NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Report 32-1075

Summary of the Functions and Capabilities of the Structural Analysis and Matrix Interpretive System Computer Program

T. E. Lang

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

April 1, 1967

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Report 32-1075

Summary of the Functions and Capabilities of the Structural Analysis and Matrix Interpretive System Computer Program

T. E. Lang

Approved by

M. E. Alper

M. E. Alper, *Manager*
Applied Mechanics Section

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

April 1, 1967

TECHNICAL REPORT 32-1075

Prepared Under Contract No NAS 7-100
National Aeronautics & Space Administration

Foreword

The Structural Analysis and Matrix Interpretive System (SAMIS) Computer Program described in this report was developed by Philco Corporation, Western Development Laboratories (WDL), Palo Alto, California, under contract to and in association with the Jet Propulsion Laboratory, Pasadena, California (JPL Contract No. 950321). Development work by WDL was supervised by P. R Cobb and R J Melosh as project engineer. The support effort by JPL was under the supervision of R R McDonald and M E Alper with T. E Lang as project engineer Engineering and programming support at WDL was provided by H. N. Christiansen, D. A. Diether and (Mrs) M Brennan Corresponding support at JPL was provided by L W Schmele, S. Utku, V. C Smith, and R. E Reed

Under a contract with NASA, the University of Georgia has established a center for the dissemination of computer programs and computer information. This center, known as Computer Software Management and Information Center (COSMIC), is working through the NASA Technology Utilization Office in conjunction with other NASA Centers and NASA Headquarters Through this joint effort, computer programs and computer information developed by or for NASA will be made available to any requester

Readers of this publication who desire further information on obtaining the SAMIS Computer Program should direct their inquiries to COSMIC at the University of Georgia Inquiries should be addressed as follows

COSMIC
COMPUTER CENTER
University of Georgia
Athens, Georgia 30601

Page intentionally left blank

Page intentionally left blank

Contents

Contents (contd)

Figures

Abstract

The functions and capabilities of a large capacity Structural Analysis and Matrix Interpretive System (SAMIS) Digital Computer Program developed to analyze frame and shell-type structures are described. Included is a description of each subprogram function with associated time-performance capabilities defined, as established by program usage at a particular computer installation. Program development considerations given to modularization of the program for functionally-diverse applications and reduction of errors in program usage are outlined. Finally two brief sections are included on program extensions currently under development, and the participation of engineering personnel in solving structural problems with the SAMIS Computer Program.

Summary of the Functions and Capabilities of the Structural Analysis and Matrix Interpretive System Computer Program

I. Introduction

This report describes the general characteristics and functions of the recently developed Structural Analysis and Matrix Interpretive System (SAMIS) computer program The program is designed to solve problems involving matrix arithmetic, with particular emphasis on structural applications The program can execute, either exclusively or sequentially, two basic operations. From input data that defines an idealization of a structure, the program generates structural matrices for any type of element available in the program element library This operation is designated the generation phase The second basic operation is termed the manipulative phase, in which either generated or input matrices are manipulated according to the rules of linear algebra In structural problems, the matrix manipulations may be sequenced to compute displacements, stresses, reaction faces, or mode shapes and frequencies. The ability to compute these quantities for structural systems which are described by a large number of simultaneous equations requires greater than in-core data access and storage capacity Because of this requirement, the program was developed as a chain system as defined by specifications of the FORTRAN II computer language operating under the IBSYS operating system Based mainly upon the constraint of computer running time, the SAMIS program operates efficiently with matrices ranging from 100th to 2,500th order

The generation phase of the program is based upon the structural concepts of the finite element method, in particular, the stiffness or displacement method To enable the program to analyze a range of structural types (truss, plate, shell, composite shell-beam etc), several elements are programmed and cataloged in the program element library Contained in the library are the general line element suitable for representing axial, bending, and torsion deformations, and the triangular plate element which models membrane and bending deformations.

A checked-out version of the SAMIS program that contains a reasonable capability to solve structural problems is described in this report This version of the SAMIS program is documented to aid users in gaining an understanding of its operation Technical aspects of the program, including definition of the algorithms used in the manipulative subprograms, derivation of the element stiffness, stress, and loading matrices in the generation link, and discussion of error control in program usage are discussed in Ref. 1. Programming aspects, including definition of input-output format and content, description of each chain link function, definition of subroutines within each link, and description of overall system logic and flow, are provided in Ref. 2 The set up and solution of typical shell and beam structural problems are reported in Ref 3

Follow-on development of additional links for the SAMIS program, to be phased into the released version of the program, is also outlined in this report These links complement the released version of the program to provide greater capability in structural analysis It is planned to release these additional checked-out links and supplementary documentation on (approximately) a yearly basis

The purpose of this report is to provide an overall description of the SAMIS program omitting detailed descriptions of mathematical algorithms, computer operations, and input-output formats It is not intended for reference in using the program For this purpose, the program user will find Refs 1–3 of value

In the following sections, the generation phase of the SAMIS is outlined, individual manipulative links are described, major concepts in program development are discussed, and the degree of participation by technical personnel in applying the program to engineering problems is outlined

II. General Description of Program Characteristics

The SAMIS program is a segmented or chain system within the guidelines of FORTRAN II computer language The program is composed of sixteen segments or links, the selection and sequencing of which are user-controlled and program-activated by a Master Intelligence link, MINTS Modularity is basically by matrix-manipulative, structural matrix-generation, and data-handling functions

Sequencing the computational steps to solve a problem solution is accomplished by writing a set of instructions This set of instructions is called the "pesudo instruction program." Conceptually, pseudo instructions are quite similar to FORTRAN instructions, the major difference being that a pseudo instruction calls for a set of subprograms to perform a matrix operation rather than defining each step of the operation. One pseudo instruction is required for each matrix operation to be performed (excepting the "serial" options) For example, to multiply two matrices, one pseudo-instruction is needed which contains information on where the matrices are located (either in-core or on prescribed tapes), what operation is to be performed (multiplication, in this instance), and where the resultant (product) matrix is to be stored (in-core or on tape). A sample pseudo instruction for this operation is shown in Fig. 1

Interpretation of this instruction is. read into core matrix KPR001, which is on tape 9, location 001, and multiply it by matrix MRC001, which is in core Designate the product matrix PRR001 and store it on tape 10, location 003

The Master Intelligence System (MINTS) controls the execution of the pseudo instructions For the multiplication operation outlined above program flow is as follows:

(1) Upon completion of the previous pseudo instruction, the MINTS link is read into core from the SAMIS library tape

(2) MINTS reads the next pseudo instruction (multiplication instruction) from the pseudo instruction

				MBC001			ROWS	11003	MBR002	
			11002	MBR001	11003	MBR002	ADDS		MRR001	
				MRR001			COLS		MRC001	
			9001	KPR001		MRC001	MULT	10003	PRR001	
			10003	PRR001	9002	FXC001	CHOL	11001	DIC001	
			11001	DIC001			INKS			
							HALT			

Fig. 1. Sample pseudo instruction

program tape and determines which tapes are needed and what function is to be performed

(3) MINTS positions all of the tapes involved (tape 9 at location 001, tape 10 at location 003), so that the next location on the tape is either the start of the input data for the operation or the designated position for the output data

(4) MINTS then locates on the SAMIS library tape the operation link (MULT) and brings the link into core

(5) Control is then shifted to the operation link (MULT), which calls for the input matrices (KPR001, MRC001), performs the calculation (multiplication), and locates the resultant matrix (PRR001) in core or on tape, as specified

(6) Control is then returned to the MINTS link, and the process is repeated with the next pseudo instruction

Generally, a pseudo instruction program for structural analysis varies little from problem to problem, so that, once an efficient program is written, it becomes a standard part of the input data for similar structural problems

It can be observed from the sample pseudo instruction that specification of data storage tapes is an integral part of each pseudo instruction This requires greater user knowledge in setting up a pseudo instruction program than if tape assignments were specified internally by the program, however, the system has greater applicability through use of this scheme. By user discretion, key data can be stored on tapes to be saved after completion of computations on the computer These data are available for subsequent runs, thus avoiding complete regeneration of data For example, in a structural problem, the summed stiffness matrix would be saved if any one of the following conditions can be anticipated· a nonrecoverable error occurring in calculations that follow the generation phase, elements of the structure likely to be redefined subsequent to current calculations, or additional loading states likely to be defined subsequent to current calculations. Another advantage in assigning tapes is that the program can be used on different computer systems and is still operable, by modification of the pseudo instructions, when one or more tape units are removed from the system for repair.

In assigning data storage tapes in a pseudo instruction program, it is more efficient to store data on a number of tapes rather than on one or two, since tape search time for particular data is reduced if the number of data groups on the tape is small In the JPL computer system, seven tapes are available for data storage during program execution (Fig 2) For the moderately long problems that were run to check out the SAMIS program (20–50 pseudo instructions), three to five tapes were used, one or two of which were saved for recovery purposes

For a computer complex with an adequate number of available tapes, the SAMIS program library (16 links) should be divided into two or more tapes to reduce search times For example, the JPL computer system has 16 tape consoles, and two tapes are available to store the SAMIS library (Fig 2, tapes A4 and B2). Because of the frequent use of MINTS, this link is stored alone on one of the library tapes. At program initiation, MINTS is read into core, and takes control of the computer to perform functions already described

Because of the sequential nature of the calculations in the pseudo instruction program, it is possible to restart calculations at any point in the pseudo program provided the data generated to this point has been saved on tape This feature of the SAMIS is termed the "recovery feature" and is predicated upon the writer of the pseudo instruction program planning, in advance, recovery points based upon likely locations for errors in the calculations. To support this feature, computer operators may be instructed to save certain tapes for re-use the next time the program is run.

Related to recovery is the recoverable error option In the SAMIS program, it is possible to mistakenly specify a matrix operation assuming in-core matrix sizes, and to find later during execution that the matrices are larger-than-core If the pseudo instruction program does not cover this eventuality, the run is terminated However, if an ERRS instruction, followed by an alternate set of pseudo instructions, is inserted in the program the run can be continued The ERRS instruction is actually a branching mechanism to redirect the sequence of pseudo instructions followed by MINTS, should a primary set of instructions fail to apply This error option can be automated in the program by action based upon the result of a test on matrix size, however, because of the sacrifice in core space and the low probability that matrices of sizes bordering on core capacity might occur without prior user knowledge, error recovery was made a user option

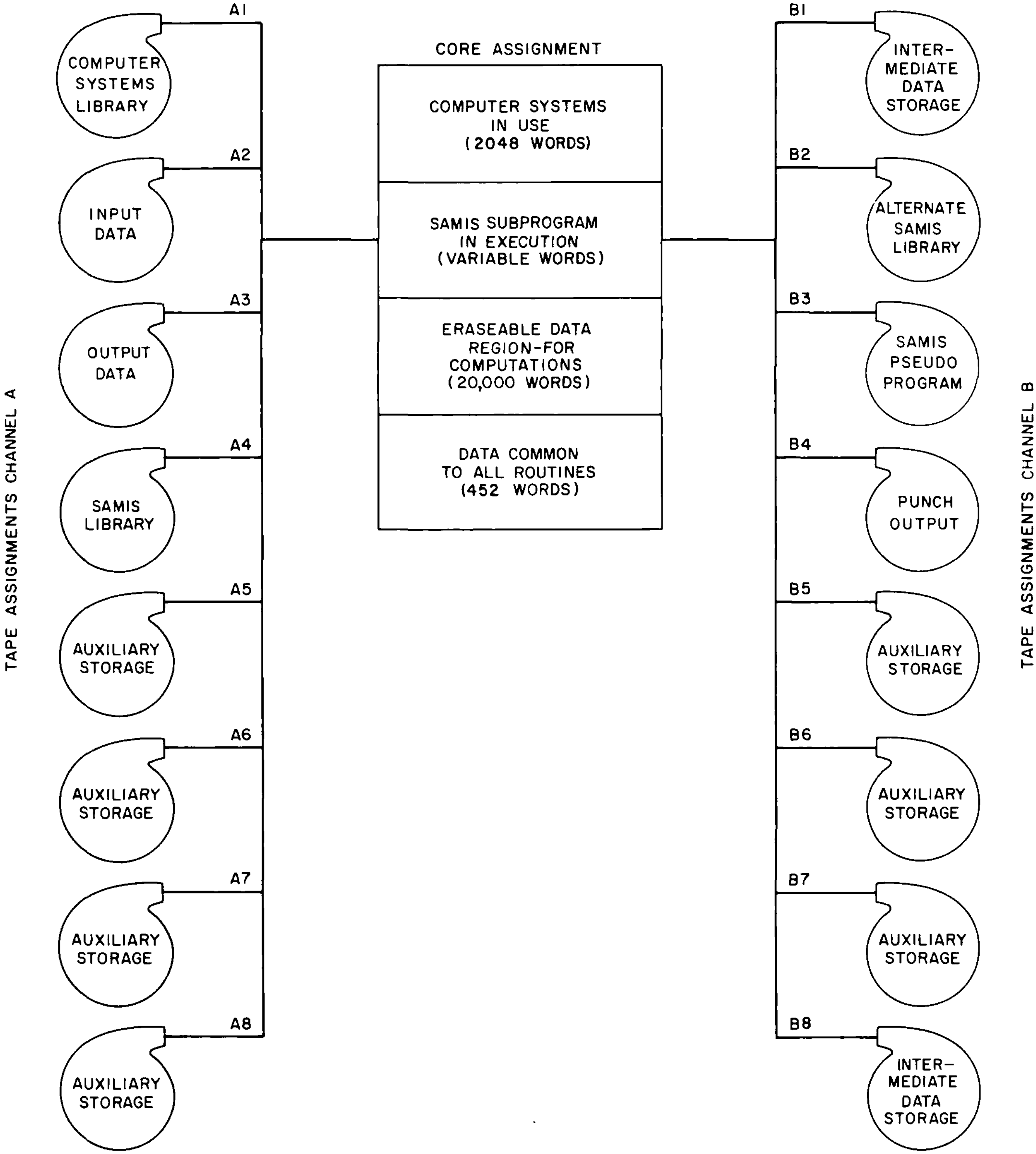


Fig. 2. Core and tape assignments of the SAMIS program on IBM 7044/7094 computer

To completely automate a structural analysis, the computer program must be set up to perform two basic functions, namely, generate the stiffness, stress, loading, and mass matrices, and then perform the manipulations necessary to solve the problem. Having summarized the overall program logic of the SAMIS program, we now can proceed to describe the generation and manipulative phases in some detail.

III. Functions and Capabilities of the Program Generation Phase

The first function, the generation of equation coefficients, is performed by the computer when geometric and materials data are input to the program. From these input data the SAMIS program will generate the following:

(1) Element stiffness and stress matrices.

(2) Fixed-node forces due to temperature distribution and gradients.

(3) Equivalent gridpoint forces due to uniform pressure loads.

(4) Gravity loading vectors from imposed accelerations.

(5) Element mass matrices for uniformly-distributed mass within each element.

This generation capability is represented in the SAMIS program by one link, given the code name BILD. Any one or all of the element matrices listed above can be generated in BILD for each type of element in the program element library.

In the development of the SAMIS program, the initial objective was to analyze regular, as well as irregular, thin plate and shell structures which may have stiffening or support beam structural attachments. Therefore, effort was devoted to the development of a flat, thin, triangular plate element of arbitrary mid-plane shape and uniform thickness. If the triangular element adapted for the program is oriented arbitrarily with respect to a set of overall coordinate axes, six variables describe the deflection state at each apex (three orthogonal displacements and three orthogonal rotations) as shown in Fig. 3. Shown in Fig. 3 are the generalized forces and moments that correspond with each displacement variable. If a number of these triangles are set in an array in which the gridpoints lie on the neutral surface of a shell, the polyhedral system might appear as in Fig. 4.

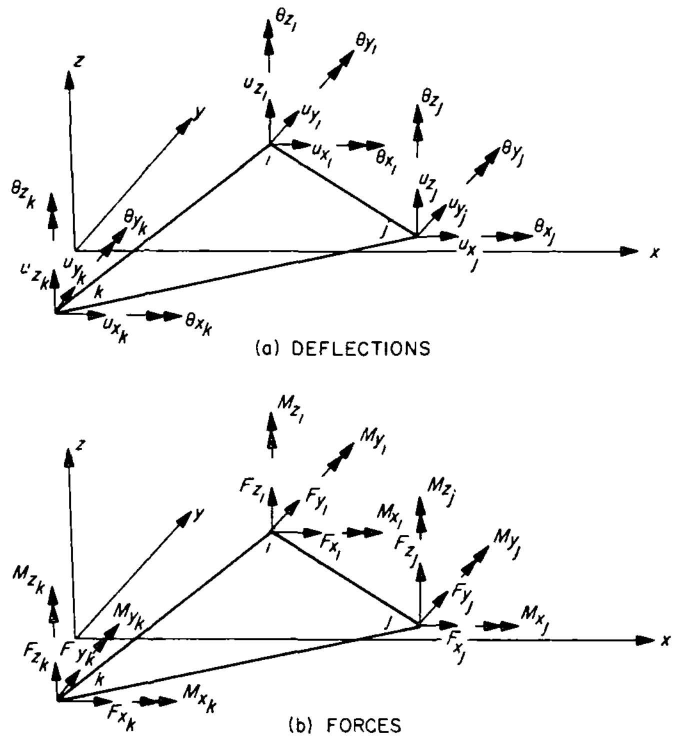


Fig. 3. Gridpoint deflections and forces

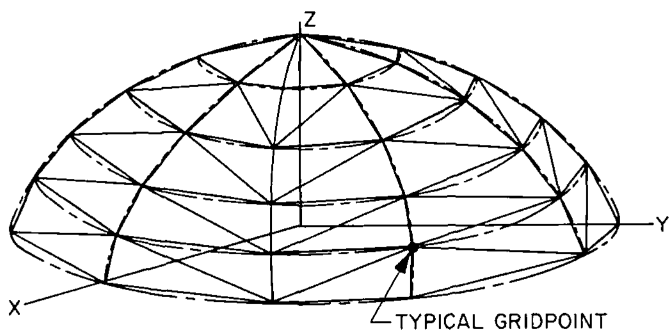


Fig. 4. Triangular grid array of a spherical shell sector

Clearly, for this idealization to be accurate each triangular element must represent membrane as well as bending states of deformation. The numerical results reported in Ref. 3 demonstrate this capability.

The triangular plate element can represent a structure having monotropic material properties (13 independent constants in the constitutive equations), varying elastic moduli with temperature (interpolation and extrapolation of a material properties table), and varying thickness and/or density properties. A restriction on this element representation when the material is nonisotropic is that the principal material axes must align with prescribed geometric axes unless a transformation of the constitutive equations is made prior to introduction into the SAMIS program

To represent stiffness of frames and trusses the general line element is universally applicable This element represents axial deformation, bending deformation in two orthogonal planes, and torsional deformation It can have arbitrarily shaped cross-sections, provided the shear center and/or twist center are coincident with the principal longitudinal geometric axis This element representation optionally includes the effects of shear deflection and rotary inertia, and idealized types of end fixity can be treated

To extend the applicability of the triangular plate and line elements in structural idealizations, three supplemental program features were incorporated in the BILD link.

(1) Capability was provided to handle "substitute gridpoints," i e , gridpoints that do not lie on the elastic axis or plane of the element, but connect to the element through weightless infinitely-rigid links The substitute gridpoint concept is applied in idealization of layered or offset stiffened structures

(2) Program changes were effected to represent the "gridpoint discontinuity condition," which is essential when idealizing hinge or ball-socket joints in structures With structural joints of these types, certain displacements are discontinuous across the joints, and the structural stiffness matrices must be modified to represent these conditions

(3) The line element stiffness equations were rederived and programmed to account for planar shear stresses acting on the edges This model is useful in representing shear panel and spar elements in built-up structural configurations.

These three supplemental element features, if interpreted as separate element representations, increase the entries in the program element library significantly.

Several different mass matrices can be generated in BILD, depending upon user preference: Potential-energy mass matrices (Ref 4), modified potential-energy mass matrices (Ref. 5), or finite-difference mass matrices (Ref. 6) can be generated for each type of element

For structural problems in which gridpoint boundary conditions are not likely to be varied, the boundary conditions can be specified in the element data. This feature results in imposition of boundary conditions prior to obtaining the total structural stiffness matrix. This capability was easily incorporated in the program because of the particular type of coding technique used to identify each element of a matrix Further elaboration on this point is given in Section V

Equivalent gridpoint forces are computed internally in the program for elements in the SAMIS library subjected to temperature gradients normal to the neutral axes, temperature distributions, uniform accelerations, and/or pressure loads This system capability eliminates lengthy manual calculations to define gridpoint forces due to these types of loads However, individual equivalent gridpoint forces may also be input, if the analyst desires to augment or replace the structural loading

It is important to recognize that the types of structures that can be analyzed are restricted only by the types of elements contained in the program library Provision has been made for adding elements to this library without serious revamping of the subroutines involved. For example, the input and output data formats are sufficiently general to accommodate a wide variety of possible element data The elements and functions of the generation link of the SAMIS are summarized in Fig 5

Execution time of the BILD link has been assessed for certain problems solved on the JPL computer Resulting times as a function of the member of elements for which structural matrices were generated are plotted in

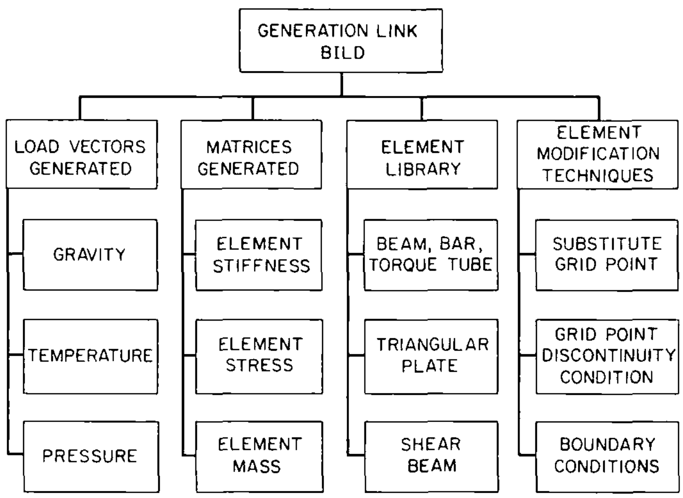


Fig. 5. Generation phase functions and elements

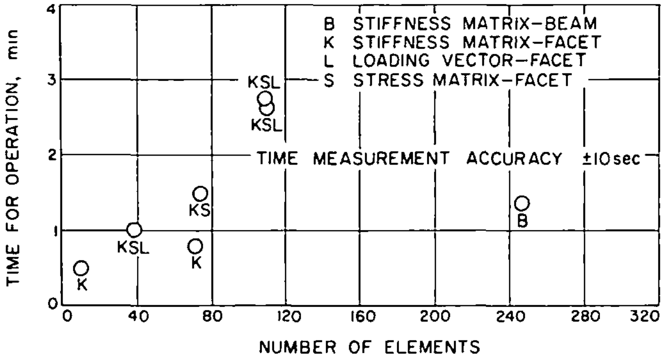


Fig. 6. Performance of BILD in element-matrix generation

Fig. 6 The time to generate the stiffness, stress and loading matrices for a single triangular plate element (FACET) is approximately 1 6 sec One data point for the beam element is shown in Fig. 6 (point B). Based only upon this point the time per element for generation of the beam stiffness matrix is 0 33 sec The general linearity of the various data points for the FACET element indicates the apparent accuracy of the data

IV. Functions and Capabilities of the Program Manipulation Phase

The second basic function of the SAMIS program is the algebraic manipulation of generated or input matrices to determine the unknowns of a problem The manipulative phase is currently made up of 15 links Five of the links perform standard matrix algebra, namely multiplication, addition and subtraction,, transposition, triangular decomposition, and row-column scaling. Three others perform functions on simultaneous algebraic equations and finding the roots and vectors of the matrix

The seven remaining links of the manipulative set are special-purpose programs for input and output of data, and for carrying out manipulations particular to the data format used in the SAMIS. The 15 manipulative links are listed in Fig 7 together with information on segment identification and data-size restrictions. The links having "serial option" blocks are set up to perform multiple operations in the function represented by the parent block. For example, serial multiplication is a link option that allows sequential multiplication of matrices without writing separate instructions for each step

The capabilities of the manipulative links of the SAMIS are probably of more general interest than those of the generation phase because the manipulative function is not restricted to structural problems Some indication of link manipulative capabilities is given in Fig. 7 by the labeling of the links as including in-core or larger-than-core operation capabilities However, additional information is given below for the basic manipulative links including certain restrictions and applications It should be noted that, as indicated in Fig 2, 20,000 words of storage are available in-core for general computational usage on a computer with a 32K core memory Hence, the order of the largest square matrix having all nonzero element values that can be placed in core is $\sqrt{20{,}000} = 141$ In structural problems, this size is rather small, however, two conditions temper this size restriction First, most structural matrices are sparse, and a matrix coding technique is used in SAMIS to take advantage of this condition and to increase the capability of in-core computation Additional comments on this coding technique are given in Section V. The information given below for each manipulative link assumes the matrices are coded unless otherwise stated

The data points shown in the performance plots in the remainder of Section IV were taken from output listings of actual computer runs. Reported are data in the only form that could be derived from the program printouts In interpreting the data, the most significant unknowns in most cases are the orders of the matrices What is reported is the number of blocks required to store each matrix Only if the average bandwidth of a matrix is known, can correlation between the number of blocks of data and the matrix order be established. One block of data contains 60 element values and 60 codes Thus 60 times the number of blocks is approximately the number of elements This number, divided by the average bandwidth, is then a representation of matrix order. However, this calculation can fail if a large number of zero elements lie between the diagonal and the last nonzero off-diagonal of each row, since, by the element coding technique employed, zero-valued elements are omitted.

The link running times also include the time required in MINTS to position tapes for link operation. For a given operation with identical matrices, the tape positioning time may be variable, depending upon the tape assignments the user selects for the pseudo instruction. In current program usage, a "worst possible case" in mismanagement of tapes would occur if a matrix were to be stored on a tape that already had on it a number of, say, element stiffness matrices. Optimum tape usage is attained when the output matrix is placed sequentially on one of the tapes from which input data was supplied In general it is good practice to spread matrix data onto several tapes rather than develop a long list on a single tape.

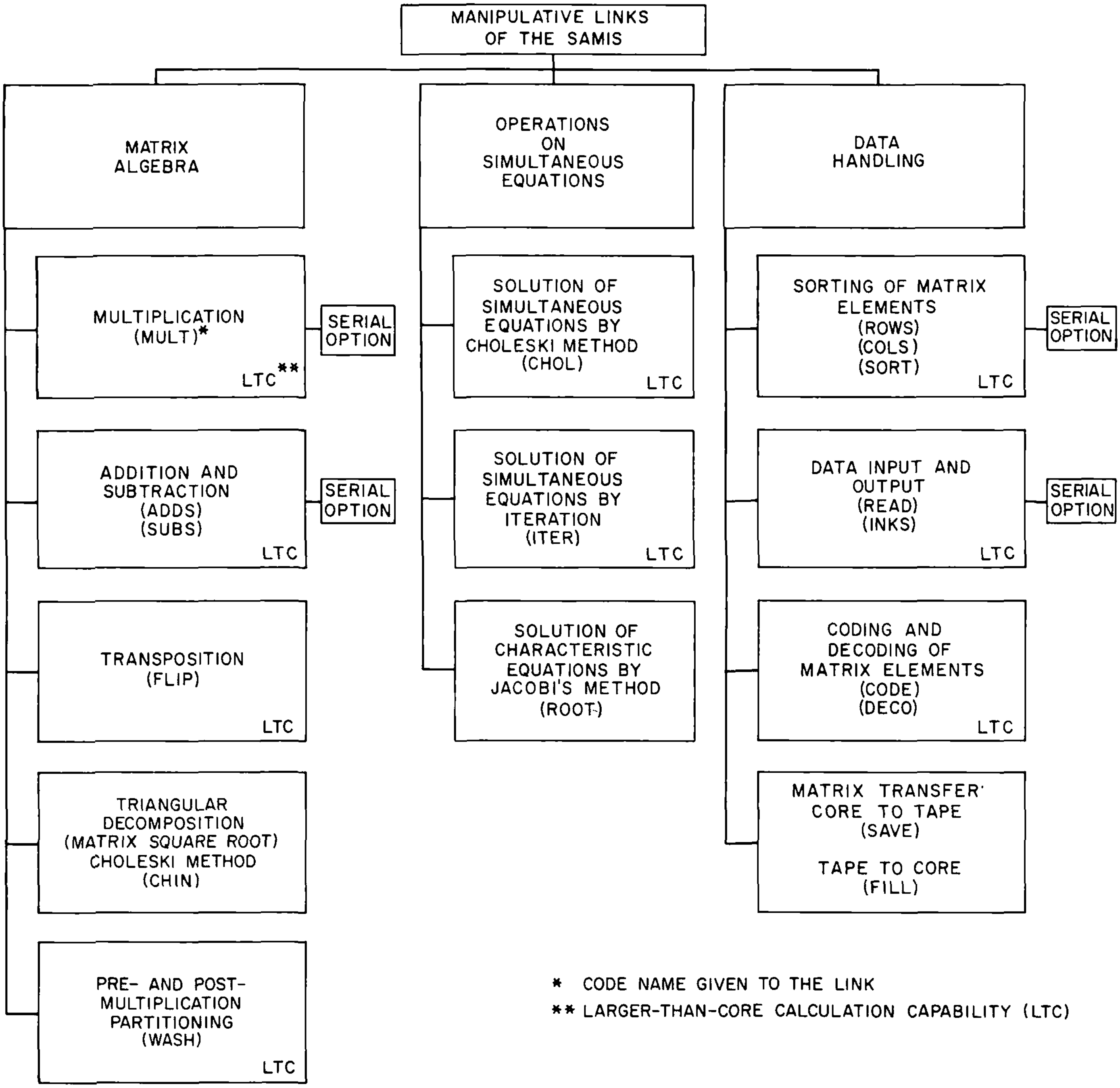


Fig. 7. Manipulative links of the SAMIS program

A. Multiplication Link (MULT)

This link multiplies two matrices together One of the two must fit in core. Because the multiplication is performed by code matching, neither matrix need be square or identically coded with the other If none of the row or column codes match, then the product is a null matrix. A simple example of the multiplication process is shown in Fig 8

The MULT link can also multiply together precoded matrices[1]; however, the matrices must both fit in core simultaneously. Another option of this link is serial multiplication, in which up to 999 matrices (arbitrary limit) may be multiplied together by a simple pseudo instruction

Of the seven most-used manipulative links of the SAMIS, the MULT link has the poorest performance characteristics. This is due to the necessity to re-sort the product matrix at completion of element multiplication If both the multiplicand and multiplier matrices fit in core, the speed of multiplication is faster than if only one fits in core Performance plots are given for the cases when both input matrices fit in core (Fig 9) and when only one matrix fits in core (Fig 10) The switch from in-core multiplication to larger-than-core multiplication occurs when the sum of the blocks of the input matrices (designated [A] and [B]) exceeds 166

The results presented in Figs 9 and 10 indicate that, when two matrices of vastly different sizes are multiplied

[1]Precoded matrices are of necessity rectangular arrays having zero-elements to complete the dimensional array Contrarily, coded matrices usually contain no zero elements either in a listing or when manipulated in the computer

COLUMN CODES

ROW CODES	11	12	15	34
11	*a*	*e*	0	0
12	*e*	*b*	0	0
15	0	0	*c*	0
17	0	0	0	*d*

X

	11	16	23	24
11	*f*	*g*	0	*k*
12	*g*	*h*	0	0
23	0	0	*i*	0
34	0	0	0	*j*

=

	11	16	23	24
11	*af+eg*	*ag+eh*	0	*ak*
12	*ef+bg*	*eg+bh*	0	*ek*
15	0	0	0	0
17	0	0	0	*dj*

Fig. 8. Illustrative multiplication operation

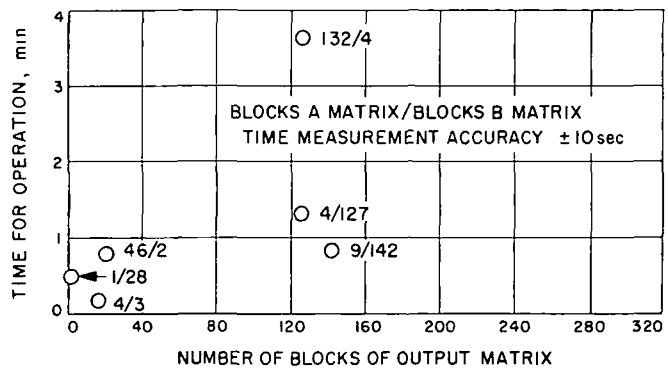


Fig. 9. Performance of MULT in in-core operation

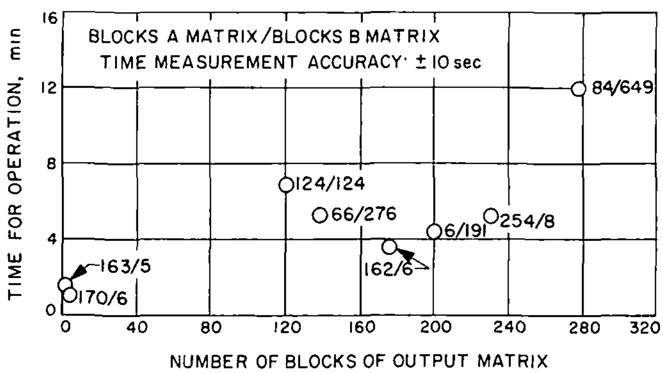


Fig. 10. Performance of MULT in larger-than-core operation

together, faster execution times are realized if the [B] matrix is the larger matrix. Based upon the single point evidence in Fig 10, if the two input matrices have approximately the same number of elements, the multiplication operation is slower than if the matrices are dissimilar in numbers of elements, with the total number of elements constant for the two cases

The data shown in Figs. 9 and 10 are for coded matrices.

B. Addition and Subtraction Link (ADDS, SUBS)

This link provides capability to add or subtract coded matrices in which neither matrix need fit in core The matrices need not be square nor must codes of the two matrices necessarily match Serial addition and subtraction are arbitrarily limited to 999 matrices as in multiplication

The element summing process used in this link is termed "wavefront summation." Basically, the process is: read into core as much of one matrix as possible (leaving some buffer), transfer to core a single block of the second matrix, combine elements by code matching. If core becomes full, one block of data with lower codes is moved to auxiliary tape as a new block is moved into core. It is assumed that the spacing between the code values of the elements leaving core is greater than any of the codes of the elements of the block entering core. Should this not be the case an internal recovery procedure is followed to complete the summation. In most problems, the spread of codes is less than the computer limit of approximately 150, and the summation process is performed without recovery.

Performance plots for both ADDS and serial ADDS are given in Figs. 11 and 12 It should be noted that these results apply also to subtraction of matrices. Based upon the results for serial ADDS, the time required for operation of this option is very sensitive to matrix size, as would be expected.

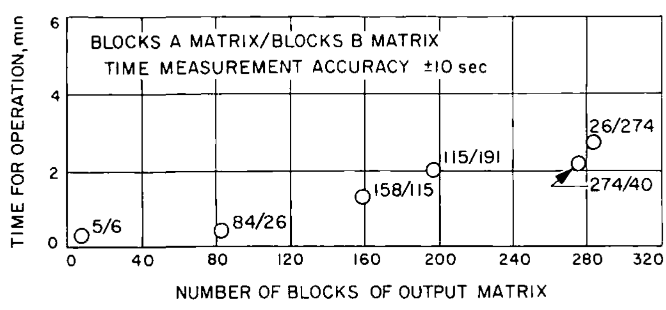


Fig. 11. Performance of ADDS or SUBS

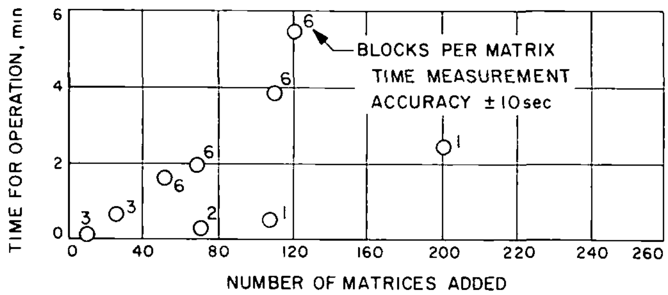


Fig. 12. Performance of serial ADDS or serial SUBS

C. Transposition and Listing Links (FLIP, ROWS, COLS)

Virtually any size matrix can be transposed by these links because the operation involves only interchanging the row and column codes of each element and relisting the array In a single pseudo instruction only one matrix can be transposed, however, a serial transposition option is not provided because the transposition is generally performed after summary or forming products of matrices.

Many data points were obtained to define the performance of FLIP. For this reason a solid boundary line is shown in Fig 13 rather than a number of data points. This boundary line is an upper limit on the time to transpose a matrix A large number of program runs had a transposition time well below this boundary, and no runs had times above this line.

As with FLIP, a large number of data points were obtained for ROWS and COLS. The performance boundary line for ROWS and COLS is shown in Fig. 13, since all three links involve code interpretation. The boundary line for ROWS and COLS is an upper limit also, with many data points below the line and none above. For all three links, the location of data points relative to the

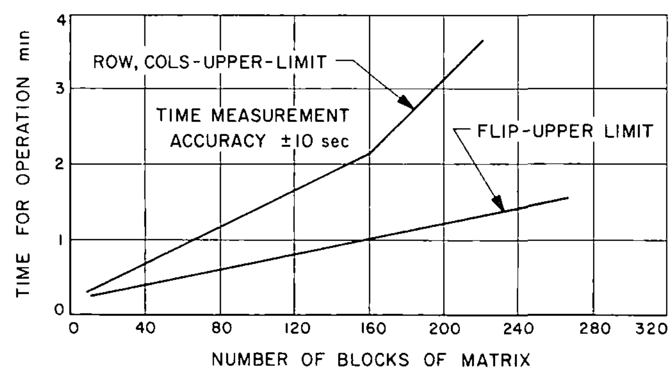


Fig. 13. Performance boundaries for FLIP, ROWS, and COLS

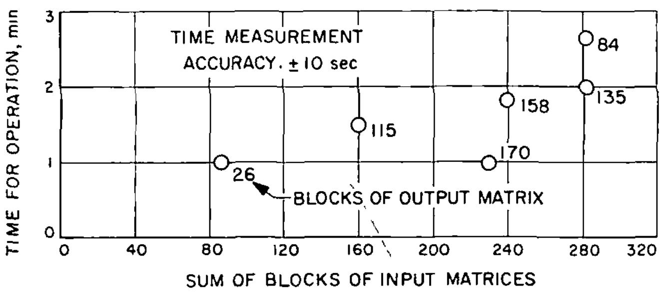
boundary line was apparently matrix size-dependent, however, no consistent trends were observed

D. Tri-Matrix Multiplication Link (WASH)

One function of this link is to partition (with attendant element scaling, if required) a matrix, say $[M]$, by row and column. This is accomplished by forming a matrix triple product $[T]'[M][T]$, where the matrix $[T]$ is diagonal and has either zero or nonzero (unit, if no scaling required) valued diagonal elements. The parent matrix $[M]$ need not fit in core, and, it need not be symmetric or square.

Additional functions have been incorporated in this subprogram to allow numerical scaling or extracting of elements of $[M]$ This operation is performed by element code matching between $[M]$ and a control matrix

Recorded times for WASH are shown in Fig. 14 for various sized matrices. No record was kept of the WASH link option used in the runs providing the data points. The results indicate that the time used by WASH is related roughly to the degree of reduction of the parent matrix.



E. Choleski Decomposition and Inversion Link (CHIN)

The purpose of this link is to define a matrix which, when multiplied by its transpose, equals the original matrix. The particular form of these matrices is: one matrix has nonzero diagonal and upper off-diagonal elements, and the second is the transpose of the first. That is,

$$[0\diagdown U]^T [0\diagdown U] = [M]$$

where $[0\diagdown U]$ is the upper triangular matrix, and $[0\diagdown U]^T$ is the lower triangular matrix. The advantage in finding the matrix $[0\diagdown U]$ in triangular form is that it can be inverted very simply compared to inverting matrix $[M]$ directly. Once the inverse of $[0\diagdown U]$ is found, the inverse of $[M]$ is determined by multiplication, that is,

$$[0\diagdown U]^{-1} [0\diagdown U]^{-1T} = [M]^{-1}$$

where $[0\diagdown U]^{-1}$ is the inverse of $[0\diagdown U]$.

This link is limited to symmetric, positive definite, square matrices. Because of the symmetry condition, only the diagonal and upper off-diagonal elements of $[M]$ must fit in the core of the computer in variable band form.[2] Thus, matrices whose complete array is larger than core can be manipulated; however, there is a definite upper limit on matrix size based upon the number of elements in the upper triangular and diagonal regions. For example, the matrix of largest order that can be manipulated by this link is 6,666, in which only the diagonal elements are nonzero. Any matrix having nonzero off-diagonal elements must be of lesser order than 6,666 to fit in core. This particular limit arises because the CHIN subprogram requires, within the 20,000-word data region, space for the input matrix plus one column for specification of bandwidth and one column for carrying out matrix decomposition. This means that, for a diagonal matrix, the 20,000-word region is divided into three equal sub-regions of 6,666 words each.

The CHIN segment is executed relatively rapidly. This is due partly to the limitations on matrix size and to the method of programming the algorithm. A limited, but truly representative, number of performance data points have been obtained from runs using CHIN. These data are shown in Fig. 15.

[2]Assuming symmetric matrices, the half-bandwidth of a row is merely the number of locations between the diagonal element and the last nonzero off-diagonal element In general, the row-bandwidth values should be made minimal to reduce calculations and provide for handling of larger-order matrices (Section V).

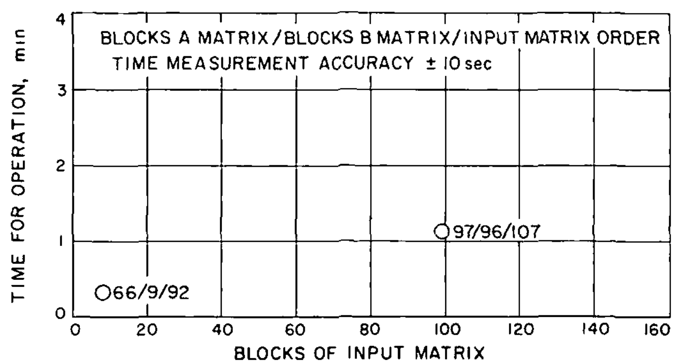


Fig. 15. Performance of the CHIN link

A typical application of the decomposition function is in solving the dynamic matrix equation The fundamental form of the equation is

$$\omega^2 [M] \{\delta\} = [K] \{\delta\}$$

where, in general, the matrices $[M]$ and $[K]$ are square and symmetric By means of the CHIN link we can decompose the mass matrix to obtain

$$\omega^2 [0\diagdown U]^T [0\diagdown U] \{\delta\} = [K] \{\delta\}$$

where

$$[0\diagdown U]^T [0\diagdown U] = [M]$$

We now define a new deflection vector $\{\delta'\}$ related to the original vector by

$$\{\delta'\} = [0\diagdown U] \{\delta\}$$

or

$$\{\delta\} = [0\diagdown U]^{-1} \{\delta'\}$$

Substituting for $\{\delta\}$, $\{\delta'\}$ modifies the dynamic matrix to

$$\omega^2 [0\diagdown U]^T \{\delta\} = [K] [0\diagdown U]^{-1} \{\delta'\}$$

Premultiplying this equation by $[0\diagdown U][K]^{-1}$, we obtain

$$\omega^2 [0\diagdown U] [K]^{-1} [0\diagdown U]^T \{\delta'\} = [I] \{\delta'\}$$

where $[I]$ is the unity matrix. The advantage in decomposing the mass matrix in the above example is that the matrix triple product $[0\diagdown U]^T [K]^{-1} [0\diagdown U]$ is a matrix that is symmetric, provided $[K]$ is symmetric. Insuring a final symmetric matrix allows subsequent use of efficient mathematical techniques for finding the roots and vectors in solving the problem The alternative to the above approach is to form the matrix product $[K]^{-1} [M]$ which is not symmetric.

F. Simultaneous Equation Solution Links (CHOL, ITER)

The CHOL link uses the Choleski triangular decomposition technique incorporated in CHIN to solve a set of simultaneous equations The basic operation is as follows. given a matrix equation with $\{\delta\}$ the unknowns

$$[K] \{\delta\} = \{P\}$$

the CHOL link calculates the solution as

$$\{\delta\} = [K]^{-1}\{P\}$$

where the soultion $[K]^{-1} \{P\}$ is found by the decomposition procedure

Limitations on the calculation are that the $[K]$ matrix must be square, positive definite, and symmetric. Because of the symmetry condition, only the diagonal and upper off-diagonal elements are read into core (in variable bandwidth form), consequently the total $[K]$ matrix can be larger than core.

The subprogram CHOL is a relatively fast link of the SAMIS. Performance data points obtained to date are shown in Fig 16, wherein the largest order set of simultaneous equations solved and reported is 476. The time required to solve this set was approximately 4 2 min.

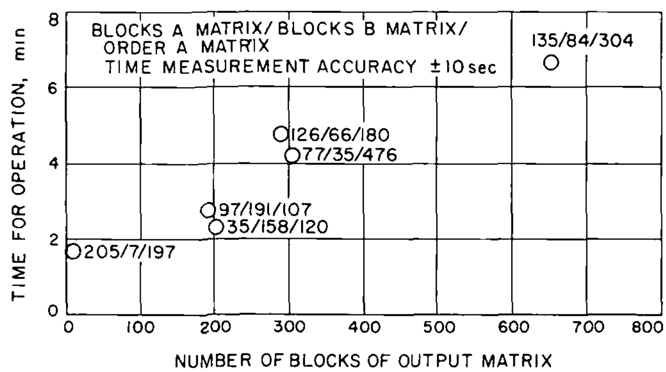


Fig. 16. Performance of CHOL link

The ITER link also solves the equation $[K] \{\delta\} = \{P\}$ but uses the accelerated Seidel Iteration Method. Limitations on this method are that $[K]$ must be square, positive definite, and have nonzero diagonal elements. However, the matrix $[K]$ need not fit in core nor be symmetric. Although the ITER subprogram is capable of solving a more general class of equations than CHOL, the computational time for ITER is generally significantly greater than that for CHOL. At the writing of this report no data points on the performance of ITER were available.

Basing the capacity of CHOL or ITER on the size of $[K]$ does not express the true limitation of the SAMIS system This can be demonstrated very easily. Assume that the storage required for the diagonal and upper off-diagonal elements of $[K]$ exceeds core A procedure called partitioning can be used effectively to allow use of the CHOL link in preference to ITER. The original matrix equation to be solved is

$$[K]\{\delta\} = [P]$$

Assume this equation is partitioned into two equal parts as follows:

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{Bmatrix} \delta_1 \\ \delta_2 \end{Bmatrix} = \begin{Bmatrix} P_1 \\ P_2 \end{Bmatrix}$$

This arrangement is actually an array of two matrix equations, namely:

$$[K_{11}]\{\delta_1\} + [K_{12}]\{\delta_2\} = \{P_1\}$$

$$[K_{21}]\{\delta_1\} + [K_{22}]\{\delta_2\} = \{P_2\}$$

Solving the second equation for $\{\delta_2\}$ yields

$$\{\delta_2\} = [K_{22}]^{-1}\{P_2\} - [K_{22}]^{-1}[K_{21}]\{\delta_1\}$$

Substituting into the first equation gives

$$[[K_{11}] - [K_{12}][K_{22}]^{-1}[K_{21}]]\{\delta_1\} = \{P_1\} - [K_{12}][K_{22}]^{-1}\{P_2\}$$

or

$$[\overline{K}]\{\delta_1\} = \{\overline{P}\}$$

In this equation $[\overline{K}]$ is of the same order as $[K_{11}]$ and $[K_{22}]$ which are one-half the order of the original matrix $[K]$. Thus, CHOL can be used to determine $[K_{22}]^{-1}$, then to solve the final equation for $\{\delta_1\}$ by inverting $[\overline{K}]$.

It is apparent that by this method of partitioning, matrix equations that exceed core storage capacity can be solved; however, several matrix manipulations must be performed in place of one.

G. Eigenvector and Eigenvalue Link (ROOT)

The function of this link is to determine the characteristic roots (λ) and associated eigenvectors $\{\delta\}$ from input of the matrix $[R]$, where $[R]$ is defined by

$$[[R] - \lambda^2[I]]\{\delta\} = \{0\}$$

Referring back to the derivation of the dynamic equation outlined in the CHIN description, it may be concluded that

$$[R] = [0\diagdown U][K]^{-1}[0\diagdown U]^T$$

and

$$\lambda^2 = \frac{1}{\omega^2}$$

The algorithm used in this link is Jacobi's Method in which requirements on the input matrix $[R]$ are that it be square, symmetric, and of order 130 or less.

Since the ROOT link requires the input matrix to be precoded, that is, a solid square array of numbers, and since the characteristics of the algorithm for finding the roots are vectors is well defined, the performance of ROOT is functionally predictable Basically the time required to determine the roots and vectors of a matrix of order N is proportioned to N^3. The proximity to which this functional relation applies is clearly indicated by the performance data presented in Fig. 17. As shown in Fig 17 the largest matrix manipulated in ROOT was of order 128, and the time used was approximately 24 min

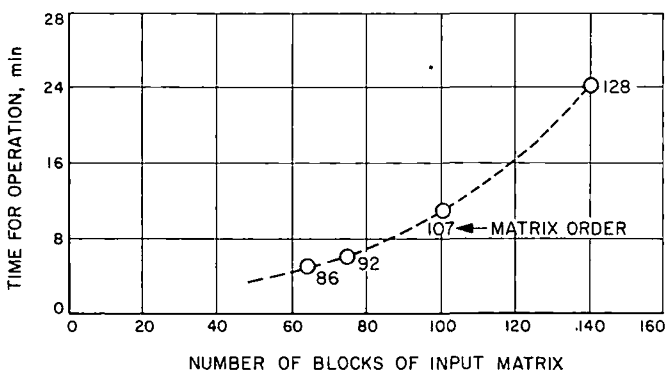


Fig. 17. Performance of ROOT link

H. Overall Program Performance in Solving Structural Problems

Two test problems of significant size were solved in checkout of the SAMIS program. One problem was to determine the mode shapes and frequencies of an unconstrained, shallow, spherical shell The second problem was the determination of the deflections and stresses of the same shallow shell, with its outer edge clamped and the shell being subjected to a uniform pressure loading and a temperature induced loading. Complete results of these problems are reported in Ref. 3.

In the setup of the problem to evaluate the low-frequency flexural modes of the unconstrained shallow shell, only one-quarter of the shell was analyzed because of the symmetry of the mode shapes This sector was idealized by 54 triangular elements (following the surface contour of the shell) having 38 discrete gridpoints (see Fig. 18), resulting initially in a stiffness matrix of order 324. The time required to generate the stiffness and mass matrices and superimpose them to obtain over-all stiffness and mass matrices was approximately 3 5 min. Since the shell was not sufficiently constrained, it could move as a rigid body, and to remove this singularity from the matrices necessitated a transformation The transformation required eight pseudo instructions and used 6.67 min to complete. Had the shell been restrained sufficiently so that rigid body motion was not possible, as is the case in most problems, then these eight operations would not have been needed Formulation of the dynamic matrix by decomposition of the mass matrix and inversion of the stiffness matrix involved five pseudo instrutions and 12 min of computer time. Of this time 5 min

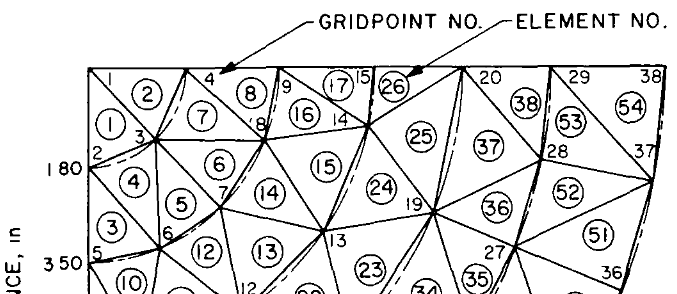


Fig. 18. Triangular grid array for quarter shallow shell

were used to form the product $[K]^{-1}[0\diagdown U]^T$ as defined in Section IV-E, above. In forming this product, the stiffness matrix was a solid square array (caused by transformation to eliminate rigid body mode) of order 185, and the matrix $[0\diagdown U]^T$ was of order 86 and a solid upper triangular array Next, the solution for the eigenvalues and eigenvectors of the final dynamic matrix, which was of order 86, required 5 5 min. Finally, the eigenvectors were transformed back to the original displacement set by four pseudo instructions requiring 3 67 min. The total time for the computer run was approximately 33 min If the boundary conditions of the problem could have been treated in BILD (as is usually the case), the running time to solve the problem would have been on the order of 22 min

The second problem of determining the deflections and stresses in the shell, loaded as shown in Fig 19, has symmetry about the axis of revolution of the shell (uniform shell) as a condition. The intention in formulating this problem was to require manipulation of matrices larger than core but to avoid the exercise of partitioning to invert the final stiffness matrix Therefore a 20 deg sector of the shell was selected and idealized by the triangular array shown in Fig 20 This array has 70 gridpoints, so the initial matrix order is 420. Generation of the 108 stiffness matrices and 108 corresponding pressure and temperature loading vectors required 3 0 min Superposition of these to form the complete structural stiffness matrix and loading vectors took 4 2 min to complete. Because the problem was referenced to Cartesian coordinates, the boundary conditions along one of the meridinal edges ($\theta = 20$ deg) were not of the type $u_j = 0$ or $u_j \neq 0$ (generalized gridpoint displacement), rather, there was a relation between the displacement components Imposition of these boundary conditions required nine pseudo instructions and 14 min

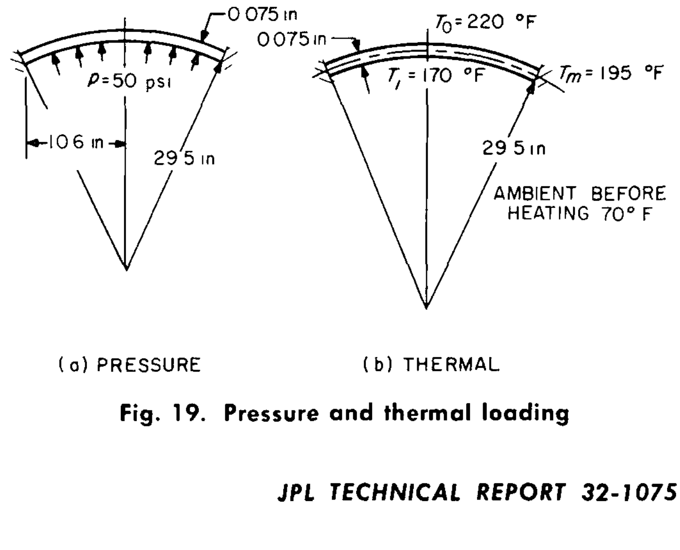


Fig. 19. Pressure and thermal loading

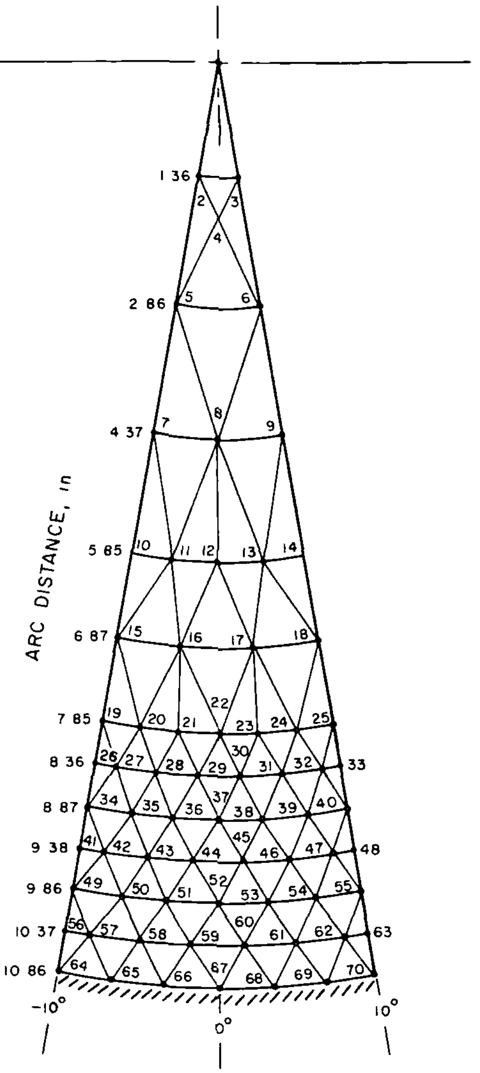


Fig. 20. Triangular array of 20-deg shell sector

of computer running time. The result of this set of operations was to reduce the order of the stiffness matrix from 420 to 307. The final matrix used 170 data blocks to store which is just 4 blocks larger than core capacity. Since CHOL operates only with the diagonal and upper off-diagonal triangular part of the matrix, CHOL could be used to solve the set of simultaneous equations without partitioning. This operation was completed in 1 2 min. Note that there were only two columns (pressure and temperature loading vectors) in the matrix P (Section IV-F, above). The shell reaction forces were computed in 10 sec and the 108 individual member stress resultants in 40 sec. Complete running time for the problem was 26 min. Had it been possible to impose the boundary conditions in BILD, the running time would have been approximately 12 min

The links described in this section are the basic manipulative functions of the SAMIS program. All are limited to calculations with linear equations, and all involve only real algebra The algorithms used in each subprogram are considered efficient for applications on a digital computer, and each is documented completely. It is worth noting that, although these links have certain limitations and restrictions, by judicious use of symmetry conditions, partitioning techniques, structural idealization tradeoffs, etc., these constraints can be circumvented in many problems to obtain accurate solutions.

V. Review of Objectives in Program Development

A. Techniques of Achieving a Balanced Program

The word balanced is used here to imply development of a computer program that possesses an equal level of performance in areas of. complexity of the problems to be solved, time expended in computations and data handling, and accuracy achieved in the results. In a contractual sense, expressing this condition by explicit constraints is difficult to do except by devising guidelines to prevent excessive effort in one area with a consequent disregard for other major problem areas. It is felt that this balance of operations, so difficult to assess except by observation of the final product, has been achieved in the SAMIS program and is a definite credit to Philco Corporation WDL, the developers of the program.

In initial definition of program development objectives, to decide upon the level of problem complexity was not difficult because of several natural limits. First, the triangular element, with six variables per gridpoint (as compared to two for a planar rod element), entails use of a large number of variables for any significant problem Therefore, it was established that the program would be designed to solve, efficiently, structural problems in which the stiffness matrices varied from 100th to 2,500th order For matrices smaller than 100th order, the calculations are performed easily in-core, and many structural analysis systems are available. For matrices larger than 2,500th order, the capacity of present computer systems is probably exceeded, in that the time required for problem solution becomes excessive, and accumulating errors are likely to destroy all accuracy Hence, the SAMIS program is designed to handle matrix problems that are intermediate in size.

With these constraints on size, core capacity may be exceeded in a single calculation, hence link logic must be formulated for in-core as well as larger-than-core calculations When core is filled, the only way to account for remaining data is to use auxiliary storage It is well known that, when auxiliary magnetic tape storage is coupled with in-core calculations, the computational time increases significantly because of tape search and read times However, when the development began, there was no alternative to this storage option and the time for operations were minimized within this constraint.

In the SAMIS program, time minimizing is achieved by performing tape search and rewinding operations on as much of a non-interference basis as possible, consistent with computer operation Every advantage has been taken of existing computer capabilities to minimize tape sequencing, however, this still remains a major contribution to total computational time Within the limitations of the existing JPL computer system, the tape problem could be alleviated by writing an independent computer monitor system, specifically adapted to the SAMIS program However, this approach is not compatible with job-sequencing practices at most computing centers.

Fortunately the developers of computer hardware recognize that this tape-core flow problem is a major limitation on computer performance Consequently, they have introduced several new components and a new monitor system to increase storage capacity and decrease data access time Disk filing reduces data access time and increases storage capacity, and the FORTRAN IV monitor system allows for tape reading and writing while calculations are performed in-core (buffering) These improvements are not yet used in the SAMIS program but when they are incorporated, operation and computation times will be reduced significantly. In planning for the eventual conversion of the SAMIS to new monitor systems, the developers have used the FORTRAN language (rather than machine language) in approximately 98% of the SAMIS program This strong emphasis on FORTRAN is justified not only by the increased simplicity of adapting the program to different monitor systems, but also by the ease of interpretation of subprogram functions by a user not completely familiar with the program

In addition to flow problems from link to link, and between core and tape, consideration was given to minimization of the quantity of data required. Stiffness matrices, coordinate transformation matrices, and mass matrices are by nature sparse matrices, having considerably more zero-valued off-diagonal elements than nonzero values. Therefore, it is efficient both in core utilization and in input data preparation to ignore the zero-valued elements. Ignoring zero-valued elements cannot be easily implemented when using index incrementing techniques which operate with complete matrix arrays. However, it can be accomplished by assigning to each nonzero element a separate code that identifies the element by row and column. By this technique, two words are needed for each matrix element (code and value). Since, however, the matrices are sparse, the result is a net reduction in core storage used per matrix (Fig. 21) Matrix algebra is basically carried out by "code matching," so that overall dimensional compatibility of matrices is not required. For example, in matrix addition, elements of each matrix having matching codes are added, while the remaining elements with nonmatching codes are relisted in the summed matrix (Fig. 22). Although this coding technique may seem to require considerable bookkeeping effort, the codes can be made up of identifying numbers that correspond to deflections or forces at gridpoints of the structure, so that code interpretation is apparent In matrix problems not associated with structures, a rational coding system must be defined for easy identification, this, however, is not a difficult task

The quantity of input data can be reduced by defining several optional coordinate systems that the user may select. For example, in a plate problem, coordinate axes with two axes in the plane of the plate reduce the

COLUMN CODES

ROW CODES	11	12	13
11	a	b	0
12	b	c	0
13	0	0	d

(a) MATRIX ARRAY AND CODES

11	11	a	11	12	b	12	11	b
12	12	c	13	13	d			

(b) CODE LISTING OF MATRIX

Fig. 21. Illustration of matrix coding technique

$$\begin{array}{c} \\ 11 \\ 12 \end{array}\begin{array}{c} \begin{array}{cc} 11 & 12 \end{array} \\ \begin{bmatrix} a & b \\ c & d \end{bmatrix} \end{array} + \begin{array}{c} \\ 12 \\ 15 \end{array}\begin{array}{c} \begin{array}{cc} 12 & 15 \end{array} \\ \begin{bmatrix} e & 0 \\ 0 & f \end{bmatrix} \end{array} = \begin{array}{c} \\ 11 \\ 12 \\ 15 \end{array}\begin{array}{c} \begin{array}{ccc} 11 & 12 & 15 \end{array} \\ \begin{bmatrix} a & b & 0 \\ c & d+e & 0 \\ 0 & 0 & f \end{bmatrix} \end{array}$$

Fig. 22. Matrix addition using coded elements

geometry input data by one-third, because only two coordinate values, instead of three, are needed to define the location of a gridpoint This advantage does not exist, however, when analyzing doubly-curved shells

Computer operation time may be reduced also by minimizing the number of times a given amount of data is handled In attempting to reduce data handling time, it became evident that mathematical, rather than structural, partitioning was more efficient Structural partitioning involves grouping data by row and column, each group representing a segment of the structure In structural partitioning, the operations are reflected in the pseudo instruction program, which becomes very large When this happens tape-search time becomes a significant factor in reducing program efficiency

Mathematical partitioning is accomplished by data handling techniques incorporated in each link This technique evolved from the condition that the links manipulate matrices that either fit in-core or are larger-than-core This technique involves partitioning of groups of data by row only and is considered mathematical because no structural interpretation can be given Since this operation is internal to the program, the user is unaware of the partitioning and is not burdened with data identification problems.

In considering program accuracy, it must be recognized that insuring accuracy over a wide spectrum of problem types and sizes is a difficult task This fact was recognized at the onset of development of the SAMIS program, and considerable effort went into determining the possible errors that could occur and rational means of reducing them. A complete description of the work done on error recognition and reduction can be found in Ref 1, hence, only a brief summary is included in this report Five categories of errors are defined in Ref. 1, and each type is described briefly in following paragraphs

One major source of error is human error in preparing input data, or *input error*. In preparing data for complex problems, there is considerable input of a repetitious nature, which is a situation conducive to error Fortunately, this type of error can be virtually eliminated by use of an input data diagnostic link. The function of this link is to check element data format, input matrix size and format, pseudo instruction format and tape assignments, and overall compatibility between these packages. A link to perform this function is being developed for the SAMIS, and is designated the CHEX link A description of this program is given in Section VII

A quite different effort was devoted to reduction of error in structural representation The stiffness method, when applied to analysis of shell-type structures, is an approximate method, in that the triangular element representing the local structure is not an exact representation. In the case of the flat-plate-triangular-element used in the SAMIS program, establishing continuity of deformations between elements requires the use of linear displacement functions that do not allow exact representation of the bending mechanism Also, internal forces in the structure are determined (lumped) only at the gridpoints of each triangle, the exact distribution of forces within the element is not known, and an averaging technique must be used to define stresses Other factors that influence this approximation are force equilibrium, stress continuity, and field compatibility. This kind of error, namely, that the triangular element is not exact in representing local structure, is termed *discretization error*.

The method of reducing discretization error in the SAMIS program centered on deriving an adequate triangular element representation. The scope of this task was not fully recognized at the start of development, and three distinct element representations were generated before an acceptable model was determined A further complication was that any element representation could not be evaluated until the generation package and most of the data manipulative links were functional and checked out The triangular element representation currently used in the SAMIS program is derived in Ref. 1 as well as in Ref 7

The only way of reducing discretization error is to derive a better element representation than the one used Thus, reduction of this error is contingent upon theoretical developments in the field of the finite element method. In anticipation of likely theoretical developments in finite element representations, e.g , Ref 8, the format of the generation link has been set up so that element changes and modifications can readily be incorporated.

A third category of error, termed *idealization error*, is related to the manner in which the actual structure is idealized by the finite element array In general, to minimize this error, the finite element breakdown should be coarser in regions where variables (deflections, curvatures, forces) change slowly and finer in regions where variables change rapidly Orientation of triangles at boundaries with respect to material and stress axes and in regions of concentrated loads is another factor for consideration in defining the triangular array. It is apparent that much of the control of this error is a function of the initial problem setup, which is dependent upon the knowledge and experience of the user. A potential user might argue that as the grid size is reduced the representation approaches an exact one, and accurate answers are assured (Fig. 23) However, there are two conditional factors in this argument. First, if the grid size is reduced, the amount of input data increases, and the size of matrices that must be manipulated increase possibly to such an extent that the calculations exceed current computer capability. Second, when the matrices become large, the number of manipulations increases, and an error develops that is inherent in any repetitious calculation, namely, round-off error This error is called the *manipulative error* and is the fourth type reported in Ref. 1.

The manipulative error is basically a function of the conditioning of the matrices being manipulated, the mathematical procedures used in each link, and the accuracy limits set on the computer. In the SAMIS program, due to operating time constraints, the accuracy limits have been established at eight places—what is termed "single precision" accuracy of the computer The mathematical procedures used in the various links of the SAMIS are based upon mathematical algorithms that are known to be efficient in computer applications. With regard to matrix conditioning in structures problems, conditioning is normally good because of high attenuation in structural coupling. This effect results in matrices that are "near diagonal," that is, only the off-diagonal elements near the main diagonal are nonzero, and are numerically less than their respective diagonal element values. Since the question of matrix conditioning depends upon the particular characteristics of individual matrices, it is difficult to establish methods of correction when ill-conditioning is encountered. In many cases, ill-conditioning is due to the element breakdown selected to represent the structure. A corrective measure is to reidealize the structure rather than consider changing any of the manipulative operations.

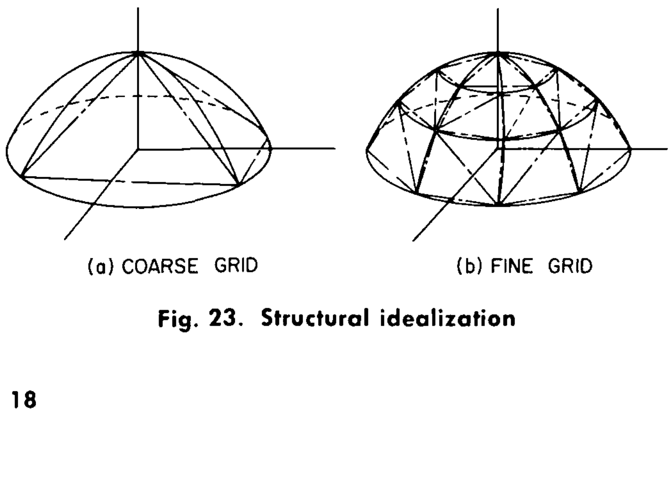


Fig. 23. Structural idealization

The fifth error classification is called *interpretation error* and is associated with the problem of interpreting the output or results of computations Visualize, for a moment, preparing a large amount of input data for an extremely complicated problem, feeding this data into the computer, and, half an hour later, obtaining some answers The question is—how good are these answers? If approximate or estimated answers are not known from another source, then this question is not easy to answer. However, there are at least four controls that may be applied to interpretation of the results First, the past experience of the program user may be helpful in establishing an "intuitive feel" for the range of answers expected. Second, equilibrium or orthogonality checks may be run to establish the validity of the results This can be done readily by simply altering the pseudo instruction program. Third, the problem can be rerun with a finer or coarser element (triangular) grid to check convergence of the solution, with the possibility of extrapolating the limiting answer if several solutions are obtained Fourth, if an energy approach is used to define the mathematical model of the finite element, then in some cases potential and complementary energy approaches can be taken, in which the answers obtained by the two approaches bracket the correct answer for the idealized structure. This procedure is called "the bounding technique" in finite element applications, and its use is predicated upon a rational energy approach being used to derive the mathematical model of a structural element. The bounding technique has been applied successfully to simple problems (Ref. 9), but remains a basic developmental task for complex elements such as the triangular element. At present, in the SAMIS program, the first, second, and third control measures can be used to interpret results. The fourth control measure, the bounding technique, is being considered for a followon effort after more information and experience is gained in using the program in the earlier phases of work

B. Methods of Achieving Program Versatility

Reference has already been made to the modular or chain approach being coupled with the pseudo instruction program to form the basis for a flexible computer program The ability to solve a wide spectrum of problems is reflected in this form of flexibility One extreme in program applicability is that a "black box" approach can be taken if the user is repeatedly solving a particular type of problem. The setup for a program of this type involves first establishing an efficient sequence of pseudo instructions Some error instructions, as well as planned recovery points, need to be included to allow for size variations in input data and recovery from computer terminating errors Once the program is set up and checked out, the user is merely instructed on input-output writeup and interpretation and on recovery procedures. With this approach, the user need not know matrix algebra even though matrices may be required input. In the SAMIS program, because the matrix coding system is related to the structural idealization, the input matrices have the appearance of tabulated data, and interpretation is possible without explicit use of the word "matrix." The output data is also printed in tabular form, and an understanding of the meaning of the codes that accompany each element value is sufficient for interpretation.

The other extreme in program usage is in the solution of the general matrix problem. Applicability of the SAMIS program to solve general problems is contingent upon the functions performed by the manipulative links. Hence, the user must be familiar with matrix algebra and the options, capabilities, and limitations of the SAMIS links. By means of this approach, problems in any discipline may be solved using the SAMIS program, provided the solutions can be effected using matrix algebra

In addition to versatility in general matrix-manipulative applications, the user will find the SAMIS program versatile in structural applications within the limitation of the elements in the program. Some of the options available to the structural analyst are the following:

(1) Selection of elastic axes when working with non-isotropic materials.

(2) Use of any of several mass matrix representations for each element.

(3) Ability to impose boundary conditions either in the program generation link, BILD, or by pre- and post-matrix multiplication.

(4) Specification of local coordinates for elements along which stress resultants are determined.

(5) Idealization of shell stiffness by either co-planar or "off-set" line elements.

(6) Selection of coordinates along which deflections are referenced

(7) Use of program-generated loading vectors for pressure and temperature induced loads.

(8) Complete freedom to "mix" element types in idealization of composite structures.

(9) Ability to interpolate and/or extrapolate material physical parameters as a function of material temperature.

(10) Ability to represent displacement and force discontinuities at structural joints.

The ease of incorporating extensions and modifications into the SAMIS program is also an important form of system flexibility There are two areas where this form of flexibility is needed First, the modular format of the program library allows for addition of new links by merely identifying the program in the master intelligence system for pseudo instruction interpretation and assigning a chain number to the subprogram for locating it on the library tape The other area where expansion is likely is in the generation subprogram library of stiffness elements Expansion problems in this area center on changes in the input data However, the input data format of the SAMIS program has been planned and set up so that any type of element, including the three-dimensional solid, can be put in the library without format changes.

C. Methods of Insuring Program Reliability

The type of reliability of concern here is that associated with the functioning of the entire program. Since a computer program is merely a set of instructions that the computer is slaved to follow, once all of the logic and flow options in the instructions have been tested and proven to be correct, reliability is assured in the sense that a submitted problem will be properly executed. The only variants are the validity of the input data and the malfunctioning of the computer itself.

For large programs such as the SAMIS, containing numerous flow patterns and options, checkout is an extensive effort In the checkout of the SAMIS program three levels of problems were generated and run. The function of the first level of problems was to establish the rationale of the logic within each link. Next, with the links grouped in a chain library, the second level of problems checked the compatibility between links (in various sequences). Finally, because the program was designed to manipulate large blocks of data that could fill and exceed core, checkout of program capacity dictated the third level of test problems

It was recognized that one comprehensive test problem could not be formulated that would check all options of the SAMIS. Therefore, a support effort was planned and subsequently integrated with overall program development. This activity by JPL personnel was intended to aid

WDL in certain critical areas and orient JPL personnel on details of the program functions and operations In addition to the checkout effort already described, this support function involved the following participation by JPL technical representatives· weekly technical meetings with WDL personnel to discuss any matters relating to program development, review of technical and programming documents as they were generated during development, and participation in writeup of some of the key segments of the SAMIS program. During this checkout phase, JPL assigned from one to three engineers and one programmer as needed to perform the various functions In retrospect, it is felt that this support effort not only effectively oriented the JPL technical personnel in understanding the SAMIS program operation but served also as a check and balance to increase the reliability and to speed up the checkout of the program [5]

Checkout of system manipulative links proceeded with the normal types of errors causing difficulty Some program features were very useful in checkout, particularly the recovery feature Cross checking of different link outputs was advantageously used. For example, the output from the links used to solve simultaneous equations (CHOL, ITER) was compared to the matrix decomposition link (CHIN) by proper selection of the set of equations. Vector orthogonality and matrix symmetry checks, core dump and data tape analyses and other standard techniques were used to check the logic and flow of the program

The other phase of the program, the generation package, was the source of two fundamental problems that impeded system checkout derivation of an adequate triangular element representation and the rational distribution of gridpoint forces over each triangle to obtain accurate stress values The problems associated with the triangular element representation have already been discussed briefly in this section. Although displacement values can be determined very accurately (Ref 3), the determination of accurate stresses is a well-recognized problem in finite element methods. and for many element types no exact procedure is available. Methods for computing stress resultants for the triangular element were investigated at JPL, and the results of this work are reported in Ref. 10 The method for computing stress resultants in the SAMIS program are outlined in Ref. 1.

[5]Several activities resulted from JPL participation to complement the primary work on program development For example, JPL documentation of the test problem input data and solution supplements the technical and usage reports prepared by WDL During program checkout, additional small test problems were submitted by JPL personnel to verify the operation and flow of certain program options not tested by the comprehensive test problems In having the opportunity to review the technical aspects of the methods used in the SAMIS program, JPL personnel have been able to initiate supporting development and research projects and to better plan the direction that should be taken to extend and improve the SAMIS program to meet future analysis needs

VI. The Engineering Function in Structural Analyses with the SAMIS Program

As is true in any computer program, a certain amount of basic data must be supplied to the system to initiate the generation and solution routines Understandably, since the elements used in shell-structure idealizations are more complicated than, say, the beam elements used in frame-structure analyses, the amount of input data for the SAMIS program exceeds that of corresponding frame analysis programs. In fact, the beam, bar, and torque-tube elements, which are types of elements needed in the analysis of frame structures, are a part of the element library in the SAMIS program, and so constitute a fractional part of the total input specification

In general, the procedures involved in establishing input for frame-type structures are well known, so emphasis will be placed here on the problems of input preparation for shell-type structures. In setting up a shell-type problem for the computer, one of the first tasks of the analyst is to establish a triangular array that adequately represents the characteristics of the structure. This must be done within the constraints on matrix size and, consequently, may involve the use of: structural planes of symmetry, local refinements in triangular grid sizes, modal convergence techniques, mathematical partitioning, and other methods to avoid bulky manipulative operations. Once the triangular grid is established, the next step is to number the gridpoints. This step is extremely important, since the matrix row-bandwidths are established by the nature of the numbering sequence In general the optimal arrangement is that in which adjacent gridpoints have numbers that are close in value. Procedures for testing the quality of particular gridpoint numbering arrangements are given in Ref 1. These procedures have been defined thoroughly in order that this calculation can be performed by engineering-aide personnel. After the triangular grid and gridpoint numbering sequence have been defined, the element input data can be prepared, also by engineering-aide personnel

The tasks remaining for the engineer are to select the appropriate set of pseudo instructions and to write the variable transformation matrices (in tabular form). The variable transformation matrices that were used to solve the program test problems are given in Ref 3 For most problems, this work can be completed in two or three days by an engineer and engineering-aide familiar with the data formats.

The SAMIS program has been developed purposely to possess the generality needed to incorporate many diverse structural conditions, however, with this advantage comes the disadvantage of a more complex input data format. That is, versatility in computer applications implies use of additional input specifications to accommodate the various program options. Consequently, the analyst may be required to supply more input data using the SAMIS program than what he may consider a minimal amount. A good example of this is that a minimal input of elastic constants might be Young's Modulus and Poisson's Ratio when the material is isotropic However, the stress-strain law used in the SAMIS program is general enough to represent a monotropic material (13 elastic constant) so that the 13 constants must be computed for a monotropic material as well as for simpler material representations However, for any given material this calculation need be performed only once and retained in a material table for use in all subsequent problems. Clearly, the analyst also has the option of developing supplemental computer programs to optimize the generation of input data for the SAMIS program

Finally, it is well to mention that the work involved in preparing input data for a computer program such as the SAMIS should be weighed against any alternatives of finding solutions with the same accuracy by other methods For structural problems involving laminated or stiffened structures, cutouts, concentrated or asymmetric loads, local support conditions, and other non-obliging conditions for closed-form or numerical integration solutions, the SAMIS program (or any comparable program) is the only means of obtaining approximate answers Therefore, one additional task of the analyst is to decide if the problem to be solved is appropriate for the SAMIS program or should be solved using other methods, other computational tools, or a combination of techniques.

Convergence on minimal analyst effort is the goal of most program developers, and it is anticipated that, as the SAMIS program is updated by conditions found through usage, changes will be made that will refine and condense the input-output format However, within the framework of a given computer system, input-output format can be condensed only so far, since a certain amount of data must invariably be supplied to the computer, which is a computational tool and not as a knowledgeable entity.

VII. SAMIS Links under Development

The description given of the SAMIS program links in Sections I through VI is of a version of the program that is currently available as a production system. Development of additional link functions which will extend the applicability of the program in structural analyses is being continued by WDL and JPL technical personnel. Below is a list and brief description of the link functions which are under development and should reach production status in the near future

A. Second-Order Differential Equation Links (LOCI, DEQS)

Consider a second-order matrix differential equation of the form:

$$[A]\{\ddot{X}\} + [B]\{\dot{X}\} + [C]\{X\} = \{P\}$$

The segments LOCI and DEQS operate with this form of equation in two distinct ways The function of the link LOCI is to determine the change in value of the roots of the homogeneous equation ($\{P\} = \{0\}$) under some parameter variation Any parameter in any two of the three matrices $[A]$, $[B]$, or $[C]$ can be varied arbitrarily. The information found by the LOCI link can be interpreted as the transfer function of the system described by the second-order equation

The function of the DEQS link is to solve digitally the second-order equation when the forcing function is nonzero Several functional forms can be selected for the forcing function, including the step, ramp, or sinusoidal types. In addition, an arbitrary forcing function can be input by representing the time history of the force by a sequence of impulses (Fig. 24) The forcing function can

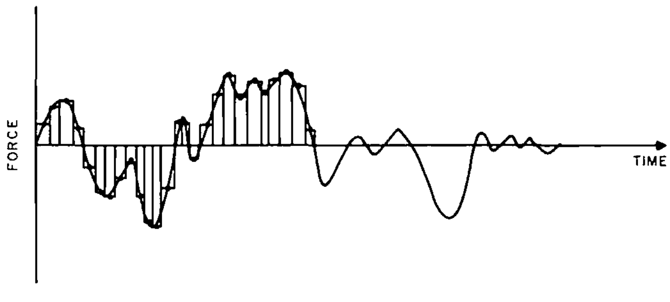


Fig. 24. Discretization of nonharmonic force

be an actual force, a specification of defined displacements, or accelerations acting on the system. Both the LOCI and DEQS links are limited to matrices of 40th order or less The Muller method is used to isolate the roots in the LOCI link and the Runge-Kutta method is used to perform the integration in the DEQS link

Increasing analysis capability beyond the 40-degree-of-freedom limit for forced-response prediction is under development, but will not be completed in the next year. This extension incorporates the method of component mode synthesis to analyze complex structures (Ref. 11 and 12).

B. Extended Choleski Decomposition Link (CHOL, Modified)

The same basic matrix manipulations performed by the current CHOL will be performed also by extended CHOL. The difference in capability between the two is that the extended version will solve simultaneous equations whose diagonal and upper off-diagonal coefficient matrix can exceed core size. Thus, using extended CHOL, a set of simultaneous equations of any order can be solved, the only limitation being that the row-halfband width must be less than 200 elements. For structural applications this limitation is not severe.

The basic difference between the two links is that, while both versions require two passes through the coefficient matrix to form the triangular decomposition, the current CHOL link requires that this decomposition fit in core while the extended CHOL link stores the triangular decomposition on tape The extended CHOL link must then pass piecewise through the decomposition twice for each solution to the set of simultaneous equations, once to perform a forward substitution for an intermediate solution and once for a back substitution to obtain the final solution.

C. Buckling Load Prediction Links (BILD, SMAD)

These special links support the capability to predict the small-deflection buckling loads. The BILD link is modified to develop stiffness matrices which are a function of internal loads. In general, the equation to define buckling loads is given by

$$[K]\{d\} - \lambda \sum_i \sigma_i [K^i]\{d\} = 0 \qquad i = 1, 2, \cdots, 6$$

where

$[K]$ is the stiffness matrix of the stress-free structure

$\{d\}$ is the vector of gridpoint displacements

λ is the unknown scalar multiplier. It defines the factor by which the prestress-state stresses must be multiplied to induce small deflection buckling

σ_i are the generalized stresses for each element

$[K^i]$ are the stiffness matrix corrections accounting for first order large deflection effects.

The SMAD link provides the capability to easily form the matrix scalar multiplications and summation indicated in forming the incremental stiffness matrix in the equation above. The σ_i scalars may be introduced directly or obtained by multiplying a set of displacements by the stress matrices. These displacements can also be found by solving for a particular loading Then the scalar λ represents the factor by which the loads must be multiplied for buckling to occur. The SMAD link provides a serial and wavefront capability like that of the ADDS link.

D. Root Extractor Link (POWR)

This link uses the power method to determine the fundamental root and associated vector of a square, symmetric matrix. The matrix is limited to in-core sizes, in the sense that only the diagonal and upper off-diagonal elements in variable bandwidth must fit in core. This link is intended to provide the capability to obtain roots and vectors for systems of approximately 500th order when the matrix is symmetric and sparse

E. Input Data Error Diagnostic Link (CHEX)

An error diagnostic link to detect errors in input data is under development to aid in minimizing the computer running time per problem. Currently, submittal of data to the computer for a problem-solving run can be a costly operation if the input data contains errors For some types of errors, particularly in the pseudo instruction program, only one error per run will be detected and diagnostic printout given. To alleviate this problem, a data checking link, CHEX, is being developed. This link is designed to operate on the IBM 7094 as a chain routine that can be called in the same manner as other links. In addition CHEX is coded to be operational as a separate diagnostic package on the IBM 1620 computer with disk filing. This provision allows the checkout of input data decks at reduced cost on the IBM 1620 prior to running the problem on the IBM 7094 to obtain a solution. The CHEX link is a two-segment program that can detect most major errors in an input data deck CHEX will have an appropriate library of comment statements that indicate the nature of each error detected

F. Conical Element Generation Routines (CONE)

For structural analysis of shells of revolution and circular plates, the recently developed truncated-cone finite element can be used to advantage The conditions on loading states for this element are that the distribution be representable by a Fourier series about a circumference. The series may be either symmetric, asymmetric, or a combination of each with respect to the rotational axis of the shell For each harmonic of the series, separate stiffness, stress, and loading matrices are generated. This element when operational in BILD, will be the third basic element-type in the program library.

Theoretical basis for the element is presented in Refs. 13, 14, and 15 Numerical results from idealizations of several shell-type structures with the conical element are also presented in these references.

Future modifications will change the conical shell element equation generation routine to provide a number of arbitrarily-spaced joints on the cone periphery in order that line and facet elements may be joined to it.

References

1. Melosh, R. J., and Christiansen, H. N., *Structural Analysis and Matrix Interpretative System (SAMIS) Program: Technical Report*, Jet Propulsion Laboratory, Pasadena, California, Technical Memorandum 33-311, November 1, 1966
2. Melosh, R. J., et al, *Structural Analysis and Matrix Interpretative System (SAMIS) Program Report Revision 1*, Jet Propulsion Laboratory, Pasadena, California, Technical Memorandum 33-307, December 15, 1966
3. Lang, T. E , *Structural Analysis and Matrix Interpretative System (SAMIS) User Report*, Jet Propulsion Laboratory, Pasadena, California, Technical Memorandum 33-305, March 1, 1967.
4. Archer, J. S., "Consistent Mass Matrix for Distributed Mass Systems," *American Society of Civil Engineers, Structures Journal,* Vol. 89, pp. 161–178, August 1963.
5. Melosh, R. J., and Lang, T. E., "Modified Potential Energy Mass Representations For Frequency Prediction," *Paper Presented at Conference on "Matrix Methods in Structural Mechanics,"* Wright-Patterson Air Force Base, Ohio, October 26–28, 1965
6. Leckie, F. A., and Lindberg, G. M., "The Effect of Lumped Parameters on Beam Frequencies," *Aeronautical Quarterly,* Vol. XIV, Part 3, pp. 224–240, August 1963.
7. Melosh, R. J., "A Flat Triangular Shell Element Stiffness Matrix," *Paper Presented at Conference on "Matrix Methods in Structural Mechanics,"* Wright-Patterson Air Force Base, Ohio, October 26–28, 1965.
8. Utku, Senol, "Stiffness Matrices for Thin Triangular Elements of Non-Zero Gaussian Curvature," *AIAA Paper No. 66-530, Fourth Aerospace Sciences Meeting*, Los Angeles, California, June 1966.

References (contd)

9. Melosh, R. J., *Development of the Stiffness Method to Define Bounds on Elastic Behavior of Structures*, University of Washington, Ph.D Thesis, August 1962, Seattle, Wash.

10. Utku, Senol, *Computation of Stresses in Triangular Finite Elements,* Jet Propulsion Laboratory, Pasadena, California, Technical Report 32-948, June 15, 1966.

11. Hurty, W. C., "Dynamic Analysis of Structural Systems Using Component Modes," *American Institute of Aeronautics and Astronautics Journal,* Vol. 3, pp. 678–685, 1965.

12. Bamford, R M., *A Modal Combination Program for Dynamic Analysis of Structures,* Jet Propulsion Laboratory, Pasadena, California, Technical Memorandum 33-290, August 15, 1966.

13. Percy, J. H., Pian, T. H. H , Klein, S , and Navaratna, D. R., "Application of Matrix Displacement Method of Linear Elastic Analysis of Shells of Revolutions," *American Institute of Aeronautics and Astronautics Journal,* Vol. 3, pp. 2138–2145, 1965.

14 Grafton, P. E , and Strome, D. R , "Analysis of Axisymmetrical Shells by the Direct Stiffness Method," *American Institute of Aeronautics and Astronautics Journal,* Vol 1, pp. 2342–2347, 1963.

15. Jones, R. E., and Strome, D R., "Direct Stiffness Method Analysis of Shells of Revolution Utilizing Curved Elements," *American Institute of Aeronautics and Astronautics Journal*, Vol 4, pp 1519–1525, 1966.