A Computational Study of Transformation Methods for Optimal Design

Ashok D. Belegundu* and Jasbir S. Arora†
The University of Iowa, Iowa City, Iowa

In this paper computational aspects of transformation methods are studied. Transformation methods, which include sequential unconstrained minimization techniques (SUMTs) and multiplier methods, are based on solving a sequence of unconstrained minimization problems. An efficient technique is given to compute gradients of the transformation function. An operations count is given to demonstrate savings of the suggested technique over two other techniques used in the literature. Computer programs implementing the use of this technique in SUMTs and multiplier algorithms are developed. Applications of these programs on a set of structural design problems are given. Multiplier methods are found to be very stable and reliable, even on some relatively difficult problems, and they perform better than SUMTs. Some ways of improving the efficiency of transformation methods are given, together with possible extensions of using the suggested approach to compute gradients of implicit functions.

I. Introduction

RECENTLY, a general framework for efficient use of transformation methods in optimal design of structural and mechanical systems was presented.[1] This paper presents a study of computational aspects of transformation methods. The term "transformation method" is used to describe any method that solves the constrained optimization problem by transforming it into one or more unconstrained optimization problems. Transformation methods include exterior and interior penalty function methods as well as augmented Lagrangian or multiplier methods.

Many of the constraint functions in optimum structural design are implicit functions of the design variables. For example, many constraints are functions of nodal displacements which are in turn dependent on the design variables. This implicit nature of the constraints makes it very expensive to calculate the functions and their gradients. Transformation methods essentially collapse all constraints of the design problem into one equivalent functional constraint[2] which serves as a penalty term for the transformation methods.

In Ref. 1, it was shown that calculating the gradient of the transformation function does not require calculating the gradients of the individual functions from which the transformation function is constructed. As a result, the gradient of the transformation function requires very little computation and, hence, the minimization of this function can be performed efficiently. Therefore, transformation methods are able to exploit the implicit nature of an optimal design problem—unlike other (primal) methods—and are especially attractive for optimizing large-scale structural and mechanical systems. In addition, transformation methods are attractive because they are globally convergent; i.e., they have been proven (in exact arithmetic) to converge to a local minimum from *any* starting design. This means that transformation methods are robust and reliable. However, global convergence should not be confused with the problem of finding a global minimum.

There are three classes of transformation methods that have been most commonly discussed in the literature: the penalty function, the barrier function, and the multiplier (or augmented Lagrangian) methods. The penalty and barrier methods have been referred to as sequential unconstrained minimization techniques (SUMTs) by Fiacco and McCormick.[3] The basic ideas behind SUMTs and multiplier methods have been presented in Ref. 1. Here, two specific multiplier algorithms, one developed by Powell[4] and the other by Fletcher,[5] are described. It is shown that the gradient of the transformation function can be calculated without calculating gradients of individual constraint functions. An operations count is presented demonstrating the savings in this approach over two other approachs used in the literature. Computer programs implementing the use of this technique in SUMTs and multiplier algorithms are developed. Applications of these on a set of structural design problems are presented and discussed.

II. Transformation Methods

To present ideas of transformation methods as they apply to the design of structural and mechanical systems, a simplified model of the design problem is considered. It is understood that the methods presented here are applicable to more complex models for the design problem.[6]

The optimal design problem is defined as follows:

$$\min f(b,z) \quad \text{subject to} \quad g_i(b,z) \le 0, \qquad i=1,...,m \tag{1}$$

where z is determined from the state equation

$$K(b)z=F(b) \tag{2}$$

Here b is a k vector of design variables, z an n vector of nodal displacements, n the number of degrees of freedom of the finite element model for the structure, $K(b)$ an $n\times n$ structural stiffness matrix, $F(b)$ an n vector of applied loads, f a cost function to be minimized, and g_i represents constraints such as on stress and displacement. Equality constraints have been omitted in Eqs. (1) and (2) to keep notation simple. However, equality constraints are treated routinely as explained later.

Equations (1) and (2) are considered explicitly in Sec. III where the method of computing the gradient of the trans-

Received Oct. 4, 1982; revision received May 2, 1983. Copyright © American Institute of Aeronautics and Astronautics, Inc., 1983. All rights reserved.

*Assistant Professor, Division of Materials Engineering, College of Engineering; currently with GMI Engineering and Management Institute, Flint, Mich. Member AIAA.

†Professor, Division of Materials Engineering, College of Engineering. Member AIAA.

formation function is described. To keep presentation of transformation methods simple, however, Eqs. (1) and (2) are restated in the standard nonlinear programming form:

$$\min f(b) \quad \text{subject to } g_i(b) \leq 0, \qquad i=1,\ldots,m \tag{3}$$

It is understood that f and g_i are implicit functions of design variables.

All transformation methods convert the constrained problem (3) into an unconstrained problem for the transformation function

$$\phi(b,r) = f(b) + P(g(b),r) \tag{4}$$

where r, in general, is a vector of controlling parameters and P is a real-valued function whose action of imposing the penalty is controlled by r. The local minimum of $\phi(b,r)$ at the νth iteration is denoted by $b^{(\nu)}$. The idea of transformation methods is very attractive because well-developed unconstrained algorithms can be used to completely solve the constrained problem. It is shown in Sec. III that the idea of transformation methods is also very attractive for design problems where implicit constraints must be treated. For a detailed theoretical basis of transformation methods Refs. 3, 5, and 7-9 should be consulted. SUMTs are discussed next, followed by multiplier methods.

SUMTs

In the penalty function methods, the most popular penalty function $P(g(b),r)$ is the quadratic loss function given by

$$P(g(b),r) = r\sum_{i=1}^{m} [g_i^+(b)]^2 \tag{5}$$

where r is a scalar parameter and $g_i^+(b)$ is equal to $\max(0,g_i(b))$. Other penalty and barrier functions have been used quite often.[3,7,9,10] It can be shown that as $r \to \infty$, $b \to b^*$, where b^* is a solution of the constrained problem.

SUMTs possess a number of undesirable properties (see, e.g., Refs. 1 and 9). In particular, the Hessian matrix of the unconstrained function becomes ill-conditioned as $r \to \infty$. Despite these difficulties, SUMTs were the first methods to be used for design and other general optimization problems. In this regard, the works of Cassis and Schmit[10] and Prasad and Haftka[11] are particularly noteworthy. These researchers have used certain "extended-barrier functions" that are free from difficulties of conventional SUMTs to solve a wide variety of structural design problems.

Multiplier Methods

The multiplier methods have been developed in the recent literature to alleviate some of the difficulties with SUMTs. The basic derivations and ideas behind the methods may be found in Refs. 5, 8, and 12-15. In these methods there is no need for the controlling parameters r to go to infinity. As a result the transformation function ϕ has good conditioning with no singularities. Furthermore, multiplier methods, like SUMTs, are globally convergent and have been proven to possess faster rates of convergence than SUMTs.[12]

In multiplier methods, the penalty function is given as[5]

$$P(g(b),r,\theta) = \frac{1}{2}\sum_{i=1}^{m} r_i[(g_i(b)+\theta_i)^+]^2 \tag{6}$$

where θ_i and $r>0$ are parameters associated with the ith constraint. The product $r_i\theta_i \equiv \lambda_i$ is the Lagrange multiplier associated with the ith constraint.

Two multiplier algorithms are given. (The reason for presenting these algorithms is that references are made to them while presenting and discussing numerical results.) Algorithm 1 has been developed by Powell (see Ref. 5, Sec. 4.4) and algorithm 2 by Fletcher (see Ref. 5 Sec. 4.70). The reader may refer to Ref. 5 and references cited in that paper for convergence proofs of these algorithms.

Algorithm 1[5]

Step 1. Set $\nu=0$, $K=\infty$; estimate vectors $b^{(0)}$, $\theta^{(1)}$, r and scalars $\alpha>1$, $\beta>1$, $\epsilon>0$, where ϵ is the desired accuracy.

Step 2. Set $\nu=\nu+1$.

Step 3. Minimize $\phi(b,r,\theta^{(\nu)})$ of Eqs. (4) and (6) subject to $b^L \leq b \leq b^U$. This minimization is performed starting from $b^{(\nu-1)}$. Also, b^L and b^U are lower and upper bounds on design variables, respectively. These constraints are considered separately because in structural optimization variables b_i cannot become negative during the iterative process. The theory remains valid. Let the solution at this step be $b^{(\nu)}$.

Step 4. Evaluate g_i, $i=1,\ldots,m$ at $b^{(\nu)}$. Set

$$\bar{K} = \max_i \; |\max(g_i, -\theta_i^{(\nu)})|$$

and

$$I = \{ i : |\max(g_i, -\theta_i^{(\nu)})| > K/\alpha \}$$

If $\bar{K} \leq \epsilon$, then stop; $b^{(\nu)}$ is the solution, and $\lambda_i^{(\nu)} = r_i\theta_i^{(\nu)}$ for $i=1,\ldots,m$ are the Lagrange multipliers. If $\bar{K}>\epsilon$, then go to step 5.

Step 5. If $\bar{K} \geq K$, set $r_i = \beta r_i$ and $\theta_i^{(\nu)} = \theta_i^{(\nu)}/\beta$ for all $i \in I$, and go to step 3. If $\bar{K}<K$, go to step 6. Note that in this step only r is changed. The Lagrange multipliers, $\lambda_i = r_i\theta_i$, are kept constant.

Step 6. Set

$$\theta_i^{(\nu+1)} = \theta_i^{(\nu)} + \max(g_i, -\theta_i^{(\nu)}), \qquad i=1,\ldots,m \tag{7}$$

If $\bar{K} \leq K/\alpha$, set $K=\bar{K}$ and go to step 2.

Step 7. Set $r_i = \beta r_i$ for each $i \in I$, set $\theta_i^{(\nu+1)} = \theta_i^{(\nu+1)}/\beta$ for each $i \in I$, set $K=\bar{K}$ and go to step 2.

The algorithm presented above is based on the Powell-Hestenes formula[7] for changing θ_i. Powell[4] shows that the algorithm cannot cycle endlessly between steps 3 and 5. The increase in r, so as to force global convergence, turns out to be an outstanding feature of multiplier methods. Results on test problems presented in Sec. IV confirm this fact. It should be noted that the formula (7) to change θ_i uses only first-order information. Powell[4] shows that algorithm 1 possesses a rapid rate of linear convergence.

The algorithm above can be readily generalized to handle problems with mixed equality and inequality constraints. Consider: minimize $f(b)$, subject to $g_i(b) \leq 0$, $i=1,\ldots,m$, and $h_i(b)=0$, $i=m+1,\ldots,\ell$. Then, we have

$$\phi = f + \frac{1}{2}\sum_{i=1}^{m} r_i[(g_i+\theta_i)^+]^2 + \frac{1}{2}\sum_{i=m+1}^{\ell} r_i(h_i+\theta_i)^2$$

$$\bar{K} = \max\left\{ \max_{1\leq i\leq m} \left|\max(g_i, -\theta_i)\right|, \max_{m+1\leq i\leq \ell} \left|h_i\right| \right\}$$

and Eq. (7) includes $\theta_i^{(\nu+1)} = \theta_i^{(\nu)} + h_i$, $i=m+1,\ldots,\ell$. The set I in step 4 is expanded to include i for which $|h_i| > K/\alpha$.

A superlinearly convergent algorithm, based on second-order information has been developed by Fletcher (see Ref. 5, Sec. 4.7) and is given below.

Algorithm 2[5]

Steps 1-5 are identical to algorithm 1.

Step 6. Compute N (the matrix of gradients of constraint functions) and W at $b^{(\nu)}$. Set

$$J = \{ i \,|\, g_i(b^{(\nu)}) > -\theta_i^{(\nu)} \}$$

$$\theta_i^{(\nu+1)} = \theta_i^{(\nu)} - [(N^T W^{-1} N)^{-1} g]_i / r_i \quad \text{for} \quad i \in J$$
$$= 0 \quad \text{for} \quad i \notin J \qquad (8)$$

NOTE: W is a positive definite approximation to $\nabla^2 \phi$. Here W is generated using a quasi-Newton update.

Step 7. Let $\Delta\theta \equiv \theta^{(\nu+1)} - \theta^{(\nu)}$. Compute $\Delta^{PH}\theta$ from Eq. (7) and $\Delta^N\theta$ from Eq. (8), where $\Delta^{PH}\theta$ and $\Delta^N\theta$ are the corrections in θ_i predicted by the Powell-Hestenes and Newton formulas, respectively. If necessary, increase r_i to satisfy:

$$r_i \geq \alpha r_i \left| \frac{\Delta^{PH}\theta_i - \Delta^N\theta_i}{\Delta^{PH}\theta_i} \right| \qquad (9)$$

Modify θ_i such that the product $r_i\theta_i$ remains the same before and after Eq. (9). Set $K = \bar{K}$ and go to step 2.

The main differences between algorithms 1 and 2 are the way in which θ_i is changed and the way in which r_i is increased. In algorithm 2, r_i is increased based on Eq. (9) as against the strategy used in step 7 of algorithm 1. Fletcher[5] developed the formula in Eq. (9) to keep r_i well scaled among themselves.

III. Computing Gradient of a Transformation Functional

Transformation methods minimize the transformation functional $\phi(f(b),g(b),r)$. Most commonly used methods require the gradients of ϕ. The major point of departure of the proposed methods is the way in which the gradient of ϕ is calculated for the structural design problem in Eqs. (1) and (2). The approach proposed here is compared with two other techniques used in the literature. It is important to keep in mind that what follows is also applicable to other engineering design problems that involve implicit functions of design.

For the problem in Eqs. (1) and (2), the transformation functional ϕ is an implicit function of design due to the implicit nature of the cost and constraint functions. In other words, we have

$$\phi \equiv \phi(f(b,z(b)), g(b,z(b),r))$$

where $z(b)$ is an n vector determined from Eq. (2), and $g \equiv (g_1,\ldots,g_m)$. To keep the discussion simple, consider the problem of determining derivatives of the exterior penalty function

$$\phi(f(b,z(b)), g(b,z(b))) = f(b,z(b)) + r \sum_{i=1}^{m} [g_i^+(b,z(b))]^2$$

Three methods of computing the derivative of ϕ are given.

Method I

The key idea here is to look upon the functional $\phi(f(b,z(b)), g(b,z(b))$ as simply the function $\phi(b,z(b))$, and then apply the standard adjoint variable technique[6] to differentiate ϕ. This is derived in Ref. 1 as

$$\frac{d\phi}{db} = \frac{\partial\phi}{\partial b} - \lambda^{\phi T} \frac{\partial}{\partial b}(K(b)z - F(b)) \qquad (10)$$

where λ^ϕ is determined from

$$K\lambda^\phi = \frac{\partial\phi^T}{\partial z} \qquad (11)$$

In the above equation, arguments of ϕ are suppressed for convenience. Also, the notation $d\phi/db = (d\phi/db_1,\ldots,d\phi/db_k)$ is used. Calculations for $\partial\phi/\partial b$ and $\partial\phi/\partial z$ are explicit and quite straightforward.

Method II

This method (the "direct method") has been used in Refs. 16, 17, and others. Here,

$$\frac{d\phi}{db} = \frac{\partial\phi}{\partial b} + \frac{\partial\phi}{\partial z}\frac{dz}{db} \qquad (12)$$

where dz/db is determined from

$$K\frac{dz}{db} = -\left[\frac{\partial}{\partial b}(Kz) - \frac{dF}{db}\right] \qquad (13)$$

Method III

This method is used widely in the literature and the reason for this will become clear subsequently. In the preceding two approaches, the function $\phi(f(b,z(b)))$, $g(b,z(b))$ was treated simply in the form $\phi(b,z(b))$. The following method, however, explicitly considers the presence of $f(b)$ and $g_i(b)$, $i=1,\ldots,m$. Thus, the derivative of ϕ is written as

$$\frac{d\phi}{db} = \frac{df}{db} + 2r\sum_{i=1}^{m} g_i^+ \frac{dg_i^+}{db} \qquad (14)$$

where dg_i^+/db is determined using the adjoint variable technique[6]:

$$\frac{dg_i^+}{db} = \frac{\partial g_i^+}{\partial b} - \lambda^{iT}\frac{\partial}{\partial b}(K(b)z - F(b)), \qquad i=1,\ldots,m \quad (15)$$

$$K\lambda^i = \frac{\partial g_i^{+T}}{\partial z}, \qquad i=1,\ldots,m \qquad (16)$$

This approach to compute $d\phi/db$ requires the gradients of individual constraint functions; that is, in Eq. (14) it is necessary to compute dg_i^+/db for each active constraint g_i. Most packaged programs for optimization require the user to supply the gradients of each individual constraint. This is partly because the codes address the nonlinear programming problems (which have functions depending explicitly on design, unlike in structural problems) and also because most optimization methods (such as gradient projection, feasible directions, etc.) do indeed require the gradients of individual constraints. As an illustration, the subroutine VF01AD,[18] which is based on a multiplier method similar to the one given in algorithm 2, requires the user to supply a subroutine in which gradients of each active constraint function are to be computed. In this case, a summation operation as in Eq. (14) in done inside VF01AD to compute the gradient of the transformation function.

Operation Count for Each Method

From the expressions given above for computing $d\phi/db$ by the three methods, a ready comparison is available. In what follows, each column of unknowns obtained by solving a linear system will be referred to as "col."

Method I: To solve for ϕ from Eq. (11): 1 col

Method II: To solve for dz/db from Eq. (13): k col

Method III: To solve for λ^i, $i=1,\ldots,m$ from Eq. (16): m col

It is evident that method I proposed here requires the least amount of computation. The savings of method I over methods II and III are $(k-1)$col and $(m-1)$col, respectively. Now, we can assume that the $n \times n$ stiffness matrix K has already been decomposed at the time of evaluating the functions. In this case, a Gaussian elimination scheme can solve each col of unknowns (with K as the coefficient matrix) in n^2 operations.[19] Therefore, the savings in method I over methods II and III are $(k-1)n^2$ operations and $(m-1)n^2$ operations, respectively, for each gradient evaluation of the functional. The key idea to be noted is that method I does not require computing the gradients of the individual functions from which the functional ϕ is constructed. Finally, it may be noted that method III is better than method II since it is usual to have $m<k$, where m is the number of active constraints.

IV. Example Problems

This section contains a discussion of results obtained for four structural design problems. The results and discussion presented below are typical of a wider set contained in Ref. 20. Each problem is solved by the exterior penalty function of Eq. (7) and by algorithms 1 and 2. These algorithms have been coded in FORTRAN programs and are referred to as SUMT, M-3, and M-5, respectively. Double-precision computation on an IBM 370/168 computer is used.

The four example problems discussed below have been solved by combining these programs with a program FROPT,[21] which analyzes the structure and generates gradients of the transformation function. More significant is the fact that the gradient of the transformation function is calculated using method I of Sec. III. A quasi-Newton method (subroutine VE05AD from the Harwell subroutine library[18]) is used to minimize $\phi(b,r)$ with respect to b, subject only to lower and upper bounds on b. In VE05AD, the initial estimate of $\nabla^2\phi$ is set automatically to I. However, VE05AD has been modified so that the initial estimate of $\nabla^2\phi$ can be set by the user. This has been done so that the approximation to $\nabla^2\phi$ can be carried forward from one minimization to the next. This idea has been suggested by Fletcher[5] and some numerical experimentation has confirmed that this is more efficient than the idea of resetting $\nabla^2\phi$ to I at the start of each minimization,[22] even when the active set has changed. The theory behind transformation methods requires that each minimization of ϕ be carried out exactly. However, this is usually not possible for large-scale problems. Therefore, a limit on the maximum number of function and gradient evaluations has been imposed during the minimization step. The idea is to carry out an approximate minimization and allow the λ parameters to change so that the right set of active constraints is determined. This idea has proved to be reasonably successful although much work (as on the lines in Ref. 23) still needs to be done to determine the level of approximation in each minimization. In the following problems the convergence criterion is that $K\leq\epsilon$, where K is defined in algorithm 1 and ϵ is a suitably small number which is selected based on the size of the problem. The scalars $\theta_i^{(l)}$, $i=1,\ldots,m$ are initially set to zero. Also, $\alpha=1.5$ and $\beta=10$ is used. Powell,[5] on the other hand, chooses $\alpha=4$. However, for structural problems, $\alpha=4$ causes parameter r to become prohibitively large. The initial value of r_i in both SUMTs and multiplier methods is based on some sort of scaling of ϕ.[5] But some trial and error is sometimes necessary. Finally, constraints are expressed in normalized form. For example, a stress constraint $\sigma(b)\leq\sigma^a$ is expressed as $\sigma(b)/\sigma^a\leq 1$. This scaling of the constraint functions is found to be effective. The reader may refer to an alternate scaling technique in Ref. 24.

Rosen-Suzuki Problem

This problem is obtained from Ref. 25 and is a standard problem for testing algorithms. The problem has four variables and three constraints (i.e., $k=4$ and $m=3$). The optimal solution is $(0,1,2,-1)$. The starting design is taken as $(0,0,0,0)$. The numbers of function and gradient evaluations taken by each algorithm are given in Table 1. All of the algorithms converged to the optimum solution. From Table 1, it is clearly evident that SUMT is quite inefficient compared with the multiplier algorithms. This substantiates the fact that SUMTs have slower rates of convergence as compared to multiplier methods.[12]

10-Bar Truss (Fig. 1)

The problem is to minimize mass of the truss subject to stress, displacement, natural frequency, and member size constraints. The cross section of each member is an I section with the depth and width as design variables. The web and flange thicknesses are each kept fixed at 0.254 cm. There are thus 20 design variables for the problem. The loading consists of 445 kN applied in the negative y direction at nodes 2 and 4. There are 10 stress constraints, 8 displacement constraints, 1 constraint on natural frequency, and lower bound constraints

Table 1 Rosen-Suzuki problem

Method	SUMT	M-3	M-5
Functions evaluations	191	67	64
Gradient evaluations	130	51	52

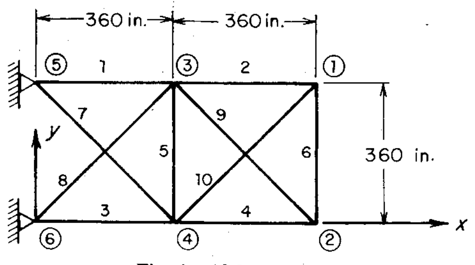

Fig. 1 10-bar truss.

Table 2 Results for 10-member truss

	M-3						M-5			SUMT	
ν	1	2	3	4	5	6	1	2	3(2)[a]	1	2
f, kg	2283.4	2307.4	2298.3	2273.8	2258.0	2258.0	2283.4	2333.7	2211.7	2242.6	2165.4
$\bar{K}$	0.025	0.002	0.0003	0.002	0.002	0	0.025	0.017	0.0006	0.048	0.0006
$F+G$	$400F + 391G$	$400F + 400G$	$400F + 400G$	$400F + 393G$	$40F + 16G$	$5F + 1G$	$400F + 391G$	$97F + 53G$	$345F + 169G$	$400F + 396G$	$151F + 97G$
I	Frequency y displacement at nodes 1 and 2, and b_5^L						Same as M-3 and stress in member 10 and b_6^L			Displacement as in M-3, stress in member 6, b_5^L, and b_6^L	
CPU, s	289						146			101	

[a] Number in parenthesis refers to the number of times $\bar{K}\geq K$ in step 5 of algorithms 1 and 2.

on each design variable, making a total of 39 constraints. The data for this problem are obtained from Ref. 6 as follows: Young's modulus = 75.2 GPa, material density = 2768 kg/m^3, displacement limit = 5.08 cm, stress limit = 172.4 MPa, lower bound on natural frequency = 22 Hz, starting design is (76.2, 76.2) cm, and b_i^L is (1.02, 1.02) cm for all members. The initial mass of the structure is 1709.2 kg and maximum constraint violation is 101%. Results for the problem are given in Table 2. In the table, ν is the iteration number, f the mass, $\bar{K}$ the maximum constraint violation, F the number of calls to the function evaluation subroutine, G the number of calls to the gradient evaluation routine, and I the active set. The final designs are given as

M-3: {(187.2, 257.3), (39.6, 1.02), (111.5, 262.9), (78.8, 128.8), (1.02, 1.02), (11.2, 1.02), (68.6, 61.0), (51.3, 206.3), (132.1, 187.7), (11.9, 44.5)} cm

M-5: {(141.2, 244.1), (23.1, 1.02), (87.4, 245.1), (94.2, 136.4), (1.02, 1.02), (1.02, 1.02), (79.5, 83.1), (51.1, 207.3), (140.7, 205.2), (11.2, 14.5)} cm

SUMT: {(163.6, 247.9), (31.0, 1.02), (161.8, 221.5), (95.8, 127.0), (1.02, 1.02), (1.02, 1.02), (69.3, 62.7), (110.5, 184.2), (132.6, 189.0), (7.9, 12.2)} cm

The optimum mass of the truss from SUMT, M-3, and M-5 is 2165.5, 2258.0, and 2212 kg, respectively. This problem has been solved extensively in the literature[6] with the cross-sectional area of each element being the design variable. The optimal solution obtained is 2173.6 kg. The different solutions obtained by each algorithm are due to the presence of multiple local minima, which can be verified by examining the active constraints given in Table 2.

Consider the results of M-5 as given in Table 2. The solution is obtained in three minimizations of the transformation function. For each minimization, a limit of 400 is imposed on the number of function evaluations. We see that the first minimization is approximate, while the next two are exact. The value of the convergence parameter K is not considered for approximate minimizations. The first minimization always takes the most amount of time because the active set changes considerably and also because subsequent minimizations make use of the approximation to $\nabla^2\phi$ which has been formed. In the third minimization, the value 2 in parentheses in the iteration row signifies that two additional minimizations are performed in which Lagrange multipliers $\lambda_i = r_i\theta_i$ are kept constant and only parameters r_i are increased. This corresponds to step 5 in algorithm 1 or 2. The increase in r then causes the parameter K to reduce from 0.017 to a value of 0.0006, which satisfies the convergence criterion of 0.001. The monotonic reduction in K reflects the global convergence property of transformation methods. This stable behavior is also seen in medium- to large-scale problems that are discussed later. It is worth noting that the CPU time of 146 s would have been considerably more had either method II or III in Sec. III been used to compute the gradient of ϕ. It is interesting to note that SUMT performed more efficiently (less computing time) than M-3 and M-5.

2-Bay, 6-Story Frame (Fig. 2)

This frame has 21 joints, 30 members, and 54 degrees of freedom. The elements of the structure are combined into 18 groups so that the optimum structure is symmetric. Each element is a hollow tubular cross section with radius and thickness as design variables. There are thus a total of 36 design variables for this problem. The starting design variables for each group are 12.7 and 2.54 cm and their lower limits are 1.27 and 0.127 cm. The objective is to minimize mass of the structure subject to constraints on stresses, displacements, and member sizes. The frame is designed for two loading conditions which include uniformly applied loads on the elements and concentrated loads on the nodes. The load data are obtained from Ref. 6, p. 263. There are 72 stress constraints, 108 displacement constraints, 36 local-buckling constraints, and a lower bound constraint on each design variable, thus giving a total of 252 constraints. For the problem, Young's modulus = 206.9 GPa, weight density = 7850 kg/m^3, and yield stress = 248.2 MPa. The displacement limits are 5.08 cm at all nodes in both the x and y

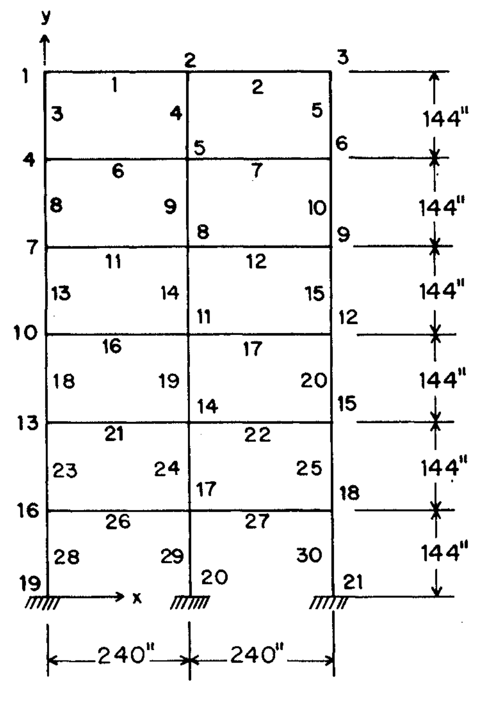

Fig. 2 2-bay, 6-story frame.

Table 3 Iteration histories for 2-bay 6-story frame

	M-3					M-5				SUMT			
ν	1	2(3)	3(6)	4	5(1)	1	2(3)	3	4(1)	1	2	3	4
f, kg	14,851	12,292	12,066	12,066	12,073	14,851	11,058	11,095	11,088	15,926	15,848	15,538	15,455
$\bar{K}$	0.021	0.006	0.004	0.003	0.0009	0.021	0.018	0.014	0.01	0.01	0.003	0.001	0.0001
$F+G$	150F + 85G	146F + 73G	92G + 40G	4F + 1G	180G + 17G	150F + 85G	323F + 214G	14F + 6G	8F + 2G	150F + 86G	150F + 12G	150F + 12G	150F + 10G
I	25 constraints					28 constraints				11 constraints			
CPU, s	230					301				248			

directions. The stress constraints are obtained from the AISC specifications[26] as

$$\frac{f_a}{F_a} + c_l \frac{f_b}{F_b} \le 1 \quad \text{and} \quad \frac{f_s}{F_s} \le 1$$

where f_a, f_b, and f_s are the actual stresses in the element in axial, bending, and shear, respectively, and F_a, F_b, and F_s are the corresponding allowable stresses. c_l is a magnification factor.[26] Further, lower and upper bounds are imposed also on the ratio of radius to thickness of the hollow tubes to safeguard against local buckling.[21,26] For more details on the stress and buckling constraints imposed the reader is referred to Ref. 27. The final results for the problem are given in Table 3. The maximum violation at the starting design is 107%. The initial mass of 22,113 kg is reduced to final values of 15,455, 12,073, and 11,088 kg, respectively, by SUMT, M-3, and M-5. The solutions may be compared with the optimum value of 11,018 kg obtained in Ref. 6 for the same frame designed under a somewhat different set of constraints. The final designs are given as

M-3: {(23.7, 0.533), (28.2, 0.64), (29.3, 0.64), (23.4, 0.74), (29.1, 0.64), (19.34, 1.17), (22.6, 0.61), (8.5, 0.51), (27.4, 0.58), (30.5, 0.66), (10.8, 0.89), (20.5, 1.37), (10.3, 0.58), (36.3, 0.79), (26.1, 0.89), (15.2, 1.17), (37.8, 0.87), (18.1, 1.22)} cm

M-5: {(23.8, 0.51), (25.7, 0.56), (26.4, 0.56), (27.9, 0.61), (32.2, 0.71), (21.5, 0.89), (24.6, 0.53), (10.4, 0.43), (23.7, 0.51), (28.2, 0.61), (15.9, 0.58), (21.7, 1.22), (11.6, 0.46), (31.6, 0.69), (29.2, 0.64), (17.3, 0.74), (40.7, 0.89), (19.4, 1.00)} cm

SUMT: {(17.0, 1.09), (22.2, 0.87), (22.3, 1.12), (19.9, 1.42), (24.5, 1.63), (18.3, 1.45), (19.1, 1.17), (8.54, 0.51), (24.8, 0.58), (29.3, 0.79), (11.4, 1.14), (25.9, 1.10), (9.91, 0.84), (25.3, 1.27), (21.7, 0.76), (13.0, 1.19), (31.5, 0.91), (16.2, 1.3)} cm

The final mass of 15,455 kg obtained by SUMT is highly nonoptimal as compared with the multiplier algorithms. A maximum limit of 150 function evaluations was imposed during each minimization.

Consider the results from M-3 as given in Table 3. In the second minimization, the scalars r_i were increased three times (see step 5 in algorithm 1) and K was forced to reduce from 0.006 to 0.004. Furthermore, in the third minimization r_i were increased six times to reduce K further to a value of 0.003. K is ultimately reduced to a value of 0.0009. The stable behavior of the algorithm on this relatively complex problem and its global convergence property is again evident. Finally, it may be observed that the final value of K from M-5 is 0.01 which, although acceptable, is not as small as from M-3. This can be remedied by increasing slightly the initial value of scalars r_i. However, for comparison purposes, the initial values of the parameters are kept the same for M-3 and M-5. Also, since the active constraints at the final design in M-3 and M-5 are different, we may conclude that the solutions correspond to different local minima.

200-Bar Truss (Fig. 3)

The problem is to find cross-sectional areas for the truss elements to minimize mass of the structure, subject to constraints on stress and on member sizes. The 200 elements of the structure are linked to 29 groups. There are a total of three loading conditions. There are 29 design variables and 87 stress constraints since there are 29 groups. Moment of inertia of each group of members is treated as a design variable and the cross-sectional area is calculated as $A = I^{0.5}$. This problem is obtained from Ref. 6. The stress in each element is limited to a value of 68.95 MPa for both tension and compression. A lower bound of 0.645 cm^2 is imposed on the cross-sectional area of each element. The loading and other data are given in detail in Ref. 6. The iteration histories for the problem are given in Table 4. The final designs with the three methods are as follows:

M-3: {0.42, 47.0, 0.42, 0.42, 155.3, 2.5, 0.42, 517.4, 8.7, 656.8, 6.7, 0.42, 1114.3, 0.42, 1602.1, 9.6, 10.4, 2386.3, 0.42, 2969.4, 22.1, 16.2, 4121.9, 9.6, 5004.4, 48.7, 3429.3, 6976.5, 7030.2} cm^4

M-5: {0.42, 40.0, 0.42, 0.42, 165.3, 375.9, 1.67, 665.6, 5.83, 0.42, 1108.0, 0.83, 1594.3, 9.2, 15.0, 2258.9, 2.50, 2967.3, 26.2, 5.0, 4249.7, 0.42, 5144.6, 27.5, 4863.7, 6238.9, 6783.7} cm^4

SUMT: {0.42, 37.9, 0.42, 0.42, 246.8, 2.5, 0.42, 366.7, 1.25, 672.2, 5.41, 1.66, 1387.7, 0.83, 1580.4, 8.3, 0.83, 2225.2, 0.83, 2886.2, 12.1, 72.0, 4424.5, 2.91, 4639.7, 40.4, 7464.3, 7350.2, 6569.8} cm^4

The starting design for the problem is as follows: 208.1 cm^4 for groups 1-12, 14, 16, 17, 19, 21, 22, 24, 26; 2081.2 cm^4 for groups 13, 15, 18, 20, 23, 25; and 8325 cm^4 for groups 27, 28, 29.

A lower limit of 0.416 cm^4 is imposed on each design variable. The maximum constraint violation at the starting design is 38.2%. The initial mass of the structure is 16,186 kg

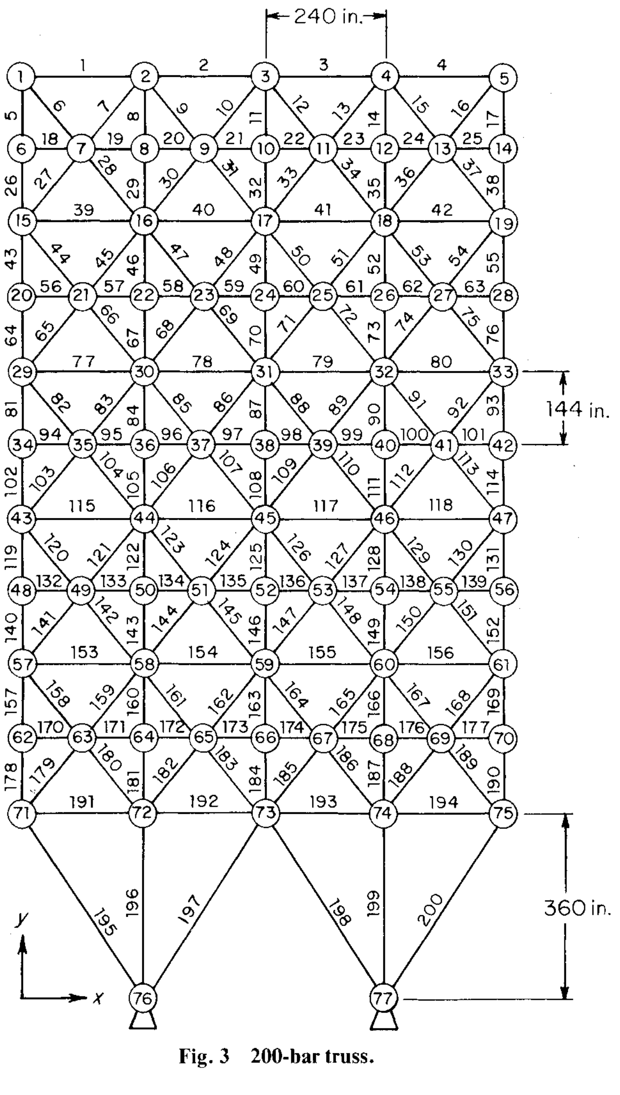

Fig. 3 200-bar truss.

Table 4 Iteration histories for 200-member truss

	M-3					M-5					SUMT			
ν	1	2	3	4	5	1	2	3	4	5	1	2	3	4
f, kg	14,817	12,705	12,263	12,228	12,066	14,817	12,589	12,130	12,043	11,912	14,555	13,322	12,885	12,503
$\bar{K}$	0.016	0.015	0.016	0.004	0.002	0.016	0.004	0.017	0.015	0.008	0.025	0.009	0.003	0.003
$F+G$	$150F + 84G$	$150F + 86G$	$150F + 87G$	$150F + 61G$	$150F + 123G$	$150F + 84G$	$150F + 100G$	$150F + 97G$	$150F + 60G$	$150F + 94G$	$150F + 93G$	$150F + 76G$	$150F + 66G$	$150F + 66G$
I	23 constraints					21 constraints					18 constraints			
CPU, s	1,524					1,556					1,400			

Table 5 Summary of results

Problem / Method	Rosen-Suzuki	10-bar truss, kg (s)	2-bay 6-story frame, kg (s)	200-bar truss, kg (s)
M-3	$67F+51G$	2,258 (289)	12,073 (230)	12,066 (1,524)
M-5	$64F+51G$	2,212 (146)	11,088 (301)	11,912 (1,556)
SUMT	$191F+130G$	2,166 (101)	15,455 (248)	12,503 (1,400)

and the final masses are 12,503, 12,066, and 11,912 kg, respectively, by SUMT, M-3, and M-5. The optimum mass of 11,912 kg obtained by M-5 is indeed the same as obtained in Ref. 6. However, the result in Ref. 6 has involved changing certain parameters during the iterative process and then restarting the program. In view of this, and also the fact that the 200-bar truss problem has caused great difficulty to many researchers, the results obtained by M-3 and M-5 are very encouraging. In fact, M-3 and M-5 have given the best results (in terms of minimum cost) as compared to many other algorithms.[20]

Results from SUMT are not as good as those obtained from the multiplier methods. A limit of 150 is imposed on the number of function evaluations during each minimization. However, none of the minimizations in the three algorithms were carried out exactly within the limit. The computer runs were terminated after five approximate minimizations. A question that arises in solving large problems such as this is: "How much accuracy is necessary for each minimization to be performed?" This is a matter which requires more study.

A summary of the optimum solutions and computing times is given in Table 5. This table and results in Ref. 20 indicate that M-3 and M-5 perform roughly the same. SUMT performs worse than M-3 and M-5, but this inferiority is most noticeable only for small problems (such as mathematical programming problems). Finally, although the 200-bar truss is a fairly large problem, computing times of 1556 s are considerable and need to be reduced. Some ways of improving efficiency of transformation methods are discussed in the next section.

V. Summary and Conclusions

Computational aspects of the transformation methods are presented and discussed. An efficient method is given to compute the gradient of the transformation function. The method requires only one forward and backward substitution regardless of the number of design variables or the number of constraints in the problem. Furthermore, the method is compared to two other approaches used in the literature. There is a savings of $[\min(k,m)-1]n^2$ operations, where n is the number of degrees of freedom for the structure. The SUMT and multiplier methods are applied to four problems which include a relatively difficult 2-bay, 6-story framed structure and also a 200-member truss.

Multiplier methods have performed better than the exterior penalty function (SUMT). A main feature of multiplier methods is their stable behavior. Specifically, the convergence parameter K is forced to decrease monotonically by increments in the penalty parameters r_i (which is in conformity with the global convergence results of Ref. 4). Among the multiplier methods, algorithms 1 and 2 both perform equally well. While the presence of multiple local minima makes it difficult to make any exact comparisons, the claim that algorithm 2 is superior to algorithm 1 as made by Fletcher[5] is not vindicated. One possible reason for this is the difference between the structural design and mathematical programming problems.

An important limitation exists in applying method I for computing gradients of ϕ. In recent years, gradients of individual constraints have been used to construct constraint approximations.[10,11] Also, individual constraint gradients are used to generate approximate Hessian matrices.[28] Individual constraint gradients are not computed in method I; however, the Hessian of ϕ is generated using a quasi-Newton approach. Method I can be used only when exact analyses are performed. While use of approximate analysis generally reduces computing time, proofs of global convergence no longer hold. There is usually a tradeoff between efficiency and reliability, and the choice of algorithm depends on the user's requirements. The topic of exact vs approximate analyses is important, however, and is an area for further research.

One difficulty in transformation methods is to decide on the accuracy with which each minimization should be performed. This is especially critical for large-scale structural problems where an exact minimization is inefficient. A second difficulty is the choice of subroutine to perform minimization of the transformation function. The program VE05AD from the Harwell Library is used in the present work. However, this subroutine requires a large number of function evaluations during line search (which involves re-analysis of the structure). For example, for the 10-bar truss solved by M-5 in Table 2 a total of 842 function evaluations are required.

Hence, it is worthwhile investigating some other minimization subroutines. Also, an approximate function evaluation scheme should be considered during line search. In view of the above remarks, we may conclude that multiplier methods are very reliable, but not efficient. Much work (as discussed above) needs to be done to improve their efficiency.

As already noted, an efficient technique has been given to compute the derivative of an implicit functional of the form $q(\alpha(b,z(b)))$, where $z(b)$ is obtained from the equation $K(b)z=F$ and α represents a vector of functions. Thus far, q has been identified only with the transformation function. This has led to payoffs in minimizing the transformation function. However, it is evident that we can take advantage of the situation whenever there occur such functionals and whose gradients are required. One application is the computation of exact second-order derivatives. Thus, we can compute $\nabla_b^2 q(\alpha(b,z(b)))$—the Hessian matrix of q—without computing $\nabla_b^2 \alpha_i(b,z(b))$ for each i. In the above remark, q can either be a transformation function or the Lagrangian function for the problem in Eqs. (1) and (2). This idea, coupled with the technique in Ref. 29, then raises the possibility of computing exact Hessians in structural optimization as opposed to using quasi-Newton updates which may offset the increased computation with faster convergence. This, at any rate, needs further investigation. Moreover, use of exact second-order information may prove effective in supporting post-optimality analysis of design sensitivity. The efficient technique which has been suggested to compute the derivative of a functional also motivates the idea of reformulation of the problem in Eqs. (1) and (2), as along the lines suggested in Ref. 2. Thus, we may formulate Eqs. (1) and (2) as:

$$\min f(b,z) \quad \text{subject to} \quad \sum_{i=1}^{m} \mu_i [\max(0,g_i(b,z))]^2 = 0 \qquad (17)$$

In Eq. (17) there is only one constraint of the form $q(\alpha(b,z(b)))=0$. The advantage of this "constraint addition" formulation is that the gradient of q can be calculated inexpensively without calculating gradients of each (active) constraint g_i. Unfortunately, the problem in Eq. (17) is singular because the gradient of q vanishes at any feasible design. Formulations such as in Eq. (17) might still prove attractive if near-optimal solutions can be obtained at an inexpensive cost, whereupon one may switch the problem in Eqs. (1) and (2) to zero on the solution.

Acknowledgment

This research was supported by NSF Grant CEE 80-08102.

References

[1]Belegundu, A. D. and Arora, J. S., "Potential of Transformation Methods in Optimal Design," *AIAA Journal*, Vol. 19, Oct. 1981, pp. 1372-1374.

[2]Arora, J. S., Haug, E. J., and Rajan, S. D., "Efficient Constraint Treatment in Structural Optimization," ASCE Paper 80-501, ASCE National Convention, Hollywood, Fla., Oct. 1980.

[3]Fiacco, A. V. and McCormick, G. P., *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley and Sons, Inc., New York, 1968.

[4]Powell, M. J. D., *Optimization*, edited by R. Fletcher, Academic Press, London, 1969, Chap. 19.

[5]Fletcher, R., "An Ideal Penalty Function for Constrained Optimization," *Journal of the Institute of Mathematics and Applications*, Vol. 15, 1975, pp. 319-342.

[6]Haug, E. J. and Arora, J. S., *Applied Optimal Design: Mechanical and Structural Systems,* Wiley-Interscience, New York, 1979.

[7]Gill, P. E. and Murray, W., eds., *Numerical Methods for Constrained Optimizations*, Academic Press, New York, 1974.

[8]Rockafeller, R. T., "Augmented Lagrange Multiplier Functions and Duality in Non-Convex Programming," *SIAM Journal of Control*, Vol. 12, 1974, pp. 268-285.

[9]Luenberger, D. G., *Introduction to Linear and Nonlinear Programming*, Addison-Wesley, Reading, Mass., 1973.

[10]Cassis, J. H. and Schmit, L. A., "On Implementation of the Extended Interior Penalty Function," *International Journal of Numerical Methods in Engineering*, Vol. 10, 1976, pp. 3-23.

[11]Prasad, B. and Haftka, R. T., "A Cubic Extended Interior Penalty Function for Structural Optimization," *International Journal for Numerical Methods in Engineering*, Vol. 14, No. 9, 1979, pp. 1107-1126.

[12]Bertesekas, D. P., "Multiplier Methods: A Survey," *Automatica*, Vol. 12, 1976, pp. 133-145.

[13]Hestenes, M. R., "Multiplier and Gradient Methods," *Journal of Optimization Theory and Applications*, Vol. 4, 1969, pp. 303-320.

[14]Schuldt, S. B., "A Method of Multipliers for Mathematical Programming Problems with Equality and Inequality Constraints," *Journal of Optimization Theory and Applications*, Vol. 17, No. 1/2, 1975, pp. 155-162.

[15]Schuldt, S. B., Gabriele, G. A., Root, R. R., Sandgren, E., and Ragsdell, K. M., "Application of a New Penalty Function Method to Design Optimization," *ASME Journal of Engineering for Industry*, Vol. 99, No. 1, Feb. 1977, pp. 31-36.

[16]Imai, K., "Configuration Optimization of Trusses by the Multiplier Method," Ph.D. Dissertation, Rept. UCLA-ENG-7842, 1978.

[17]Carpenter, W. C. and Smith, E. A., "Computational Efficiency in Structural Optimization," *Engineering Optimization*, Vol. 1, 1975, pp. 169-188.

[18]Harwell Subroutine Library, A.E.R.E., Building 8.9, Harwell, Didcot, Oxon, OX11ORA, England.

[19]Atkinson, K. E., *An Introduction to Numerical Analysis*, John Wiley and Sons, New York, 1978.

[20]Belegundu, A. D. and Arora, J. S., "A Study of Mathematical Programming Methods for Structural Optimization," Division of Materials Engineering, The University of Iowa, Iowa City, Tech. Rept. CAD-SS.82.5, Aug. 1982 (also available at NTIS).

[21]Belegundu, A. D. and Arora, J. S., "A User's Manual for Frame Optimization Package: FROPT (Level 1.0)," Division of Materials Engineering, The University of Iowa, Iowa City, Tech. Rept. CAD-SS.81-1, July 1981.

[22]Buys, J. D., "Dual Algorithms for Constrained Optimization Problems," Ph.D. Thesis, University of Leiden, England, 1972.

[23]Coope, I. D. and Fletcher, R., "Some Numerical Experience with a Globally Convergent Algorithm for Nonlinearly Constrained Optimization," *Journal of Optical Theory and Applications*, Vol. 32, No. 1, Sept. 1980, pp. 1-16.

[24]Root, R. R. and Ragsdell, K. M., "Computational Enhancements to the Methods of Multipliers," *ASME Journal of Mechanical Design*, Vol. 102, No. 3, 1980, pp. 517-523.

[25]Rosen, J. B. and Suzuki, S., "Construction of Nonlinear Programming Test Problems," *Communications of Association for Computing Machinery*, Vol. 8, 1965, p. 113.

[26]American Institute of Steel Construction, "Manual of Steel Construction," 7th Ed., 1973.

[27]Belegundu, A. D. and Arora, J. S., "Optimal Design of Framed Structures with Different Cross-Sectional Shapes and Materials," Division of Materials Engineering, The University of Iowa, Iowa City, Tech. Rept. 54, April 1979.

[28]Haftka, R. T. and Starnes, J. H. Jr., "Applications of a Quadratic Extended Penalty Function for Structural Optimization," *AIAA Journal*, Vol. 14, June 1976, pp. 718-727.

[29]Haug, E. J., "Second Order Design Sensitivity Analysis of Structural Systems," *AIAA Journal*, Vol. 19, Aug. 1981, pp. 1087-1088.