPSLS
      COMMON /ONE/ NF,NC
      NC=NC+1
      DDW=0.
      DO 20 I=1,NI
      DDW=0.
      DO 10 J=1,N
      DDW=DDW+PAGR(I,J)*(X(J)-XQ(J))
   10 CONTINUE
      CON(I)=COEF(I)+DDW
   20 CONTINUE
      RETURN
      END
```

2. SUBROUTINE NISOPT(X) (for static analysis only)

```
      SUBROUTINE NISOPT(X)
      DIMENSION X(1)
      COMMON /IUNI/ IUN1,IUN2
      CHARACTER*1 STUFF(80)
C     OPEN INPUT FILE AND TEMPORARY OUTOPT FILE
      IUN1=52
      IUN2=53
      OPEN(UNIT=IUN1,FILE='TRUSS',STATUS='UNKNOWN',ERR=100)
      OPEN(UNIT=IUN2,FILE='TRTMP',STATUS='UNKNOWN',ERR=100)
      PI=3.1415927
      ICASE=0
      NA=3
C     READ IN DATA DECK ONE CARD IMAGE AT A TIME
    8 READ(IUN1,10) (STUFF(I),I=1,80)
   10 FORMAT(80A1)
      IF(STUFF(1).NE.'*'.OR.STUFF(2).NE.'D'.OR.STUFF(3).NE.'1')
     * GO TO 25
```

```
C       GENERATE SETS OF THREE CARD IMAGES WITH BEAM PROPERTIES
        WRITE(IUN2,10) (STUFF(I),I=1,80)
        DO 20 I=1,NA
        AREA=X(1)*X(1)*PI
        YYI=.25*PI*X(I)**4
        ZZI=YYI
        XJ=YYI+ZZI
        WRITE(IUN2,13) AREA,YYI,ZZI,XJ
   13   FORMAT(F11.6,',',F11.6,',',F11.6,',',F11.6)
        WRITE(IUN2,14)
   14   FORMAT('0')
        WRITE(IUN2,15) X(I)
   15   FORMAT('0,0,0,0,1,0,',F11.6,',0.')
C       BURN THREE EXISTING CARDS
        DO 18 J=1,3
        READ(IUN1,10) (STUFF(K),K=1,80)
   18   CONTINUE
   20   CONTINUE
        GO TO 8
   25   IF(STUFF(1).NE.'*'.OR.STUFF(2).NE.'Z'.OR.STUFF(3).NE.'1')
       * GO TO 50
        WRITE(IUN2,10) (STUFF(I),I=1,80)
        COLSE(UNIT=IUN1)
        CLOSE(UNIT=IUN2)
C       REOPEN FILES AND COPY BACK TO ORIGINAL TRUSS FILE
        OPEN(UNIT=IUN1,FILE='TRUSS',STATUS='UNKNOWN',ERR=100)
        OPEN(UNIT=IUN2,FILE='TRTMP',STATUS='UNKNOWN',ERR=110)
   30   READ(IUN2,10) (STUFF(I),I=1,80)
        WRITE(IUN1,10) (STUFF(I),I=1,80)
        IF(STUFF(1).EQ.'*'.AND.STUFF(2).EQ.'Z'.AND.STUFF(3).EQ.'1')
       * GO TO 45
        GO TO 30
   50   WRITE(IUN2,10) (STUFF(I),I=1,80)
        GO TO 18
   45   CONTINUE
        CLOSE(UNIT=IUN1)
        CLOSE(UNIT=IUN2)
        CALL NISAFF(6HTRUSS )
        OPEN(UNIT=IUN1,FILE='TRTMP',STATUS='UNKNOWN')
        WRITE(IUN1,60)
   60   FORMAT('FIXDATA',30(1H ),/,'TROUT',30(1H ),/,'0',/,'100',/,
       * 'TRUS',30(1H ),/,'10                    ',/,'26           ')
        WRITE(IUN1,65)
   65   FORMAT('33',/,'8',/,'24',/,'25',/,'99',/,'27',/,'99')
        CLOSE(UNIT=IUN1,STATUS='KEEP')
C       INITATE NISA JOB
        CALL PROG1
        RETURN
        END
```

University Libraries
University of Missouri

Digitization Information Page

Local identifier Shih1985

Source information

Format	Book
Content type	Text
Source ID	Gift copy from department; not added to MU collection.
Notes	

Capture information

Date captured	August 2024
Scanner manufacturer	Fujitsu
Scanner model	fi-7460
Scanning system software	ScandAll Pro v. 2.1.5 Premium
Optical resolution	600 dpi
Color settings	8 bit grayscale
File types	tiff
Notes	

Derivatives - Access copy

Compression	Tiff: LZW compression
Editing software	Adobe Photoshop
Resolution	600 dpi
Color	grayscale
File types	pdf created from tiffs
Notes	Images cropped, straightened, brightened