Matrix Algorithm for Structural Modification Based upon the Parallel Element Concept

Jaroslaw Sobieszczanski*
Parks College of Aeronautical Technology, St. Louis University, St. Louis, Mo.

This paper deals with the reduction of the computer time needed to perform arithmetical operations for the structural analysis consisting of an original solution of a complex redundant structure, followed by a number of consecutive modifications. A matrix algorithm is derived in the paper for this purpose from the so-called parallel element method and discussed in connection with the Force or Displacement methods used for the original solution. This algorithm is compared with the so-called "initial strain" method of modification. Computer time for all the algorithms is evaluated showing the parallel element algorithm as much more economic than the initial strain approach for a larger number of consecutive modifications. Computer accuracy tests are reported and two numerical examples are provided. Applications are pointed out for structural optimization, nonlinear analysis and computer-aided design.

Nomenclature

I = unit matrix
$B_x(l \times n)$ = matrix of element forces statically equivalent to the unit redundancies
$F(l \times l)$ = quasi-diagonal flexibility matrix of unassembled structure
$K(l \times l)$ = quasi-diagonal stiffness matrix of unassembled structure
$B_{f0}(l \times m)$ = matrix of element forces statically equivalent to the unit loadings
$f(m \times 1)$ = vector of loading parameters
$C(l \times s)$ = matrix of element forces statically equivalent to the unit dummy forces associated with the displacements d
r = number of elastic degrees of freedom (nodal displacements) that is equal to the number of load components statically permissible for the given structure
$A(l \times r)$ = matrix relating element deformations to the nodal displacements, known also as a matrix of interconnections
$R(r \times m)$ = matrix relating all the load components, statically possible for the given structure, to the loading parameters contained in f
$\Delta F(a \times a)$ = quasidiagonal flexibility matrix of the unassembled new elements
$B_{y0}(l \times a)$ = matrix of element forces statically equivalent to the interaction forces q corresponding to the unit values of y
$\bar{B}_{y0}(r \times a)$ = matrix relating all the statically possible load components to the forces of interaction q represented by y

Subscripts

1 = the structure before modification
2 = the structure after modification

1. Introduction

Recent developments in computer-aided design, structural optimization, and nonlinear analysis has brought a renewed interest in the structural modification methods. In the computer-aided design, especially if the screen-light pen type of input/output is used, the length of time required by modification of the structure displayed on screen is an essential factor in this direct man-machine contact. In structural optimization and nonlinear analysis performed by gradual changes of the initial solution, computer time required is often found prohibitive if each change of a complex redundant structure is to be accomplished by repeating the computation ab initio.

Hence, the obvious importance of the structural modification algorithms which reduce the modification computer time by several orders of magnitude. Discussion of a certain algorithm of this class in comparison with an alternative one is the subject of this paper.

2. Formulation of the Problem

A complex, redundant structure has been solved so that its element forces p have been related to the external loadings f by a matrix B:

$$p_1 = Bf \tag{1}$$

a similar relationship has been found for the nodal displacements d:

$$d_1 = F_c f \tag{1a}$$

After this solution has been obtained the structure undergoes a modification defined as follows: 1) alteration of the cross-section properties (area, moment of inertia, etc.) of one or more elements of the structure; 2) adding new structural elements between existing nodal points; 3) change in the loading cases considered.

The objective to be achieved is to compute new element forces p_2 and nodal displacements d_2 for the modified structure, by means of a number of arithmetical operations significantly smaller than that which would have to be carried out if the structure were to be completely recalculated ab initio. An additional goal is to organize the modification algorithm in a way so that it will be ready for the next modification once the current one has been completed. This is important if more than one modification is to be performed, as it takes place in the majority of practical applications. Finally the algorithm should not introduce unacceptable error into the calculation.

3. Solution of Original Structure

Let us assume that the original solution has been obtained by means of matrix force or displacement methods in the

Received May 3, 1968; revision received June 25, 1969.
* Associate Professor of Aerospace Engineering. Member AIAA.

following standard forms respectively:
force method

$$p_1 = [I - B_x(B_x{}^tF_1B_x)^{-1}B_x{}^tF_1]B_{f0}f \tag{2}$$

$$d_1 = C^tF_1p_1 \tag{3}$$

displacement method

$$p_1 = K_1A(A^tK_1A)^{-1}Rf \tag{4}$$

$$d_1 = (A^tK_1A)^{-1}Rf \tag{5}$$

Introducing notation $Z_1 = I - B_x (B_x{}^tF_1B_x)^{-1} B_x{}^tF_1$ (see Ref. 8) and $\bar{Z}_1 = K_1A(A^tK_1A)^{-1}$, one may rewrite Eqs. (2) and (4) in the form

$$p_1 = Z_1B_{f0}f = \bar{Z}_1Rf; \quad Z_1(l \times l); \quad \bar{Z}_1(l \times r) \tag{6}$$

The matrices B_x and B_{f0} considered previously may be, of course, generated automatically by a routine procedure of a "structure cutter" type.[13]

4. Modification Algorithm

Numerous papers have been published on the modification problem. Generally speaking they fall into one of three categories: 1) those based upon the concept of parallel element[1,2,4,6,9]; 2) those based upon the initial strain approach[3,5,6,7,12,13]; and 3) those based upon an algebraical approach, dealing with influence of the matrix element change on the matrix inverse.[10,11]

The initial strain approach (ISA) method is best known and is widely accepted in practice. It will be taken into account further for purposes of comparison. The parallel element concept is adopted by the present writer as a basis for a modification algorithm, developed in a new form below especially for the purpose of this paper. The methods of the last category, dealing only with a fragment of the whole problem, which contain many operations other than merely matrix inversion, are not discussed here.

4.1 Parallel Element Concept in Matrix Form

Under an approach known in the literature as a parallel element concept,[1,2,4,6] analogous to the compensation theorem used in network theory,[9] modification of a structural member is interpreted as a parallel superposition of an additional element on the altered one. Algebraical sum of the stiffnesses of the original and new elements produces the desired stiffness of the modified element.

The new element may also be added between the nodal points that were not connected directly. Thus, the algorithm is capable of handling a modification that not only alters the existing elements but also introduces the new ones.

The problem is then reduced to the calculation of the forces of interaction (q) acting between the added element and original structure from the conditions of equilibrium and compatibility at the nodes between which the new element is added. Let us satisfy first the conditions of equilibrium by selecting part of the forces q as statically independent forces y, according to the relationship

$$q = Sy \tag{7}$$

The independent forces of interaction y will be referred to as modification unknowns.

The desired modification is described by means of matrices ΔF and B_{y0} established as a known input. Creation of ΔF and B_{y0} is a relatively straightforward matter, as shown in the following examples.

Examples of the modifications by means of a parallel element are shown in Fig. 1 for the in-plane loaded framework (Fig. 1a), truss (Fig. 1b), and thin-wall structure for which Figs. 1c and 1d show the modifications of the stiffener (longeron) and shear panel, respectively.

In all the figures the parallel elements are shown together with the forces of interaction. The example of the framework may now be discussed in a somewhat greater detail in order to demonstrate the forces and matrices introduced earlier. The matrix of the interaction forces may be defined as:

$$q = \{y_1,y_2, (y_1 + y_2)/b, -(y_1 + y_2)/b\}$$

if the axial deformation is neglected as it usually is in framework analysis.

The forces y_1,y_2 may be selected as statically independent ones and grouped into a vector

$$y = \{y_1,y_2\}$$

related to q by means of Eq. 7 in which the matrix

$$S = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1/b & 1/b \\ -1/b & -1/b \end{bmatrix}$$

represents the equilibrium solution for the beam (axial force excluded) according to the sign convention shown in the Fig. 1a.

As far as matrix ΔF is concerned it represents the flexibility of the parallel element so it has to have the same character as the matrix F_{ij} of the beam. The flexibility matrix F_{ij} of the isolated beam may be written with respect to the end moments chosen as the element forces

$$(F_{ij}) = \frac{b}{6EI}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$$

Naturally, the choice of element forces for the added element and the element to be modified must be the same, so the values y_1,y_2 will also be end moments as shown in Fig. 1a.

Compatibility between the original and parallel beam requires a flexibility matrix of the latter in form

$$\Delta F_{ij} = \frac{b}{6EIW}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$$

where W = dimensionless coefficient.

The W coefficient may be found easily from the principle, according to which the flexibilities of the parallel elements are added as reciprocals.

Hence, to stiffen the beam v times, or, more precisely, to make $(EI)_2 = v \cdot (EI)_1$ one has to put $W = v - 1$: for instance to stiffen twice, $v = 2$, $W = 1$; to weaken twice $v = \frac{1}{2}$, $W = -\frac{1}{2}$, etc.

If more than one element is modified the ΔF matrix takes a quasi-diagonal form such as matrix F_1. For two (even not adjacent) beams modified for the framework

$$\Delta F = \begin{bmatrix} \Delta F_{11} & \\ & \Delta F_{22} \end{bmatrix}$$

It is important with respect to the computer time that no additional matrix operations are required to create the matrix ΔF, because its submatrices ΔF_{ij} have the same form as the matrices of elements to be modified, and therefore, may be generated by the same standard procedure that was applied for the creation of matrix F_1 in the original solution without any additional storage requirements. Thus, the coefficients W and the numbers of the elements to be modified will suffice as the description of the required modification.

The matrix B_{y0} contains the values of element forces statically equivalent to the unit values of y forces. Thus, it

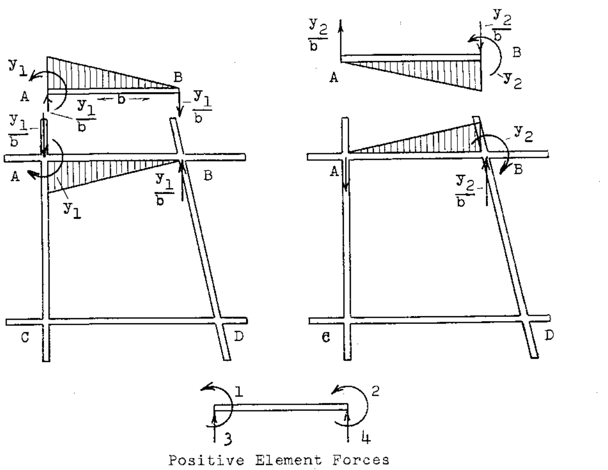

Fig. 1a Example of modification by parallel element—fragment of a framework.

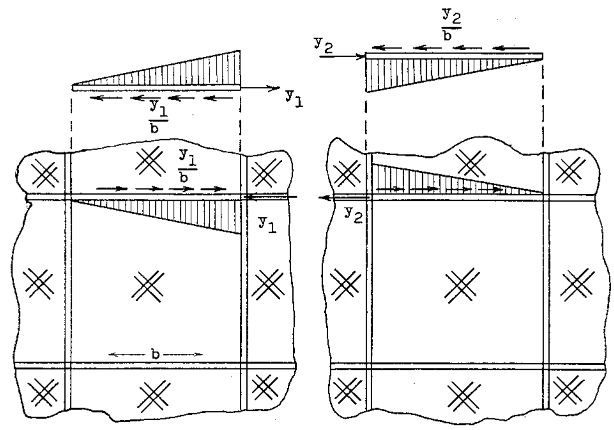

Fig. 1c Example of modification by parallel element—stiffened panel, stiffener modified.

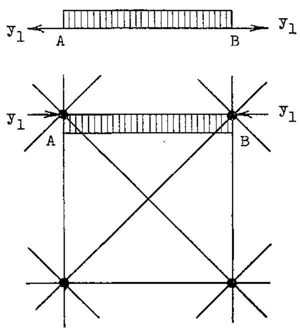

Fig. 1b Example of modification by parallel element—truss.

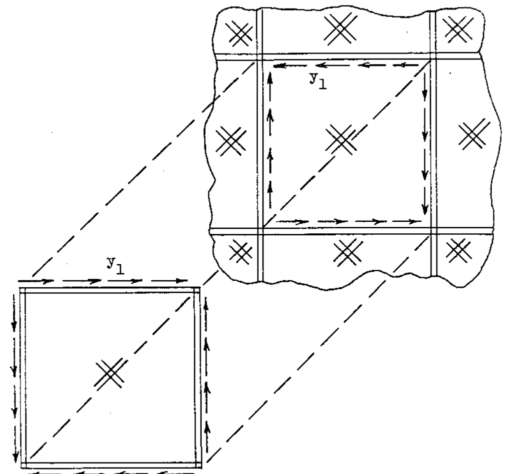

Fig. 1d Example of modification by parallel element—stiffened panel, shear panel modified.

is constituted of "l" rows and "a" columns. In the creation of this matrix one need not of course, be bound by the set of "cuts" used to establish the matrices B_{f0} and B_x of the original solution. Additional "cuts" may be introduced in order to limit the nonzero elements of the matrix B_{y0} to only those which represent the element to be modified. For the framework example (Fig. 1a), it was achieved by releasing the ends A,B by means of the fictitious pin joints. The nonzero portion of the matrix B_{y0} is then

$$B_{y0} = \begin{array}{c|cc|} & y_1 = 1 & y_2 = 1 \\ \hline & 0 & 0 \\ A & -1 & 0 \\ B & 0 & -1 \\ & 0 & 0 \\ \hline \end{array} \quad a = 2$$

This technique may always be used and its application is even more apparent for the other examples shown in Fig. 1. Should some other elements be modified simultaneously, the nonzero elements would be located in the matrix B_{y0} accordingly. Their number would always be a. There is an exception for a modification consisting in adding a new structural element where none existed before. In this case, the elements of the B_{y0} matrix may be nonzero only for that segment, or bay, of the structure which contains the new element provided that the structure is rigid in itself, as virtually all flying structures are, without attachment to an external foundation.

The practical consequence of this is that the matrix B_{y0} may be created without any additional matrix operations by a standard subroutine fed with a number of the element to be modified and a code number providing information as to the element (beam, rod, stiffener, panel, etc.). The matrix B_{y0}, being very sparsely populated, need not be stored with all the zeros. The nonzero elements are rather stored together with their addresses. Several specialized techniques[8] have been developed for such sparsely populated matrices. In conclusion one may state that no additional time for arithmetical operations is required to create the matrices ΔF and B_{y0}. There will be a certain time needed to execute the standard routines creating them, but as in all the so-called compute-bound jobs, it will be very small in comparison with the time of the operations on large matrices, involved in the original solution which would have to be repeated for each modification, should it be analyzed by recomputation ab initio.

Another practical conclusion, in such a case, is that the additional storage needed for the matrices ΔF and B_{y0} is, in view of the aforementioned considerations, negligibly small in comparison with that required by the large matrices representing the original structure, therefore their introduction does not require any additional communication with the storage devices. Obviously the modification may include simultaneous alteration of many elements all represented by matrices ΔF and B_{y0}.

A compatibility equation may now be written according to the virtual work principle, assuming that the original solution has been obtained by means of the force method (FM)

$$y^t \Delta F y + y^t B^t_{y0} F_1 Z_1 B_{y0} y + y^t B^t_{y0} F_1 Z_1 B_{f0} f = 0 \quad (8)$$

hence

$$y = -(\Delta F + B^t_{y0} F_1 Z_1 B_{y0})^{-1} [B^t_{y0} F_1 Z_1 B_{f0}] \cdot f \quad (9)$$

Element forces in the modified structure may then be expressed by superposition

$$p_{2,1} = p_1 + Z_1 B_{y0} y \quad (10)$$

for those elements or original structure which remain unaltered,

$$p_{2,2} = p_1 + Z_1 B_{y0} y - y \quad (11)$$

for those elements of original structure which were made to cooperate through parallel action with the new elements, and

$$p_{2,3} = y \quad (12)$$

for these new elements which were added where no original element existed.

Introducing matrices

$$G = B^t_{y0} F_1 Z_1 \text{ and } D = [\Delta F + G B_{y0}]^{-1} \quad (13)$$

we may express both $p_{2,1}$ and $p_{2,2}$ in form of a vector p_2 according to Eqs. (9–11)

$$p_2 = [Z_1 + (Z_1 \cdot B_{y0} - V) D G] B_{f0} f \quad (14)$$

where V is a "dispersing" zero-unit matrix, required by matrix calculus formalism to carry out the superposition (11) where the number of elements of the vector y is incompatible with the two other superimposed vectors. In actual programing the matrix V does not appear and is replaced by a proper addressing.

Equation (14) may be further abbreviated by the notation

$$Z_2 = [Z_1 + (Z_1 \cdot B_{y0} - V) D G] \quad (15)$$

which yields

$$p_2 = Z_2 B_{f0} f \quad (16)$$

The formulas corresponding to the Eqs. (15) and (16) may be easily obtained if the displacement method has been used for the original solution by substituting in Eqs. (8–11) the product $Z_1 B_{y0}$ by $\bar{Z}_1 \bar{B}_{y0}$ and $Z_1 B_{f0}$ by $\bar{Z}_1 R$.

Matrix $\bar{B}_{y0}$ replaces here the input matrix B_{y0} as the part of description of the modification and is established with respect to the forces y in similar way as matrix R is established with respect to forces f.

Matrix F_2 for the modified structure may be obtained in a straightforward manner by summation of the stiffnesses of the parallel elements

$$F_2 = [(F_1)^{-1} + H_1 (\Delta F)^{-1} H_2]^{-1} \quad (17)$$

where H_1 and H_2 play the same role as V in Eq. (14).

Operation (17), written according to the matrix formalism, is not carried out in that way in actual programing, because it may be simplified to the computation of the coefficient W, as described in the example of framework modification in Sec. 4.1 (see Fig. 1a).

Displacements of the modified structure may now be recomputed in a straightforward manner

$$d_2 = C^t F_2 Z_2 B_{f0} f \quad (18)$$

Should the loading change occur, it may easily be accounted for by a simple change of elements of matrices B_{f0} and f in Eqs. (9, 16, and 18).

Thus all the objectives set in Sec. 2 are achieved, including the last one, because Z_2 and F_2 from Eqs. (15) and (16) may be substituted for Z_1 and F_1 in Eqs. (13–15, and 17), in order to carry out the next consecutive modification.

Elements that were added to the structure where no original element existed before (let us refer to them as to the elements of category 3 have to be discussed separately. Their element forces $p_{2,3}$ are expressed by Eq. (12), only for the state of the structure after this modification in which they were added.

For each next modification the forces $p_{2,3}$ may be expressed as

$$p_{2,3} = (\Delta F_3)^{-1} [B_{y03}{}^t F_1 Z_1 B_{f0} f] - V_3 y \quad (19)$$

where the term in parentheses represents deformations of the modified structure associated with the independent forces of interaction between the elements of category 3 and the structure. Matrix B_{y03} consists of element forces in the cut structure corresponding to the unit values of the aforementioned independent forces of interaction, whereas V_3 is a submatrix of V corresponding to the elements of category 3.

4.2 Alternative Algorithm Resulting from the Initial Strain Approach

From the concept known in the literature as the "initial strain approach" one obtains forces in a modified structure as

$$p_2 = [B - B_x \bar{D}^{-1} B^t_{xj} (B_{xj} \bar{D}^{-1} + \Delta F^{-1})^{-1} B_j] f \quad (20)†$$

where the matrices

$$\bar{D} = (B_x{}^t F_1 B_x),\ B = B_{f0} - B_x \bar{D}^{-1} B_x{}^t F_1 B_{f0}$$

B_{xj} and B_j are the submatrices of B_x and B corresponding to j modified elements. The matrices $\bar{D}$ and B_{xj} need not be created for each modification since they are assumed to be available from the initial solution.

Essential differences between the parallel element algorithm [Eqs. (13, 15, and 16)] and this approach may already be seen. In the parallel element method (PEM) the original solution is updated each time the structure is modified forming a closed loop from Eq. (17) to Eq. (13).

Under the initial strain approach each consecutive modification is referred always to the very first original solution represented in Eq. (20) by the matrices $\bar{D}$ and B which remain the same each time that equation is applied to a consecutive modification. Thus, in performing modification number j we have to take into account in ΔF and B_{xj} all previous modifications from 1 to j which affected structural elements other than those altered in current modification j. In other words, even if jth consecutive modification is simple and is to alter only one member, the computation has to be conducted in such a way as if all the alterations from 1 to jth were simultaneously imposed on the original structure, in form of a single complex modification.

Therefore, the total number of simple operations (computer time) required to complete a series of j consecutive modifications by means of Eq. (20) will increase very rapidly with j (proportional to j^4 as results from detailed analysis) while it increased merely proportional to j in the parallel element algorithm. This difference is of decisive importance if the two alternatives are to be compared as shown in the next section.

To obtain a comparison between the methods of modification we shall investigate now the number of simple operations involved in the algorithm proposed here in the form of Eqs. (13) and (14), the initial strain formula Eq. (20), and original solutions Eqs. (2–5).

5. Comparison of the Required Number of Simple Operations

The number of simple operations required to perform the elementary matrix operations is presented in Table 1.[11,15]

† For derivation of the formula see, for instance, Ref. 3 or Ref. 7.

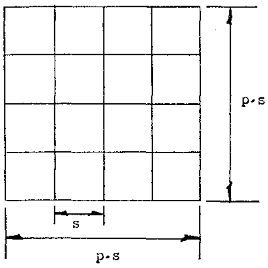

Fig. 2 Framework used as example 2.

The number of operations has been computed according to Table 1 for Eqs. (2, 4, 13, 14, and 20). According to the Section 4.1, no additional arithmetical operations for creation of the matrices ΔF and B_{y0} were taken into account.

The total computer time of the whole operation will depend not only on the number of simple arithmetical operations considered here, but may be affected by the time of communication with the storage devices. This aspect of the comparison of the methods will be taken into account in a further discussion.

The results for the comparison of the number of operations are presented in Table 2 in two versions designated * and **. Version * corresponds to all matrices treated as the full ones. In version ** the sparsity of population was taken into account for matrices F_1, K_1, and A. That sparsity is characterized by the constants c and g. The constant c denotes dimensions of the submatrices of the matrices F_1 and K_1 which are sparsely populated because of their quasi-diagonal structure. In case of a plane framework assembled of two-moment one-axial force beams (no torsion considered) each submatrix will have 3×3 dimensions, hence, $c = 3$.

The constant g denotes the number of element deformations generated by a single displacement of a nodal point. This number determines the sparsity of matrix A. For a node where four beams of a framework are interconnected (e.g., joint A, Fig. 1a) that number is four, hence the value of g in Table 2.

One has also to elaborate on the way the number of operations were counted. Namely, it is important to realize that there is no need to program the actual computation exactly in the same way which may be convenient for the derivation of the expressions involved in that computation. This would usually lead to the unnecessarily large number of operations and waste of computer time, since in the theoretical derivations one wants to keep the expressions compact, while in the actual computation one wants to minimize the number of arithmetical operations by choosing a sequence of matrix operations, to make the dimensions of the intermediate matrices as small as possible. That freedom of choice is due to the associative law for multiple matrix multiplications ($A \cdot B \cdot C$ . . . etc.) and is restricted by the storage and other constraints resulting from the particular language and hardware properties.‡

Matrix Z_1 and Z_2 [in the expressions (14) and (15)] provides good example of the preceding. They abbreviate the notation but they are large ($l \times l$) matrices and their utilization as such, in large computations would raise the highest power of l in the expressions of Table 3 from l^2 to l^3. It might also saturate the operational memory, requiring intensive communication with the storage devices. Therefore, Eq. (14) is more efficiently programed on the basis of its full, somewhat lengthy form obtained with respect to Eqs. (2) and (13);

$$p_2 = \Big([I - B_x(B_x{}^tF_1B_x)^{-1}B_x{}^tF_1] + \{[I - B_x(B_x{}^tF_1B_x)^{-1}B_x{}^tF_1]B_{y0} - V\} \cdot \{\Delta F + B_{y0}{}^tF_1[I - B_x(B_x{}^tF_1B_x)^{-1}B_x{}^tF_1]B_{y0}\}^{-1} \cdot \{B_{y0}{}^tF_1[I - B_x(B_x{}^tF_1B_x)^{-1}B_x{}^tF_1]\}\Big)B_{f0} \cdot f \quad (21)$$

In choosing the most ecnomical sequence of operations, reflected in the expressions of the Table 2, it has been assumed that usually $1 > n \gg a$, $l \gg m$, and $r \gg m$.

In Eq. (21), the matrices B_x, $(B_x{}^tF_1B_x)^{-1}$, B_{f0} are assumed available, as they have been created for the original solution. F_1 is also known either for the original structure, if the modification is a first one, or from Eq. (17), if it is not. In larger computations, a communication with the storage devices may be necessary in order to make these matrices available. Additional computer time required by this is not reflected in Tables 2 and 3 but is discussed separately in Sec. 6.

Since for simple modification $a \ll n$ and $a \ll r$ the two considered modification algorithms clearly will require computer time much shorter than that which would have to be used for repeating the original solution. It may also be noted how the sparsity of population of the matrices may reduce the number of operations. The required computer times of the arithmetical operations may be evaluated and compared for each individual case by means of the aforementioned formulas for each algorithm in question [including force vs displacement method (DM)]. Two numerical examples shown in Tables 3 and 4 provide an idea as to the order of magnitude of the computer time reduction. The semimonocoque fuselage used as example 1 (Table 3) has been idealized as an assemblage of shear panels (one shear flow), longerons (two axial end forces), and straightened ring segments (two axial end forces and two end moments). Three elastic degrees of freedom (radial and axial displacement, and rotation in the ring plane) have been assumed for each nodal point (ring-longeron intersection) and the following symbols were used: z (number of rings), u (number of longerons), hence the number of panels is $u(z - 1)$. It has also been assumed that each modification alters a single shear panel only. To make the comparison possible the time required for simple addition of two numbers has been denoted t and one has assumed that simple multiplication required $3t$ (this is an average estimate; for IBM 360-50, for instance, it is $3.15t$; see Ref. 14). The numerical estimates shown in Tables 3 and 4 apply to the relatively large computational problem (p,u,z larger than 4), for which mainly the whole discussion is relevant. This permits one to neglect the lower power terms in the formulas of Table 2, hence the compact expressions in Tables 3 and 4.

As found from the examples in Tables 3 and 4, application of a proper modification algorithm provides very radical computer time savings. Time savings increase with the number of consecutive modifications and with the number of structural elements.

The influence of the parameter a is shown in Table 3 for example 1. As one could expect, the larger the a (more elements altered in a single modification) the smaller the time savings. For each combination of the original structure

Table 1 Number of simple arithmetical operations in matrix operations

Matrix operation	Simple arithmetical operation: Multiplication	Addition	Formation of reciprocals
${}_mA^n + {}_mB^n$	0	mn	0
${}_mA^n \cdot {}_nB^p$	mnp	$mp(n-1)$	0
${}_nA^{n\,-1a}$	$n^3 - 1$	$n^3 - 2n^2 + n$	n

^{a} Gauss elimination method.[20]

‡ It is well known to those involved in numerical work, but disregarded in mathematical texts, that number of operations to multiply for instance ${}^{10}A^{10} \cdot {}^{10}B^{10} \cdot {}^{10}C^{1}$ is much smaller for $A(BC)$ than for $(AB)C$.

Table 2 Number of simple arithmetical operations for: force method (FM), displacement method (DM), parallel element method (PEM), initial strain approach (ISA)

Method	Eq.	Version	Simple arithmetical operation: Multiplication	Addition	Recipr.
FM	(2)	*	$l^2(3n+1)+2n^2l+n^3+lm-1$	$3l^2n+n^2(2l-3)+n^3+n(1-2l)+l(m-1)$	n
		**	$l^2(n+1)+2cln+2ln^2+n^3+lm-1$	$l^2n+n^2(l+\bar{c}-2)+n^3+2\bar{c}n(l-1)+l(m-1)+n(l-1)$ [b]	n
DM	(4)	*	$l^2(r+1)+r^2(l+1)+r^3+r\cdot(l+m)$	$l^2(r-1)+r^2(l-1)+r^3+l(r-2)+r(m-1)$	r
		**	$r^3+r^3(g+1)+r(l+lc+m+g)+l-1$ [a]	$r^3+r^2(\bar{g}-1)+r(\bar{c}l+\bar{g}+m-1)+1\bar{c}-\bar{g}-2\bar{c}$ [b]	r
PEM	(13)	*	$2l^2(a+1)+a^2(2l+1)+a^3+l(a+m)-1$ [b]	$2l^2(a+1)+l(m-3)+a^3$	a
	(14)	**	$l^2(2a+1)+2a^2l+al\cdot(c+m+1)+am+a^3$ [a]	$l^2(2a+1)+a^2l-al(2-\bar{c})+a^3-a^2+a(1+ml)+l-a\bar{c}$ [b]	a
ISA	(20)	*	$2a^3+a^2(n+1)+n^2(a+1)+(m+n)(a+l)$	$2a^3-3a^2+n^2(a+1)+m(a+l)+n(a^2+l-3)-1$	$2a$

[a] Coefficient $c = 1, 3, 6$, respectively, for truss, framework composed of members with no torsional stiffness, and framework composed of members possessing a finite torsional stiffness.
[b] For a framework composed of members possessing a finite torsional stiffness $g = 4$, $\bar{g} = g - 1$, $\bar{c} = c - 1$.

parameters l,m,n (or l,r,m), one could compute a "break even" value of parameter a above which the recomputation ab initio becomes more economical in terms of the number of arithmetical operations, than the modification method. This break even value, however, would be of rather academic interest, since the number of elements altered in a routine step-by-step design process is rather small, for practical reasons.

As far as the number of loading cases m is concerned, its influence on the results shown in Tables 3 and 4, is imperceptible if l is large, because in the formulas of Table 2 it is associated with lower power terms and usually, $m \ll l$.

As far as the comparison between the parallel element and initial strain approach is concerned, each of them may be advantageous for a particular combination of the parameters.

One may compare the computer time of the arithmetical operations for the two methods under standardized conditions, using the example 2 of a framework shown in Fig. 2. For this framework $l = 12(p^2 + p)$, $n = 6p^2$, $r = 6(p + 1)^2 - 6$. Single loading case and modification altering one beam only have been assumed, hence $m = 1$ and $a = 6$. Time needed to complete j consecutive modifications, each of them affecting another member, is evaluated. The result is plotted on Fig. 3, which shows that under these conditions the initial strain approach is more economic than the parallel element concept, for limited (up to 17) number of subsequent modifications; but for the large number of modifications the time required by it increases rapidly for the reasons discussed in Sec. 4.2 and exceeds time required by the parallel element method.

6. Storage Access Time

As it has been pointed out in Sec. 5 [Eq. (21)], a certain amount of information may have to be retrieved from the storage devices for execution of each consecutive modification. It appears from the Eq. (21) that the matrices F_1,B_x,B_{f0} and $(B_x{}^tF_1B_x)^{-1}$ are the ones to be retrieved for the FM/PEM mode of operation. The matrix F_1, however, is so sparsely populated that not only need it not be transferred to the storage, but it may not, and in an efficient program should not, appear as a matrix at all. It may be replaced by a special routine which is called whenever a matrix operation involving F_1 is to be performed. The routine is simple to write on the basis of a quasi-diagonal structure of the matrix F_1, consisting of the submatrices, which may be written in a typical form:

$$\begin{pmatrix}\text{flexibility}\\ \text{parameter}\end{pmatrix}\cdot\begin{pmatrix}\text{dimensionless matrix standard for the}\\ \text{structural element of a given type}\end{pmatrix}$$

for instance

$$\frac{b}{6EI}\cdot\begin{bmatrix}2 & -1\\ -1 & 2\end{bmatrix}$$

for a beam as in example Fig. 1a. Thus, all that is needed is a value of that flexibility parameter and sort of a code number identifying the type and consecutive number of an element, in order to associate the flexibility parameter with the proper standard dimensionless matrix built into the operational part of the program.

Thus, only 2 numbers per structural element have to be stored instead of the full matrix F_1. Let l_1 denote twice the number of elements. That leaves the matrices B_x,B_{f0}, and $(B^tFB_x)^{-1}$ to be retrieved. They have the dimensions $(l \times n)$, $(l \times m)$, and $(n \times n)$, respectively, hence the number of words to be transmitted is

$$N = n(l + n) + lm + l_1$$

Table 3 Example 1: semimonocoque fuselage

Computer time, t: *FM[a]	*DM[a]	Reduction of computer time: $\frac{*PEM^a}{*FM^b}$	$\frac{*PEM^a}{*DM^b}$	$\frac{ISA^a}{*FM^b}$	$\frac{ISA^a}{*DM^b}$
$648(uz)^3$	$948(uz)^3$	$\frac{0.60}{uz}(a+1)$	$\frac{0.415}{uz}(a+1)$	$\frac{0.0062}{uz}(a+1)$	$\frac{0.0042}{uz}(a+1)$

[a] For the particular example $r = 3uz - 6$; $l = 7uz - 3u$; $n = uz - 2u + 6$; $m = 1$.
[b] Reduction of computer time $= \dfrac{\text{time of modif. algorithm}}{\text{time of repeated orig. sol.}}$.

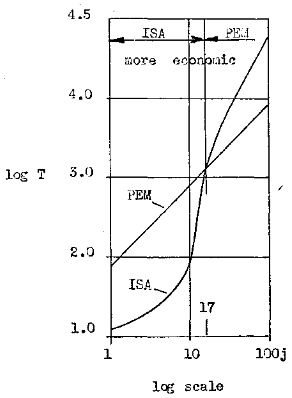

Fig. 3 Comparison of the computer time $= T \cdot 10^{-6} \cdot t$ required to complete j consecutive modifications.

Time for the retrieval T is the sum of the reading time T_R and average access time T_a to the first word of the array. The first one is N/reading rate, the second is a constant.

It is useless to attempt a general discussion of the influence of the retrieval time because of the large variety of storage devices, which create innumerable combinations with the various digital computers. It may, however, be interesting to refer to a particular numerical example to demonstrate the orders of magnitude. Assume the use of example 2 (Table 4) with $p = 5$ and $m = 3$, computer IBM/360-50, and storage device IBM/2911 Disc Storage Drive.

According to Ref. 14, the unit t (time of a simple addition) is equal to $6.88 \cdot 10^{-6}$ sec, $T_a = 75.10^{-3}$ sec, and reading rate $= 156.10^3/4$ words/sec. All this data is applicable for a single precision computation.

From Table 2, time for the arithmetical operations may be obtained for a single modification by means of PEM:

$$T_{PEM} = 4032.5^4 \cdot 6.88 \cdot 10^{-6} \text{ sec} = 17.35 \text{ sec}$$

Retrieval time T is for $N = 6 \cdot 5^2[12(5^2 + 5) + 6 \cdot 5^2] + 1080 + 60$

$$T = \frac{(150 \cdot 510) + 1080}{156/4} \cdot 10^{-3} + 0.075 = 2.055 \text{ sec.}$$

So it constitutes about 11% of the arithmetical time. Thus, the actual total computer time of the arithmetical operations and the retrieval will be 19.38 sec. This gives an idea as to the deviation from Table 3 results for the computer time reduction, which are to be expected due to the retrieval time. Similar results may be obtained for DM and ISA. From the preceding, one may conclude that the time for the arithmetical operations dominates, so that the influence of the time of the storage communications, however perceptible, seems not to be strong enough to qualitatively change the results shown in Tables 3 and 4.

To complete this consideration, one has to mention a case when matrices are so large that some of the operations in Eq. (21) have to be carried out by means of partitioning and storage/retrieval of the intermediate results. However, this will cause the time to increase in absolute values, similarly the total time of the original solution will increase; therefore, it seems reasonable to expect no significant change in the relative reduction of computer time as defined in Tables 3 and 4 with the corrections resulting from the particular example given. Practical significance of a more general theoretical investigation of the matter is doubtful due to a very large variety of possible program organizations for this case of extremely large matrices.

Another example of the actual computer time may be given for the fuselage—example 1 of Table 3. For instance for $z = u = 12$ and unit $t = 6.88 \cdot 10^{-6}$ sec (IBM 360-50, Ref. 14) one obtains the actual computer times: time for FM = 1330 sec = 22.1 min. Ratio *PEM/*FM for $a = 1, 2, 9$ is, respectively, 0.003, 0.0045, 0.03; hence time for PEM modification is 6 sec, 9 sec, 60 sec. This example demonstrates the practical significance of the modification time reduction problem, for instance, to a designer using a computer as a design aid, by means of a light pen/screen, as an input/output device. For a designer, a 22.1 minute wait (recomputation ab initio) to see the result of the modification on the screen would be rather unacceptable, whereas a less than 10 sec period (special modification algorithm) is felt as almost instant reaction, radically expediting the whole process.

7. Accuracy of the Modification Algorithm

There has been a relatively large experience accumulated as to the initial strain approach including accuracy data. Therefore, the accuracy consideration were limited within the present work to the proposed algorithm of a parallel element. Accuracy was tested by means of computer experiments on typical structures.

The program written in FORTRAN IV as a standard subroutine stored on a disk was tested on an IBM 360-50 for redundant truss, framework, and semimonocoque structures, thus all three of the most important main types of structures were tried. Several variants of the tests were performed as follows: 1) first the internal forces for several variants of member flexibility distribution were calculated directly for each structure; 2) then the same variants were examined as created by consecutive modifications starting from a certain original structure and solutions were compared with the direct ones. Each modification chain was closed by a modification providing a return to original state. The number of modifications in one chain "from original to original" varied from 5 to 65. Member flexibility was modified by a factor varying from 10^5 (practical removal of a member) to 10^{-5} (practical rigidization).

The number of elements altered in one modification varied from one to all members of the modified structure. The largest error did not exceed 0.1/1000 and the errors indicated random scatter with no accumulation of some particular tendency.

Accuracy of the method does not seem to be affected at all by the magnitude of modification (relative flexibility change, number of modified members) and is more than satisfactory for engineering applications.

8. Conclusions

The following conclusions may be significant for the organization of the matrix algorithms for such problems as structural optimization, nonlinear analysis by means of a

Table 4 Example 2: framework

	Computer time of arith. operations, t FM	DM	PEM	ISA
*	$14700p^6$	$7500p^6$	$8064p^4$	$1008p^4$
**	$7400p^6$	$1400p^6$	$4032p^4$	...
	Reduction of computer time[a]			
	$\frac{*PEM}{*FM}$	$\frac{**PEM^b}{**FM}$	$\frac{*PEM}{*DM}$	$\frac{**PEM}{**DM}$
	$\frac{0.5}{p^2}$	$\frac{0.58}{p^2}$	$\frac{1.08}{p^2}$	$\frac{2.9}{p^2}$
	$\frac{ISA}{*FM}$	$\frac{ISA^c}{**FM}$	$\frac{ISA}{*DM}$	$\frac{ISA}{**DM}$
	$\frac{0.068}{p^2}$	$\frac{0.145}{p^2}$	$\frac{0.134}{p^2}$	$\frac{0.72}{p^2}$

[a] Reduction of computer time $= \frac{\text{time of modif. algorithm}}{\text{time of repeated orig. sol.}}$.
[b] For instance if $p = 10$; **PEM/**FM = 0.0058.
[c] For instance if $p = 10$; ISA/**FM = 0.00145.

step-by-step modification of the initial linear solution, or computer-aided design.

1) The proposed modification algorithm may cooperate with both force and displacement methods being used for the original solution and may also handle entirely new elements attached to the structure. The algorithm provides very significant computer time reductions of the order 100, 1000, or more for large structures. Its accuracy was tested and found very good. Substantial time reductions may also be achieved due to the sparsity of matrix population and proper use of the associative law of matrix multiplication.

2) Comparison of the algorithm with the commonly known method of initial strain approach shows the latter as more economical for a small number of modifications. For larger numbers of consecutive modifications, however, the parallel element method provides larger time savings and the difference increases very rapidly in its favor with the increasing of that number.

References

[1] Cicala, P., "Effect of Cut-outs in Semimonocoque Structures," *Journal of the Aeronautical Sciences*, Vol. 15, No. 3, March 1948, p. 171.

[2] Michielsen, R. F. and Dijk, A., "Structural Modifications in Redundant Structures," *Journal of the Aeronautical Sciences*, Vol. 20, No. 4, April 1953, p. 286.

[3] Argyris, J. H. and Kelsey, S., *Modern Fuselage Analysis and the Elastic Aircraft*, Butterworths, London, 1963.

[4] Grzedzielski, A. L. M., "Note on Some Applications of the Matrix Force Method of Structural Analysis," *Journal of the Royal Aeronautical Society*, Vol. 64, No. 594, Aug. 1960.

[5] Argyris, J. H. and Kelsey, S., "Initial Strains in the Matrix Force Method of Structural Analysis," *Journal of the Royal Aeronautical Society*, Vol. 64, No. 596, Aug. 1960.

[6] Argyris, J. H., Kelsey, S., and Grzedzielski, A. L. M., "The Discussion on the Initial Strain Concept," *Journal of the Royal Aeronautical Society*, Vol. 65, Feb. 1961, pp. 127–137.

[7] Pestel, E. C. and Leckie, F. A., *Matrix Methods in Elastomechanics*, McGraw-Hill, New York, 1963.

[8] Argyris, J. H., *Recent Advances in Matrix Methods of Structural Analysis*, Pergamon Press, London, 1964, p. 2.

[9] MacNeal, R. H., "Application of the Compensation Theorem to the Modification of Redundant Structures," *Journal of the Aeronautical Sciences*, Vol. 20, No. 10, Oct. 1953, pp. 726–727.

[10] Kosko, E., "Effect of Local Modifications in Redundant Structures," *Journal of the Aeronautical Sciences*, Vol. 21, No. 3, March 1954, pp. 206–207.

[11] Sack, R. L., Carpenter, W., and Hatch, G., "Modification of Elements in the Displacement Method," *AIAA Journal*, Vol. 5, No. 9, Sept. 1967, pp. 1708–1710.

[12] Pipes, L. A., *Matrix Methods for Engineering*, Prentice-Hall, Englewood Cliffs, N.J., Chap. 6, Sec. 10–17.

[13] Przemieniecki, J. S., *Theory of Matrix Structural Analysis*, McGraw-Hill, New York, 1968, p. 204.

[14] IBM 360-50 File S360-01 Form A22-6891-1 and IBM SYS/360 #520-0903, International Business Machines, p. 13.

[15] Fox, L., *An Introduction to Numerical Linear Algebra*, 1st ed., Oxford University Press, New York, 1965, Chap. 7, pp. 175–178.

Bibliography

Best, G. C., "The Stress Area Method of Designing Beams," *Journal of the Aeronautical Sciences*, Vol. 12, No. 3, July 1945, pp. 298–304.

Martin, H. C., *Introduction to Matrix Methods of Structural Analysis*, McGraw-Hill, New York, pp. 182–188.

von Hoerner, S., "Homologous Deformations of Tiltable Telescopes," *Proceedings of the American Society of Civil Engineers, Journal of the Structural Division*, Appendix A, No. ST 5, Oct. 1967, pp. 461–485.

Poppleton, E., "The Redesign of Redundant Structures Having Undesirable Stress Distributions," *Journal of the Aerospace Sciences*, Vol. 28, No. 5, April 1961, pp. 347–348.

Poppleton, E., "A Note on the Design of Redundant Structures," UIIA TN 36, July 1960, University of Toronto.