NASA Conference Publication 2059

Research in Computerized Structural Analysis and Synthesis

Research-in-progress papers
presented at a symposium
held at Washington, D.C.
October 30 - November 1, 1978

NASA

NASA Conference Publication 2059

Research in Computerized Structural Analysis and Synthesis

Harvey G. McComb, Jr., *Compiler*
NASA Langley Research Center

Research-in-progress papers presented at a symposium cosponsored by NASA Langley Research Center, Hampton, Virginia, and the George Washington University, Washington, D.C., in cooperation with the National Science Foundation and the American Society of Civil Engineers, and held at Washington, D.C., October 30 - November 1, 1978

NASA
National Aeronautics
and Space Administration

Scientific and Technical Information Office

1978

FOREWORD

A Symposium on Future Trends in Computerized Structural Analysis and Synthesis was held at Washington, D.C., on October 30 - November 1, 1978. NASA Langley Research Center and The George Washington University sponsored the symposium in cooperation with the National Science Foundation and the American Society of Civil Engineers. The purpose of the symposium was to provide a forum for structural technologists, computer hardware experts, and computer software experts to examine recent developments and discuss trends in this very rapidly advancing technology area.

The symposium was organized into 19 sessions with a total of 77 papers. Most of these papers are contained in the proceedings:

Noor, Ahmed K.; and McComb, Harvey G., Jr. (eds.): Trends in Computerized Structural Analysis and Synthesis. Pergamon Press, Ltd., 1978.

Topics discussed in the symposium included

(1) Trends in engineering software development

(2) Supercomputers, minicomputers, and microprocessors

(3) Computer graphics

(4) Computer-aided instruction

(5) Symbolic computing

(6) Numerical analysis

(7) Nonlinear analysis

(8) Structural modeling and material characterization

(9) Structural synthesis

(10) Structural analysis and design systems

(11) Advanced structural applications

This NASA conference publication primarily contains papers presented in three research-in-progress sessions of the symposium. However, the first four papers in this publication were presented at sessions entitled "Structural Analysis and Design Systems," "Potential of Supercomputers and Microprocessors," and "The Current Status of Microcomputer Arrays."

Harvey G. McComb, Jr.
Compiler

CONTENTS

THE ATLAS INTEGRATED STRUCTURAL ANALYSIS AND DESIGN SOFTWARE SYSTEM

Rodney L. Dreisbach
Boeing Commercial Airplane Company

Gary L. Giles
NASA Langley Research Center

SUMMARY

The ATLAS System provides an extensive set of integrated technical computer-program modules for the analysis and design of general structural configurations, as well as capabilities that are particularly suited for the aeroelastic design of flight vehicles. ATLAS is intended for use in a production environment in the aerospace industry, and therefore it provides many user-oriented features and much flexibility of use to meet a wide variety of applications. The system is based on the stiffness formulation of the finite element structural analysis method and can be executed in batch and interactive computing environments on the Control Data Corporation (CDC) 6600/CYBER computers. Problem-definition input data are written in an engineering-oriented language using a free field format. Input-data default values, generation options, and data quality checks provided by the preprocessors minimize the amount of data and flowtime for problem definition/verification. Postprocessors allow selected input and calculated data to be extracted, manipulated, and displayed via on-line and off-line prints or plots for monitoring and verifying problem solutions. The sequence and mode of execution of selected program modules are controlled by a common user-oriented language. The modules can be used to perform a variety of single and multidisciplined structural analysis and design tasks in both large and small problem environments. The system is organized to facilitate maintenance and new developments. A data base and data manager are used for automatic communication of data between program modules and for execution of selected modules with interfaced external programs. Utility user interfaces are provided for interactive execution control, data-file editing, and graphical display of selected data.

INTRODUCTION

Development of ATLAS was initiated by The Boeing Commercial Airplane Company in 1969, and was used initially in 1971 for performing preliminary design studies of the National Supersonic Transport. Continued development efforts have resulted in the release and application of a number of extended versions of the ATLAS System. Parts of the efforts were conducted under the NASA Langley contract NAS1-12911 during the period of 1974-1978 and the Army contract DAAG46-75-C-0072 during 1975.

Use of ATLAS in a number of specific airplane design situations is described in references 1-5 in terms of the problem complexity, the results obtained, cost, and the benefits provided through using ATLAS. These applications involved stress, loads, flutter, weights, and aerodynamic technical disciplines. The automated strength resizing of an arrow-wing supersonic cruise aircraft (ref. 2) with approximately 20 000 design variables demonstrated the practicality of using ATLAS during the preliminary design process. The use of ATLAS to perform a stress analysis of a large sports stadium (3400 nodes, 9600 elements, 20 000 freedoms, 70 million words of storage) and a detailed three-dimensional stress analysis of a gas turbine engine blade (3200 nodes, 350 solid elements, 9500 freedoms, 15 million words of storage) is described in references 5 and 6, respectively, in terms of problem definition, solution approach, data management, and cost. A practical substructured stress analysis of a 747 airplane model (8628 nodes, 13 751 elements, 13 substructures, 218 load cases) was recently performed, with the results correlating well with previous analyses, to demonstrate further the large-problem-solving capabilities of ATLAS. Methods to automate the strength/stiffness (flutter) aeroelastic design process for metallic and composite structural components (refs. 3 and 4) have been implemented in ATLAS during its continued development and application. The system has provided capabilities for thorough and cost-effective evaluation of new airplane designs and advanced structures.

This paper presents an overview of the technical capabilities and the functional organization of version 4 of ATLAS which has been in production use on Boeing projects since March 1977, and has been installed at the NASA Langley Research Center. Particular emphasis is given to the design of the user interfaces, and the current and future system developments are previewed. Further detailed documentation of the system is given in reference 7 which describes the input data and program execution, the program design and data management, the engineering methods, and a number of demonstration problems.

ATLAS TECHNICAL CAPABILITIES

The many technical analytical capabilities of ATLAS can be grouped as (1) linear stress analysis, (2) bifurcation buckling analysis, (3) weights analysis, (4) vibration analysis, (5) flutter analysis, (6) substructured analyses, and (7) structural design. The basis of these computations is the stiffness method of finite element analysis. Within the scope of ATLAS development activities, high priority has been placed on the technological disciplines that contribute directly to the evaluation of strength and stiffness (flutter) characteristics of flight structures.

Each of the foregoing general capabilities can be either combined or further subdivided depending on the type of problem being solved, the selected method of analysis, and the desired end results. The various analytical steps required for solving different types of problems are performed by executing a user-selected set of computational modules (preprocessors, processors, and postprocessors) in a particular sequence. The ATLAS computational capabilities are grouped by modules, each of which is related to a particular engineering discipline. The ATLAS modules and their functions are summarized in table 1.

The ATLAS structural element library includes an axial ROD and BEAM with offsets and linear cross section variations, an elastic-support SCALAR element, triangular and quadrilateral membrane PLATE and membrane/bending GPLATE elements with offsets and orthotropic features, a family of BRICK (3-dimensional) elements with up to three optional nodes per edge and orthotropic features, triangular and quadrilateral CPLATE elements with offsets for composite laminates, and built-up SPAR, COVER and CCOVER elements. A SPAR is comprised of a shear web, linearly tapered upper and lower caps with axial stiffness, and two rigid posts at either end of the web. COVER and CCOVER elements are comprised of two PLATE and CPLATE elements, respectively, each of which is separated by rigid posts between the corresponding corner nodes. SPAR elements are used typically to model ribs and spars, whereas the COVER and CCOVER elements are used to model the structural surfaces of box beam constructions and wing-like configurations for aerospace vehicles.

Unsteady aerodynamic loads for performing automated flutter solutions can be based on the following: (1) strip-theory method for subsonic incompressible flow, (2) assumed-pressure function and doublet-lattice methods for subsonic compressible flow, (3) the Mach-box method for supersonic flow, and (4) structural-flexibility effects associated with the truncated vibration modes.

General features of ATLAS, in addition to those described in table 1, include the following: (1) a free-field input data format, (2) many data default values, (3) common data-generation options for all data types, (4) multiple coordinate systems for node geometry definition and structural response, (5) user-selected data printout, and (6) data management and graphics utilities. The multilevel substructuring capability for stress, mass, and vibration analyses provides automatic management of the substructure interact data for any number of interact levels.

Automated interfaces of execution and problem data between ATLAS and a number of computer programs that are external to ATLAS have been established. These capabilities include an interface of ATLAS structural and mass data with FLEXSTAB (ref. 8) for performing aeroelastic and elastic stability analyses, an interface of FLEXSTAB steady-state loads with ATLAS for performing stress analysis and structural-design functions, and input data interfaces between ATLAS and NASTRAN®.

SYSTEM FUNCTIONAL ORGANIZATION

The design of ATLAS has been based primarily on the convenient user-selection of system functions for performing either single or multidisciplined structural analysis/design tasks in a timely, thorough, and cost-effective manner. Continued emphasis has been directed toward the development of a system that

- Provides a common executive module for convenient and versatile user-controlled technical-analysis flow and control of design cycles

- Provides a common data base that eliminates duplicate input-data preparation and centralizes the control of design data

- Provides data-management algorithms for convenient interfacing of the computational modules with external computer programs

- Uses data preprocessing and data generation codes to minimize the amount of input data and the time required for data preparation and debugging

- Uses automated structural-sizing algorithms to minimize the amount of hand-sizing of structural members, particularly during preliminary design studies

- Provides advanced engineering methodologies equally useful for performing design tasks in a small or large problem environment

- Provides data postprocessing codes to extract, manipulate, and display (print/plot) selected data for monitoring analysis/design activities

- Provides interactive capabilities for editing data files, for executing selected system modules, and for generating on-line and off-line graphical displays

- Provides an open-ended, low-cost admission of new program codes

The architecture developed for ATLAS is a modular system of overlaid program codes with common executive and data-base management components as illustrated in figure 1. Each module performs a well-defined engineering, mathematical, or clerical task. This modular design supports the foregoing system attributes and allows for effective development and maintenance activities. Additionally, through centralized management of the program modules, the reliability of the aggregate code is increased.

User interfaces with ATLAS are defined via the problem-definition data deck and the executive control deck. The problem deck contains the data defining the mathematical model of the physical problem to be analyzed, whereas the control deck defines the analysis functions to be performed.

Generally, the only limitations on problem size are those imposed by practical considerations of job-execution time and by the ultimate capacity of the auxiliary storage devices of the computer installation. Computational procedures have been designed particularly to provide reasonable efficiency for solution of large problems. Sparse matrix solution techniques and automatic out-of-core processing are used for increased cost effectiveness.

The characteristics of the primary components of the ATLAS System will be described in the following order: (1) executive modules, (2) computational modules, and (3) the data base management capabilities. Then, a description of the user interfaces with the entire system are presented.

EXECUTIVE MODULES

The executive modules and their basic functions, as they support analysis control and data communication, are as follows:

<ins>Precompilers</ins> — Translate the user-oriented ATLAS language execution directives, as defined via the control deck, into equivalent FORTRAN statements. The resulting FORTRAN code is compiled at execution time to create an ATLAS Control Program Module.

<ins>Control Program</ins> — Control the execution sequence of selected computational modules and set run-time execution parameters. Each control statement included in a control deck and each interactive control directive initiate one or more execution steps in the analysis of the problem defined by the data deck.

<ins>Interactive Executive</ins> — Interpret module-execution control directives, as input via a terminal keyboard during interactive processing, and perform interactive text editing of data files.

ATLAS (0,0) Overlay — Monitor the execution of all computational modules per instructions from the Control Program.

Execution of ATLAS can be performed in a batch, interactive, or mixed computing environment. The user has complete command of the type and method of analysis and the problem-execution steps to be performed by ATLAS. These functions are defined either by the executive control deck or by terminal keyboard input during interactive processing. Execution directives are used to define the sequence of computations, the execution (run-time) parameters, the management of analysis results (e.g., print, plot), the scheduled restart of problem execution, and the contingencies when data errors are encountered.

The control deck can also be used to perform special analytical computations that are not provided directly by the ATLAS program, manipulate ATLAS data, and manage data for interfacing modules (sub-programs) of the ATLAS System with external computer programs. FORTRAN and SNARK (ref. 9) code can be intermixed with ATLAS statements to create a control deck which may include subroutines and overlays.

COMPUTATIONAL MODULES

The three types of ATLAS computational modules and their basic functions are as follows:

<ins>Preprocessors</ins> — Read, decode and interrogate the problem-definition data deck; generate data based on a minimal number of input parameters; load problem-execution restart data to resume processing.

<ins>Processors</ins> — Perform technical numerical computations.

<ins>Postprocessors</ins> — Extract, format and display (print/plot) input data and

analysis results; save problem-execution restart data for processing by a subsequent job.

Each module is referenced by a different name. Generally, there are a preprocessor and a print postprocessor associated with each technical processor. The preprocessor reads and interprets the set of input data corresponding to that particular technology, whereas the postprocessor generates user-selected formatted printout of the input and calculated data corresponding to that technology. Table 1 contains a summary of all computational modules and their technical functions.

A technical processor performs a task that is related to a particular engineering theory or discipline. The STIFFNESS Processor, for example, contains the code that represents the finite-element structural theory used in ATLAS. This module computes the stiffness and stress matrices for the finite elements used to define the structural model.

Certain processors are utility in nature in that they are used to perform general-purpose, normally out-of-core, mathematical operations. Examples of such operations include assembling of the elemental stiffness, mass, loads and displacement matrices by the MERGE Module, the solution of sets of linear symmetric equations by the CHOLESKY Module, and the matrix additions and multiplications effected by the MULTIPLY Module.

Certain preprocessors and postprocessors are also utility in nature in that they can be used to perform operations for multidisciplinary tasks. These modules are shown separately in figure 1. Examples of such operations are the identification of node and element subsets for structural and mass models performed by the SUBSET-DEFINITION Preprocessor for selected print/plot displays and subsequent data processing or the generation of on-line and off-line plots of selected input and calculated data effected by the GRAPHICS Postprocessor.

DATA BASE MANAGEMENT

Automatic transmission of data from one module to another is accomplished primarily by the use of named, random-access disk files. All input data interrogated by the preprocessors are stored in one file, whereas the data generated by a processor are stored in a separate file that is reserved for that module. Any of the data stored in the data base, however, can be accessed by any computational module. The data file and data matrix names are predefined in the codes with options provided for user-naming of certain utility matrices, thereby providing greater versatility of data management, particularly during large-problem solving. Additionally, checkpoint-restart procedures are provided for convenient, stepwise problem executions.

The SNARK package (ref. 9), which is an integral component of ATLAS, is used by the ATLAS modules to transfer data matrices directly from/to the data files to/from data arrays in central memory. Zero matrix elements are suppressed when a matrix is written to disk. SNARK is also used by the ATLAS modules to manage the blank-common core (the primary, dynamic work area) during the computational processes. The library routines for management of data matrices and blank common, and the user-oriented language for performing matrix and scalar mathematics as provided by the SNARK package, can be used to create a Control Program. These functions allow convenient access and manipulation of ATLAS data via a Control Program to perform special, user-defined computations, or to interface data with computer programs that are external to ATLAS.

USER INTERFACES

The engineer communicates with ATLAS via the problem deck and the control deck (see fig. 1). These decks can be established either in a card deck form or in a disk file. User-defined comments, which do not affect any of the processing activities, may be embedded in the decks for identification conveniences.

Execution of an ATLAS job can be performed in an interactive mode, a batch mode, or in a mixed mode. Generally, small problems are handled most conveniently by interactive processing of the entire job. Large problems are solved typically by performing selected preprocessing and postprocessing activities in an interactive mode, whereas the remaining computational tasks are performed in the batch mode. Criteria used to select job processing modes are schedules, budget, and the allowable on-line computer central memory.

Generally, only ATLAS language statements are needed to create a Control Program. The statement "READ INPUT", for example, is used to read a data deck, whereas execution of a stress analysis can be initiated by a "PERFORM STRESS" statement. Module execution options (e.g. identify load cases or specify convergence criteria) can be selected via a parameter list included in the control statement. Catalogs of control statement procedures can be referenced directly for performing a number of typical structural analyses.

Activities that can be performed interactively during execution of ATLAS include (1) define, interrogate, and/or edit data and control files via user-oriented text-editor requests, (2) execute selected modules to perform computations and to manage data, and (3) create and manipulate plots of selected data. A module control command can be executed immediately after it is input via the terminal, or multiple commands can be entered and stored to create a procedure. Execution of a command procedure is processed either without any terminal interruption or with execution control returned to the terminal user after processing each command.

When ATLAS is executed using a Tektronix 4000-series graphics terminal, the ATLAS graphics conversational mode can be initiated between any job execution steps to create graphical displays of selected design data. The engineer conducts an interactive dialogue by using a function menu, two plot directories, and a plot transformation menu. Figure 2 illustrates the function menu, as well as example GNAME and plot ID directories. The GNAME directory contains a list of the user-specified plot group names, whereas a list of plot identifiers for a certain GNAME is contained in a plot ID directory. A particular plot is displayed on-line by proper selection of a

GNAME entry followed by the appropriate selection from the plot ID directory. New displays of a selected plot can be created by the transformation menu.

A summary of the plot types that can be created by ATLAS is presented in table 2. Various plot-type options, including exploded-model views, viewing positions, scales, annotation, and superimposed "before and after" deformed model plots, can be requested at execution time. Selected plots viewed on the console screen can also be directed to any of the following off-line plot devices: Gerber, CalComp, COMp80, and the PDP-11/Vector-General minicomputer systems.

During execution, ATLAS attempts to trap all possible anomalies. When an ambiguity in the data is detected that can be resolved without user interaction, a warning message is issued and the job execution proceeds uninterrupted. However, when a system error occurs, or when a fatal inconsistency is detected either in the data or in the user-selected execution logic, an error message is issued. In this case, only the execution directives included in an "ERROR PROCEDURE" within the Control Program are processed prior to terminating the job. Example error conditions are when loads are specified for inactive freedoms or when quadrilateral plates have reentrant corners or excessive warpage.

Many solution-accuracy checks are performed automatically or are provided as execution options. In all cases, the results of the checks are identified in the output. Examples of these types of checks include (1) any singularities and the number of significant digits lost during decomposition of coefficient matrices of the finite element equilibrium equations, (2) load-reaction equilibrium, (3) overall weight and c.g. of the model, and (4) equilibrium and orthogonality of vibration and general buckling solutions.

CURRENT AND FUTURE DEVELOPMENTS

Extended ATLAS capabilities developed by Boeing that are scheduled for near-term release for general use include the following:

- Control program procedures for analysis of structures wherein large displacements (geometry modifications), large strains (geometric stiffness effects), and/or nonlinear materials are pertinent

- Multipoint kinematic constraints defined generally by nonhomogeneous equations

- ATLAS-interfaced finite elements
  —Isoparametric membrane plates with up to three user-specified edge nodes; using the quarter-point singularity options, the residual strength of complex damaged structure can be investigated
  —An isoparametric laminated plate with coupled membrane/bending behavior for analysis of composite structure, particularly interlaminar shear

- Oscillatory aerodynamic effects from steady lift and drag forces on flutter characteristics

- User specification of select vibration modes for parametric flutter studies
- Flutter optimization procedures to modify a flutter-prone structure with a minimum weight penalty
- Conversational input of problem data and module execution directives for select types of analyses
- New plot types for more convenient interpretation of stress data and aerodynamic data
- Interactive graphics query of the ATLAS data base using the current ATLAS dictionary of data components and a relational algebra data model to print and plot user-selected design data

Longer term plans include the use of minicomputers as terminals to the large computer for performing ATLAS analyses (ref. 10) and the integration of capabilities into ATLAS for synthesis of flight-control and propulsion/airframe systems, aero-acoustic structural response, design of detailed structural components, and durability/damage tolerant analyses.

Centralized controlled procedures have been used for development and integration of new analytical capabilities and system concepts, for program version control and for system documentation. Based on periodic reviews of the technical requirements, an extended version of the system is developed and is subjected to exhaustive qualitative and quantitative modular and system-level checkout tests prior to its release for general use. In concert with continued system developments, it is necessary that the following factors be acknowledged: management acceptance, engineering acceptance, control and management of program changes and growth, as well as performance/cost characteristics.

CONCLUDING REMARKS

The ATLAS integrated software system has provided for cost-effective analysis and design of structures in a production environment. Its use in the earliest stages of the interdisciplined aeroelastic design process has resulted in a thorough understanding of complex structural behavior for a variety of applications. Efficient computer processing and user-oriented features have reduced the cost and flowtime for solving large and small problems involving single or multidisciplined design tasks. Continued developments and wider applications have helped define the requirements for extending the ATLAS analytical capabilities and user interfaces toward a complete, unified computer-aided-design system for performing more detailed and timely aerospace vehicle structural designs.

REFERENCES

1. Thomas, R. M.; Backman, B. F.; Flood, F. D.; Gray, F. P.; Hansteen, H. B.; Pratt-Barlow, C. R.; and Wahlstrom, S. O.: Aircraft Strength and Stiffness Design Automation. USA-Japan Design Automation Symposium, Tokyo, Japan, Aug. 1975.

2. Robinson, J. C.; Yates, E. C., Jr.; Turner, M. J.; and Grande, D. L.: Application of an Advanced Computerized Structural Design System to an Arrow-Wing Supersonic Cruise Aircraft. AIAA Paper No. 75-1038 presented at the AIAA Aircraft Systems and Technology Meeting, Los Angeles, Calif., Aug. 1975.

3. Turner, M. J.; and Grande, D. L.: Study of Metallic Structural Design Concepts for an Arrow Wing Supersonic Cruise Configuration. NASA CR-2743, Dec. 1977.

4. Turner, M. J.; and Grande, D. L.: Study of Advanced Composite Structural Design Concepts for an Arrow Wing Supersonic Cruise Configuration. NASA CR-2825, Apr. 1978.

5. Miller, R. E., Jr.; Hansteen, H. B.; Samuel, R. A.; Backman, B. F.; Lewis, C. M.; and Varanasi, S. R.: Recent Advances in Computerized Aerospace Structural Analysis and Design. Computers and Structures, vol. 7, no. 2, Apr. 1977 (also presented at the Second National Symposium on "Computerized Structural Analysis and Design," George Washington University, Washington, D.C., Mar. 1976).

6. Lewis, C. M.; Samuel, R. A.; and Yen, F.: 3-D Stress Analysis of a Turbine Blade. Army Materials and Mechanics Research Center Report, AMMRC CTR-77-14, Mar. 1977 (reprint of Boeing document D6-42735 prepared under contract DAAG46-75-C-0072).

7. ATLAS-An Integrated Structural Analysis and Design System. NASA Contract NAS1-12911, Boeing Co.: Dreisbach, R. L.: ATLAS User's Guide. vol. 1, May 1978; Erickson, W. J.: System Design. vol. 2, Mar. 1977; Dreisbach, R. L.: User's Manual. vol. 3, Mar. 1977; Gray, F. P.: Random File Catalog. vol. 4, Apr. 1977; Samuel, R. A.: Demonstration Problems, vol. 5, June 1977; Samuel, R. A.: LOADS Module Theory. vol. 6, Oct. 1975; Backman, B. F.: DESIGN Module Theory. vol. 7, May 1977.

8. Hink, G. R.; et al.: A Method for Predicting the Stability Characteristics of an Elastic Airplane, vol. III - FLEXSTAB Program Manual. NASA CR-114714, Oct. 1974.

9. Erickson, W. J.: SNARK User.s Manual. Boeing document BCS-G0686, June 1975.

10. Storaasli, O. O.: On the Role of Minicomputers in Structural Design. Computers and Structures, vol. 7, pp. 117-123, Feb. 1977 (also presented at the Second National Symposium on "Computerized Structural Analysis and Design," George Washington University, Washington, D.C., Mar. 1976).

TABLE 1.- ATLAS PREPROCESSORS,PROCESSORS AND POSTPROCESSORS

MODULE NAME	PREPROCESSOR	PROCESSOR	POSTPROCESSOR	TECHNICAL FUNCTION
ADDINT		●	●	Add and/or interpolate with respect to reduced frequencies those generalized airforce matrices generated by AF1, DUBLAT, FLEXAIR, MACHBOX, RHO3, or by previous execution of ADDINT
AF1	●	●	●	Define aerodynamic model; Calculate subsonic, incompressible-flow aerodynamic loads for FLUTTER; Strip theory method
BC (BOUNDARY CONDITION)	●		●	Define boundary conditions for structural model
BUCKLING		●	●	Calculate bifurcation buckling loads and mode shapes
CHOLESKY		●		Solve systems of linear symmetric equations
DESIGN	●	●	●	Define structural resize data; Perform regional optimization of composite structures; Resize structural models based on a fully-stressed design, thermal and local-buckling effects, and geometric and margin-of-safety constraints
DETAIL	●			Define finite-element cross-section shapes for thermal gradients and stiffener locations for plate-element local buckling
DUBLAT	●	●	●	Define aerodynamic model; Calculate subsonic compressible-flow aerodynamic loads for FLUTTER; Doublet-lattice method
EXTRACT			●	Extract selected problem-definition and analysis data from the primary ATLAS data-base for GRAPHICS
FLEXAIR		●	●	Calculate generalized airforce matrices that include flexibility effects of truncated structural modes
FLUTTER	●	●	●	Define structural damping; Modify and solve the flutter equations
FREEBODY			●	Print internal nodal forces acting on selected finite elements
GEOMETRY	●			Define, generate, and interrogate three-dimensional geometry components for structural and mass models
GRAPHICS			●	Generate online and offline plots of data selected via EXTRACT; Batch and interactive-graphics modes of execution
INTERACT	●		●	Define substructure interaction problem
INTERPOLATION		●		Define mode-shape interpolation functions for AF1, DUBLAT, FLEXAIR, MACHBOX and RHO3
LOAD	●			Load previously-generated problem data from SAVE to restart execution
LOADS	●	●	●	Define static loads; Calculate nodal loads due to inertial forces, pressure gradients, and thermal gradients; Cumulate static loads and specified displacements
MACHBOX	●	●	●	Define aerodynamic model; Calculate supersonic-flow aerodynamic loads for FLUTTER; Mach Box method
MASS	●	●	●	Define mass model that may complement STIFFNESS; Define fuel and payload management data; Generate mass matrices and detailed weight statements for primary and secondary structure, fuel, and payload
MATERIAL	●		●	Define finite-element material property data and design-allowable stresses for STIFFNESS, STRESS, MASS, and DESIGN
MERGE		●		Assemble stiffness, mass, loads, and displacement matrices
MULTIPLY		●		Add and multiply matrices
NODAL	●		●	Define local coordinate systems and nodal data for STIFFNESS and MASS models
PRINT			●	Print system-generated matrices
REACTION			●	Print reaction forces and load-reaction equilibrium checks for structural model
RHO3	●	●	●	Define aerodynamic model; Calculate subsonic compressible-flow aerodynamic loads for FLUTTER; Assumed pressure modes method
SAVE			●	Save problem data for execution checkpoint; Use LOAD to restart execution
STIFFNESS	●	●	●	Define structural model; Generate elastic-stiffness, geometric-stiffness, and stress matrices for elements
STRESS	●	●	●	Assemble nodal displacements; Calculate element stresses and internal nodal forces; Superimpose displacements and stresses
SUBSET DEFINITION	●			Define subsets of nodes and elements in STIFFNESS and MASS models for subsequent processing; Define subsets of data-component labels for EXTRACT
VIBRATION		●	●	Calculate natural vibration frequencies and mode shapes in addition to generalized mass and stiffness matrices

TABLE 2.- TECHNICAL-DATA PLOT TYPES

TECHNICAL DATA DISPLAY	PLOT TYPE					
	UNDEFORMED GEOMETRY	DEFORMED GEOMETRY	SCALAR-GRID	CONTOUR	GRAPH	MATRIX
STRUCTURAL AND/OR MASS MODEL						
• Nodes	●					
• Stiffness finite-element grid	●					
• Stiffness element properties			●	●		
• Mass finite-element grid	●					
• Exploded node/grid subsets	●					
DISPLACEMENTS/STRESSES						
• Nodal displacements		●				
• Element stresses			●	●	●	
STRUCTURAL RESIZE RESULTS						
• Stiffness element properties			●	●		
• Strength-designed margins of safety			●	●	●	
• Thermal-designed margins of safety			●	●	●	
MODE SHAPES						
• Vibration		●				
• Buckling		●				
FUEL/PAYLOAD MANAGEMENT						
• Loadability diagrams					●	
FLUTTER SOLUTION						
• V-g and V-f graphs					●	
MATRIX DATA						
• Any ATLAS matrix						●

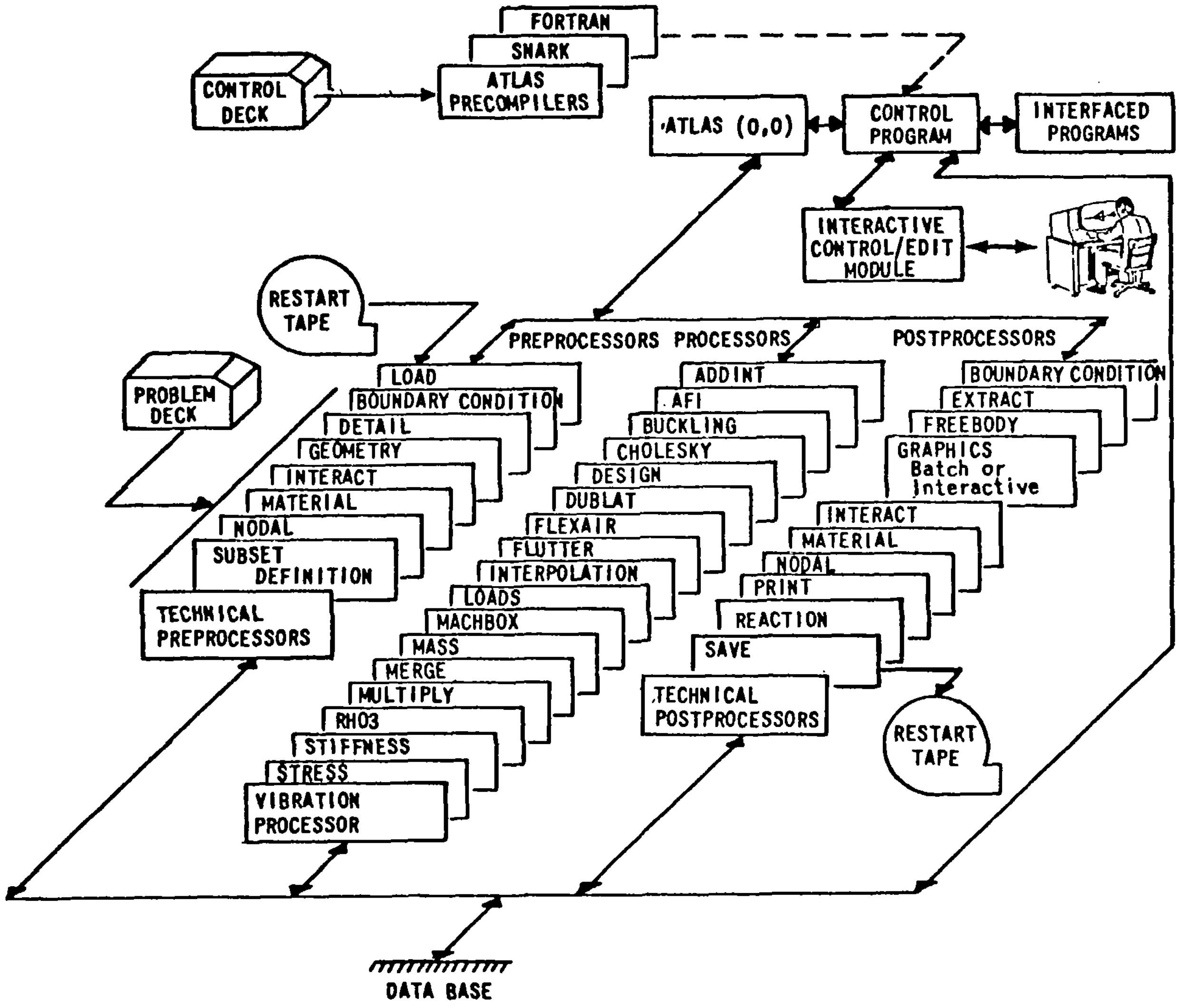
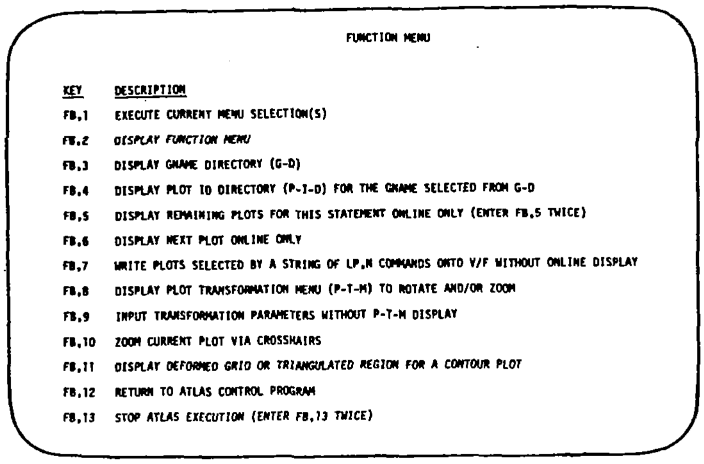


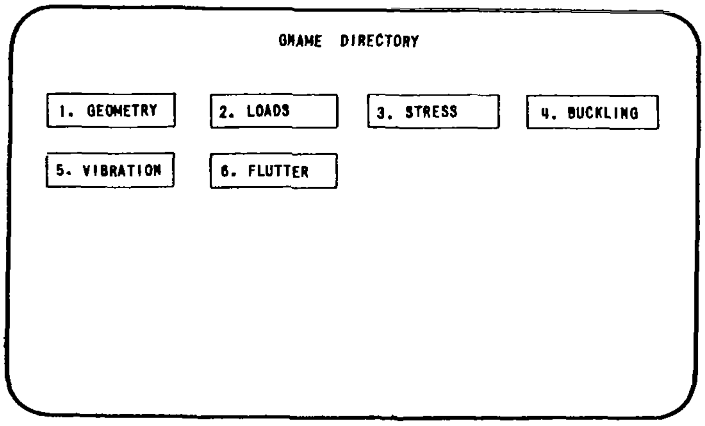
Figure 1.- The ATLAS system modular design.



```
                         FUNCTION MENU

KEY     DESCRIPTION
FB,1    EXECUTE CURRENT MENU SELECTION(S)
FB,2    DISPLAY FUNCTION MENU
FB,3    DISPLAY GNAME DIRECTORY (G-D)
FB,4    DISPLAY PLOT ID DIRECTORY (P-I-D) FOR THE GNAME SELECTED FROM G-D
FB,5    DISPLAY REMAINING PLOTS FOR THIS STATEMENT ONLINE ONLY (ENTER FB,5 TWICE)
FB,6    DISPLAY NEXT PLOT ONLINE ONLY
FB,7    WRITE PLOTS SELECTED BY A STRING OF LP,N COMMANDS ONTO V/F WITHOUT ONLINE DISPLAY
FB,8    DISPLAY PLOT TRANSFORMATION MENU (P-T-M) TO ROTATE AND/OR ZOOM
FB,9    INPUT TRANSFORMATION PARAMETERS WITHOUT P-T-M DISPLAY
FB,10   ZOOM CURRENT PLOT VIA CROSSHAIRS
FB,11   DISPLAY DEFORMED GRID OR TRIANGULATED REGION FOR A CONTOUR PLOT
FB,12   RETURN TO ATLAS CONTROL PROGRAM
FB,13   STOP ATLAS EXECUTION (ENTER FB,13 TWICE)
```

(a) Graphics function menu options.

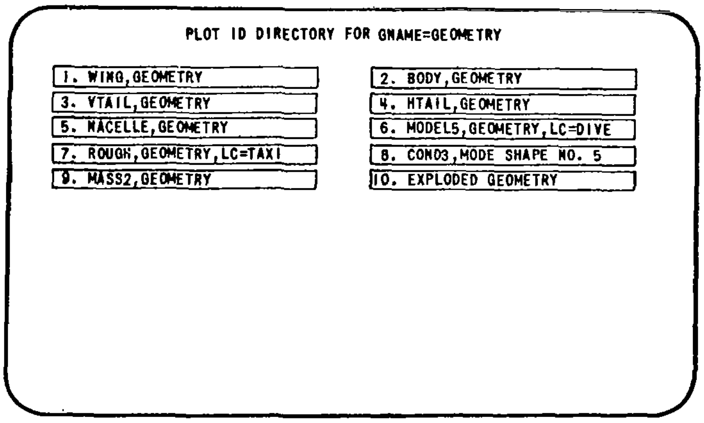


```
                  GNAME  DIRECTORY

1. GEOMETRY    2. LOADS    3. STRESS    4. BUCKLING

5. VIBRATION   6. FLUTTER
```

(b) Example GNAME directory.



```
          PLOT ID DIRECTORY FOR GNAME=GEOMETRY

1. WING,GEOMETRY                 2. BODY,GEOMETRY
3. VTAIL,GEOMETRY                4. HTAIL,GEOMETRY
5. NACELLE,GEOMETRY              6. MODEL5,GEOMETRY,LC=DIVE
7. ROUGH,GEOMETRY,LC=TAXI        8. COND3,MODE SHAPE NO. 5
9. MASS2,GEOMETRY               10. EXPLODED GEOMETRY
```

(c) Example plot ID directory.

Figure 2.- ATLAS graphics function menu and example plot directories.

NUMERICAL AERODYNAMIC SIMULATION FACILITY

F. R. Bailey and A. W. Hathaway
NASA Ames Research Center

INTRODUCTION

The rapid advancement of computer speed and storage over the last two decades has fostered an equally rapid advancement in computational fluid dynamics simulation capability. A major growth field is computational aerodynamics, which combines the disciplines of aerodynamics, fluid physics, mathematics, and computer science for the purpose of simulating aerodynamic flow fields through the numerical solution of approximating sets of fluid dynamics equations. The field of computational aerodynamics, even in its current formative stage, is emerging as an important aerodynamic research and design tool.

Critical to the advancement of computational aerodynamics capability is the ability to simulate flows about three-dimensional configurations that contain both compressible and viscous effects, including turbulence and flow separation at high Reynolds numbers. While it is currently possible to accomplish this in two dimensions, bridging the gap to three dimensions is beyond the capability of current computers.

The Numerical Aerodynamic Simulation Facility (NASF) is proposed to provide this needed increase in computational capability by carefully matching the characteristics of the problems to be solved with advances in computing system architecture. The NASF Project has been under way for approximately three years, and the remainder of this paper will describe Project activities to date.

CHARACTERISTICS OF NAVIER-STOKES SOLUTION ALGORITHMS

The first step in the development of the NASF was to ascertain whether the problems to be solved were such that architectural innovations could reasonably be expected to produce the performance gains desired. To this end, analyses were conducted of two solution techniques for solving the Reynolds averaged Navier-Stokes equations describing the mean motion of a turbulent flow with certain terms involving the transport of turbulent momentum and energy modeled by auxiliary equations. The first solution technique (ref. 1) is an implicit approximate factorization finite-difference scheme applied to three-dimensional flows. The implicit formulation avoids the restrictive stability conditions when small grid spacing is used to obtain spatial resolution of large gradients in viscous dominated regions. The approximate factorization reduces the solution process to a sequence of three one-dimensional problems with easily inverted matrices. The second technique (ref. 2) is a hybrid explicit/implicit finite-difference scheme which is also factored and applied to three-dimensional flows. In this scheme, implicit techniques are used only in the surface normal direction where the grid spacing is the finest. Both methods are applicable to problems with highly distorted grids and a variety of boundary conditions and turbulence models. These early analyses indicated the following three fundamental characteristics:

1) Massive but rather simple calculations

   While a typical Navier-Stokes solution will involve approximately a trillion calculations, these calculations are almost evenly divided between adds and multiplies, with very few divides (on the order of 3%) and essentially no intrinsic functions such as square root. In addition, almost one-third of all computations are in the form of "multiply-add" combinations.

2) Massive but well structured data bases

The large number of grid points necessary to describe non-trivial geometries gives rise to data bases for Navier-Stokes problems of on the order of 40 million words. In addition, this data must be accessed in each of the three spatial directions associated with the factored algorithm approach at least once per iteration. There is only local grid data interaction, however, allowing the sweeps through the data to take place independently.

3) Simple control

The basic algorithms consist of identical operations performed on large blocks of data, which makes them quite well suited to parallel solution approaches. In addition, there are relatively few recurrences or branches, and those that do exist are within deeply nested loops, again allowing the use of parallelism to improve computational speed.

These results clearly indicated that the Navier-Stokes algorithms were in fact ideally suited to parallel processing techniques, the only architectural alternative for meeting NASF goals in a reasonable time frame.

PRELIMINARY NASF SPECIFICATIONS AND GUIDELINES

In addition to providing advanced capability for computational aerodynamics research, a basic goal for the NASF is to be a tool to aid in the design of aerospace vehicles. To serve such a role, it is necessary that solutions be available in relatively short times to make it practical to sort through many possible configurations early in the design cycle when aerodynamic factors have the largest impact on the shape of a new vehicle. This gives the following basic statement of required NASF performance:

> The solution of the Reynolds-averaged Navier-Stokes equations, for grids of one million points, in less than ten minutes.

Assuming reasonable advances in numerical method efficiency (estimating a factor of four improvement by 1983), this in turn gives the following estimates for processing rate and storage:

> A processing rate of approximately one billion floating point operations per second, for the add/multiply/divide mix of Navier-Stokes algorithms, on a data base of 40 million words.

This number of one billion floating point operations per second has served as a quick-reference performance figure throughout the Project. It should be kept in mind, however, that the basic performance goal of the NASF is <u>not</u> stated in terms of raw processing rate, but rather in terms of elapsed time for Navier-Stokes solutions. This of course has profound implications on total system architecture, benchmarks, and so forth.

In addition to the above performance goal, the NASF has very stringent requirements on reliability and trustworthiness. Reliability in this sense means that the machine must be capable of performing useful work a high percentage of the time -- it must not break very often and when it does, it must be easy to repair -- and trustworthiness means that the computations it does produce must be correct -- the user must not have any reason to distrust results. The NASF will be a very large and very complex system; without early and strong emphasis on both reliability and trustworthiness its value for practical engineering use can be severely compromised.

In order to assure the attainment of these goals, one other concept is being strongly emphasized throughout the project: the NASF is <u>not</u> being developed as a computer science project, but rather as the construction of a fluid dynamics research and engineering tool. Therefore, while state of the art technologies must be used throughout in order to reach the substantial performance expectations, there are no specific attempts being made to <u>stretch</u>

the state of the art in individual areas. For example, a basic concept of the Facility views it as being made up of two separate components:

Flow Model Processor (FMP)

> The FMP is the computational engine and storage for solving the Navier-Stokes problems. As it will operate on only one job at a time, it will also contain sufficient staging memory to allow buffering of output for the current and previous job as well as input of code and data for the next job.

Support Processor System (SPS)

> The SPS will be responsible for the overall operation of the Facility, including compiling and scheduling jobs for the FMP, managing the file system, and providing an interactive user interface.

In this concept, then, essentially <u>all</u> custom hardware will be in the FMP itself; the SPS will be a standard, proven, off-the-shelf computing system. Likewise the development of custom software will be minimized: the FMP will have the minimum operating system possible for staging of input and output, and the SPS will have minimum modifications to its standard operating system. This philosophy is also being followed in the development of the FMP programming language. First, it will be based on ANSI FORTRAN 77, with only the minimum extensions required for expressing problem parallelism and memory hierarchy management. Second, there will be <u>no</u> attempt made to have the FMP compiler automatically recognize parallelism in standard FORTRAN constructs; <u>all</u> problem parallelism must be explicitly specified by the programmer. It is strongly felt that both of these restrictions are necessary in order to develop, in the time frame necessary, an operational compiler capable of delivering acceptable FMP performance.

Thus we see that while the NASF has substantial goals in terms of performance and reliability, every effort is also being taken to ensure that it is actually buildable and usable. The Facility is not based on any breakthroughs in either hardware or software technology and will instead achieve its goals through architectural innovations and through the combining of several state of the art technologies under strict reliability guidelines and controls.

PRELIMINARY FEASIBILITY STUDIES

After the encouraging results of the early Navier-Stokes analyses, two independent, parallel feasibility studies were conducted, one by Burroughs Corporation and one by Control Data Corporation. The major efforts of these contracts were the analysis of the two Navier-Stokes algorithms described above and the development of candidate processor architectures optimized for their execution. Extensions of these studies further refined the two baseline configurations, including the development of simulators for performance and functional verification. These two baseline configurations are described later.

A second phase of feasibility studies is currently under way, with the same contractors, with the goal of extending the baseline designs even further, including software and total facility specifications. These further studies will also assess the proposed configurations for weather/climate simulation and will investigate modifications which could improve FMP performance for such applications.

In addition to these two studies, Ames is involved in numerous other contract efforts. Most of these efforts are for technical assistance in areas such as evaluation of feasibility study results, reliability considerations, and performance analysis.

Before discussing the baseline architectures in detail, it is necessary to review two fundamental concepts.

FUNDAMENTAL ARCHITECTURAL CONCEPTS

The first of these concepts is that of parallelism, as illustrated in figure 1. As indicated earlier, Navier-Stokes solutions involve many operations being done repetitively on large blocks of data points; this is illustrated by the program segment in figure 1a. This loop will cause the three operations indicated by the boxes to be done N times, once for I=1, once for I=2, and so forth; each execution of the three operations is called an "instance." Further it is assumed that the N instances are data independent, that results of one instance are not used as inputs by another. (This condition is generally satisfied by the types of problems envisioned for the NASF.) Doing the N independent instances of the loop sequentially, as on current general purpose computers, is illustrated in figure 1b. Since this cannot hope to achieve the processing rate planned for the NASF, some form of parallelism must be used; this is illustrated in figure 1c.

Given this general form of parallelism, it is then necessary to map the N instances of the loop onto physical hardware; methods for doing this are shown in figure 2. One technique, known as "horizontal slicing," consists of performing all N instances of the first operation in the loop, then performing all N instances of the second operation, and so forth. This is the technique used in vector or pipeline computers such as the STAR-100, the ASC, and the CRAY-1. The second general technique, known as "vertical slicing," consists of assigning the N independent instances to N independent processors and having them each perform all of the operations for one particular instance. This technique is commonly referred to as parallel or concurrent processing.

Both forms of slicing assume that in fact N operations can be done in parallel, and unfortunately this assumption is frequently not valid on real hardware. In this case, the problem is solved as a combination of serial and parallel computations. First assume a vertical slicing approach, with P processors available (P<N). The N instances are grouped into M "cycles," with P instances per cycle. M is of course chosen so that MxP is equal to or slightly larger than N. The P processors are then started computing the instances in the first cycle. When they complete, the second cycle is done, and so forth until all M cycles are completed (with some processors possibly being idle during the last cycle). This same problem can manifest itself in the horizontal slicing approach as well if the number of data elements which can be operated on at once is less than N. This can be caused either by exceeding the maximum allowable vector length (as on the CRAY-1 if N>64) or by having insufficient memory to store long vectors of temporary results (as frequently happens on the STAR-100). Again the solution is to perform a series of parallel operations.

Figure 3 illustrates a slightly different problem, more typical of applications other than Navier-Stokes. As shown in figure 3a, the N independent instances in this case involve data dependent or subscript dependent branching ("subscript dependent" means things like boundary conditions). In the horizontal slicing approach, shown in figure 3b, the steps are as follows: perform the first operation for all N instances, make the decision for all N instances (storing the result as a logical or bit vector), perform the third step for all applicable instances (under control of the logical vector produced earlier), perform the alternate third step for the other instances (again controlled by the logical vector), and finally perform the fourth step for all. Actual implementations of "under control of the logical vector" may involve techniques such as sparse vector processing, "gather" operations, index lists, and so forth. Additional overhead is required, however, and all operations must take place in rigid sequence. In the vertical slicing approach each of the individual processors is free to take only the appropriate branch; there is no need to wait on the alternate not selected. This means that the net time for completing all instances can be significantly reduced. Note also that in this type of problem the horizontal slicing approach is illustrative of both vector processing (STAR-100, CRAY-1, etc.) and lockstep parallel processing (ILLIAC IV, BSP, etc.); the vertical slicing approach is applicable only if the processors are able to execute independently.

The other basic concept relates to storage and accessing of three-dimensional data, as required by the 3-D Navier-Stokes codes. As shown in figure 4, sweeping through the data in each of the three directions associated with the factored algorithms involves picking up planes in each of the three orientations. That is, when advancing in the X direction (having the I subscript advance once per loop), it is necessary to operate on Y-Z planes as vectors (the N instances are composed of all J and K values for the current value of I). Similarly, the Y direction sweep fetches X-Z planes and the Z direction sweep fetches X-Y planes. Since computer memory is not oriented in three dimensions, some mapping must be applied between the high level language construct of a three-dimensional array and the linear memory of a computer; an example based on FORTRAN is given in figure 5. Here we can see that accessing the data in all three directions involves not only fetching contiguous words (the most efficient in most architectures), but also groups of words and individual words separated by constant intervals. Thus the requirement that the FMP permit efficient three-dimensional access to the entire 40 million word data base has significant impact on the FMP architecture.

Note finally that the orderly data accessing of the Navier-Stokes algorithms, together with the local grid data interaction, allows the use of a multi-level memory hierarchy. That is, while it is necessary to have extremely fast access to data being fed directly to processors, other data (previous or subsequent planes) may be contained in a slower, block-accessed backing store, to be moved into the faster processor memory only as required.

We will now describe the baseline NASF configurations developed by the two contractors. Both concepts are still in a state of development and should be considered preliminary and subject to modifications as indicated by further analyses. Nevertheless, they indicate the basic directions being pursued.

CONTROL DATA BASELINE CONFIGURATION

Figure 6 shows a simplified block diagram of the Control Data FMP concept. The basic philosophy espoused by CDC was to build the absolutely fastest pipeline processing unit possible, and then combine as many of them as required to meet the processing rate requirements of the NASF. As might be expected, this gave rise to an architecture which is quite similar to the STAR-100: there is a scalar processing unit, which also contains a control unit to do instruction processing; a lockstep group of eight vector pipeline processing units, with buffer registers; a memory mapping unit, allowing complex memory accessing modes at nearly full vector speed; a main random access memory of eight million words; and a block-accessed CCD (charge coupled device) memory of 256 million words. This CCD memory will serve both as the staging memory (for buffering FMP input and output) and as the primary storage for grid data, with appropriate blocks being moved into the eight million word main memory for actual processing. The architecture also provides for both 64-bit and 32-bit processing modes, with 32-bit processing taking place at twice the 64-bit rate.

Reliability features of the CDC architecture are substantial. All memory contains single-error-correction-double-error-detection (SECDED) circuitry, with appropriate block error correcting codes in the large CCD memory. In addition, the vector processors employ a unique variable-redundancy concept which provides substantial error detection. Each processor actually consists of two identical, independent sets of functional units, plus an input operand select unit and a set of coincidence checkers. For simple operations, such as AxB, the same operand streams are sent to each set of functional units and the units perform identical, redundant calculations. The outputs of each pair of redundant components are sent to the coincidence checkers, providing complete redundancy and a very high degree of error checking. As mentioned above, however, many computations in Navier-Stokes problems are of more complex forms, such as (A+B)x(C+D). The vector processors are also capable of performing such complex calculations although in this case all components would be needed in the actual computation, giving no redundancy and no checking. Operations of intermediate complexity also exist, which allow some units to operate redundantly and be checked while others operate independently. Thus there is a dynamic trade-off being made between maximum speed, in which all units operate independently, performing three floating point operations every machine cycle, and maximum error detection, with all units performing redundant, checked calculations. It is felt that a normal instruction mix will provide ample checking while still allowing the peak performance rates required of the NASF.

In addition to the error detection provided within the pipelines, there is also redundancy in the form of a ninth, spare processing unit which can be activated at any time. Thus if a processor fails, the spare processor can be configured in and the FMP can resume operation while the failed processor is being repaired. Note that no instruction re-try is planned, however; if a processor fails, the currently active job is aborted prior to reconfiguring the pipelines and resuming operation.

The scalar unit of the Control Data FMP is very similar to the scalar unit of the STAR-100A. This is expected both to save design effort and to produce a more reliable device.

The map unit is a substantial extension of the current STAR-100 architecture in that it provides the capability of accessing memory in each of the three directions shown in figure 5: contiguously (Z sweep), evenly spaced groups (Y sweep), or evenly spaced words (X sweep). There can be a significant performance degradation for non-contiguous accesses, but the map unit operates concurrently with many vector operations, allowing complex fetching of the next vector to overlap with processing of the current one.

Since the Control Data approach is basically horizontal slicing, as is the STAR-100, the proposed FMP programming language draws heavily from STAR FORTRAN. Unfortunately STAR FORTRAN is hardly an ideal high level language due to the need for frequent use of the "Q8" construct, which is essentially embedded assembly language. CDC does feel that they have learned quite a bit from the STAR FORTRAN experience, however, and that reasonable FORTRAN syntactic extensions can in fact be devised which will allow true high level specification of horizontal slicing parallelism. An early attempt had also been made to specify a vertical slicing extension, using a construct which was similar to the Burroughs proposal (to be described later). It was felt that this would require considerably more compiler sophistication than was reasonable, however, and so the approach was dropped.

BURROUGHS BASELINE CONFIGURATION

In the Burroughs NASF baseline configuration, the FMP-SPS interconnection is quite similar to that used for the Burroughs Scientific Processor (BSP) to build on experience and save development costs. The Burroughs FMP design is not at all similar to the BSP, however; figure 7 shows a simplified diagram of the proposed FMP.

The approach taken by Burroughs is also quite different from that of Control Data. Here the processors are relatively simple devices, but the replication factor is much higher. There are 512 independent processors proposed, each with 32,768 48-bit words of local storage for program and data. There is also a control unit, which controls overall FMP operation and allows synchronization of the processors. The control unit does <u>not</u> do instruction decoding and sequencing for the processors, however; the FMP is <u>not</u> a lockstep machine like the ILLIAC IV or the BSP. Instead, each processor has its own program storage and instruction decoding capability. While this potentially allows the FMP to operate as a full multiple instruction multiple data stream (MIMD) machine, this is not the mode of operation planned for the NASF. The primary reasons for this are the primitive interprocessor cooperation capability provided (there is no automatic "locking" of data in extended memory, no capability for synchronization of subsets of processors, etc.) and the lack of a sophisticated program distribution mechanism (it is planned that the same program will be broadcast to all processors at once by the control unit). Therefore the Burroughs FMP is envisioned as a "synchronizable array machine," in which the processors will in fact all execute the same code with frequent synchronization although individual processors will be able to take different data dependent branches and also to utilize data dependent arithmetic algorithms as well as individual instruction retry and other error recovery procedures.

The memory associated with each processor will in actuality be treated as scratchpad storage, with problem grid data being passed through the processor memories as planes appropriate to the current sweep direction. The actual problem data base will be held in a 34 million word random access memory, which transfers data to and from the processor memories through a transposition network (to be described later). This 34 million word extended memory is organized as 521 separate modules, the prime number 521 being chosen to minimize bank conflicts. That is, with a prime number of memory banks, the only conflicts which will arise are for data intervals of exact multiples of 521. With other numbers of banks, say 512, conflicts could occur for intervals of 2, 4, 8, 16, and so forth.

The Burroughs configuration also utilizes CCD technology for the staging memory, with a proposed 134 million words. Transfers to and from the 34 million word extended memory would be on a job basis, rather than the plane-at-a-time access planned by Control Data.

The Burroughs design also makes use of SECDED in all memories, but error detection within the processors poses a quite different problem than in the fewer pipeline processors of the CDC design. Studies are currently under way to determine appropriate error detection mechanisms for such a large array of processors.

The transposition network originally proposed by Burroughs has undergone some fundamental changes recently, as follows. The original FMP design was tailored specifically to Navier-Stokes problems, and as mentioned earlier, these problems have well structured data accesses and relatively few branches. Thus a network was designed which would provide conflict-free (that is, full speed) accessing for words separated by a constant skip distance. Such a sequence of words located at a constant interval of p words is called a "p-ordered vector." Thus the network would handle the X and Z sweeps at full speed, and sufficiently large arrays would also be efficient in the Y sweep (if the number of words in each group is large compared to the number of processors). This network was quite simple and easy to control, but it did require that all processors access data from different extended memory modules (which will always be the case with p-ordered vectors, due to the prime number of banks) and that all processors be synchronized prior to extended memory accessing.

Subsequent feasibility studies were concerned with the suitability of the FMP for other problems, however, notably weather/climate simulation codes. These programs have significantly more data dependent branching, as well as less orderly data accessing. In fact, analyses showed that significant performance degradations were being caused by the need for synchronization and the restricted extended memory accessing permitted. To solve this problem, a new transposition network is currently under investigation. This network is much more general than the previous one, with processors making requests independently and the network itself determining what path conflicts exist and temporarily blocking conflicting requests. The basic interconnection of the network is similar to an omega network, but the design calls for a self-setting mode of control. That is, rather than having the control unit compute the proper setup for the network (a computation which can guarantee optimum operation but which is prohibitively complex), the network is set up by combinatorial logic cascading through the individual stages. Once a path is established between a processor and an extended memory module, that path is held until all data are passed. The path is then released, allowing blocked requests to proceed.

Simulations of various mechanisms for implementing the new network are in progress, and preliminary indications are that a self-setting network can in fact be built which is as efficient as the old network for contiguous words, only slightly less efficient for evenly spaced single words, and significantly more efficient for spaced groups, particularly small groups. Thus the new network may cause a small degradation for Navier-Stokes problems (although even this is not obvious since the network accounts for relatively little of the Navier-Stokes time) but should be substantially better for other applications with more random data accessing characteristics.

The Burroughs FMP is basically a vertical sliced architecture: the processors are independent during execution of a cycle of instances, with synchronization necessary only at top and bottom. As such, the programming language extensions are minimal and quite easy to use. Currently the major extension is a DOALL construct, which performs essentially like a standard FORTRAN DO-loop except that the programmer has promised that all instances of the loop are data independent, and thus may all be done in parallel. The compiler has a relatively easy job of carving up instances into cycles, and most serial optimization techniques are applicable.

PERFORMANCE EVALUATIONS

As mentioned earlier, Ames is undertaking independent performance evaluations of each of the baseline FMP configurations. This is being done by first programming each of the codes in the appropriate FMP FORTRAN, then hand compiling them (assuming an unsophisticated compiler), and finally timing the machine code, either by hand or with simulators.

The results of these evaluations, together with maximum processing rates quoted by Control Data and Burroughs, are shown in figure 8. All performance numbers are in terms of billions of floating point operations per second; note that the initial performance specification for the FMP was 1.0.

The first line of figure 8 gives the maximum rates of the two configurations for simple operations (such as AxB). Note that the Control Data configuration performs twice as fast in 32-bit mode. The second line shows the absolute maximum processing rate possible for each architecture, using the most complex operations available (AxB+CxD for Control Data and AxB+C for Burroughs). The next two lines are the performances estimates for the two Navier-Stokes codes provided by NASA; the first is a hybrid explicit-implicit method and the second is totally implicit. Notice that the Control Data figures are for 32-bit; it is unknown how much of the actual code could be executed with sufficient accuracy in this mode. The final two lines are estimated figures for the GISS global circulation model and the MIT spectral model, the two weather/climate codes being considered.

SUMMARY

Although feasibility studies are continuing, it has been fairly well demonstrated that the NASF can in fact be built, that it can meet Navier-Stokes performance objectives, and that it will perform efficiently on other applications as well. Subsequent phases of the Project will verify these conclusions through detailed design, actual fabrication, and subsequent operation of the Numerical Aerodynamic Simulation Facility.

REFERENCES

1. Pulliam, T. H. and Steger, J. L.: On Implicit Finite-Difference Simulations of Three Dimensional Flow. AIAA Paper 78-10, 1978.

2. Hung, C. M. and MacCormack, R. W.: Numerical Solution of Supersonic Laminar Flow Over a Three-Dimensional Compression Corner. AIAA Paper 77-694, 1977.

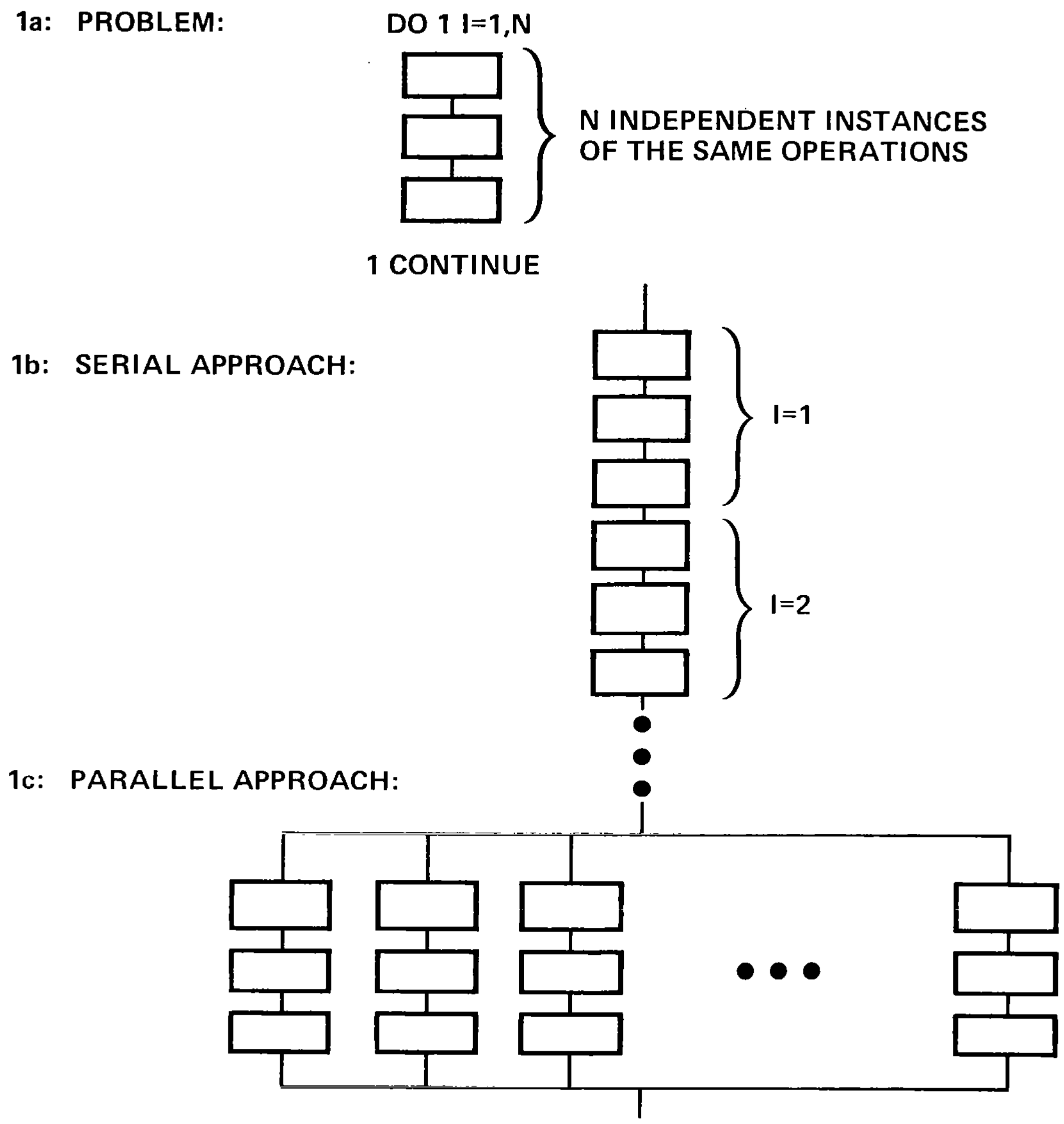
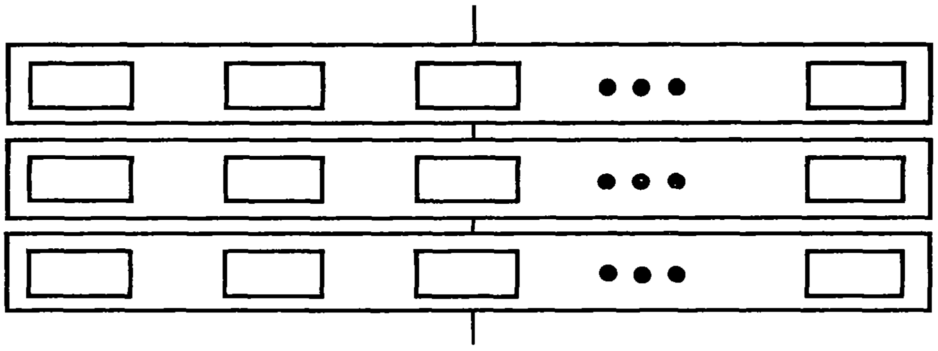


Figure 1.- Concepts of parallelism - basic.

2a: HORIZONTAL SLICING (VECTOR OR PIPELINE APPROACH)



2b: VERTICAL SLICING (MULTIPLE PROCESSOR APPROACH)

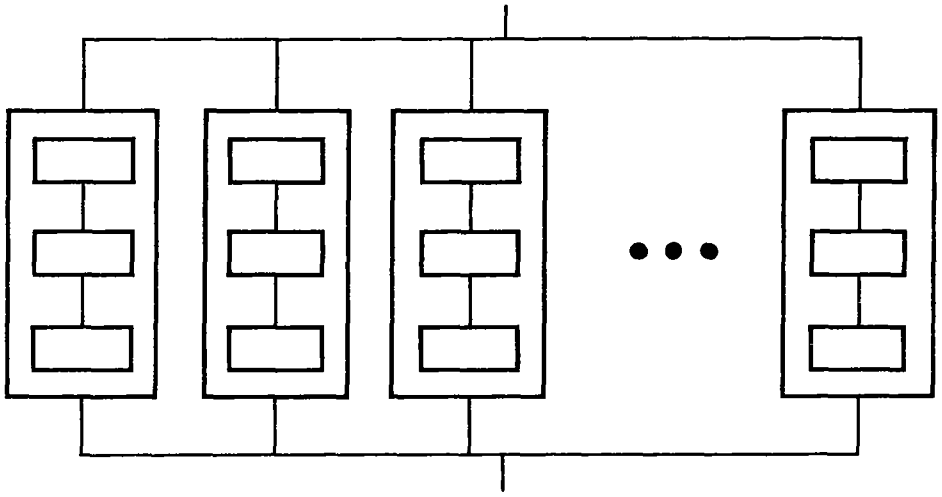


Figure 2.- Concepts of parallelism - slicing.

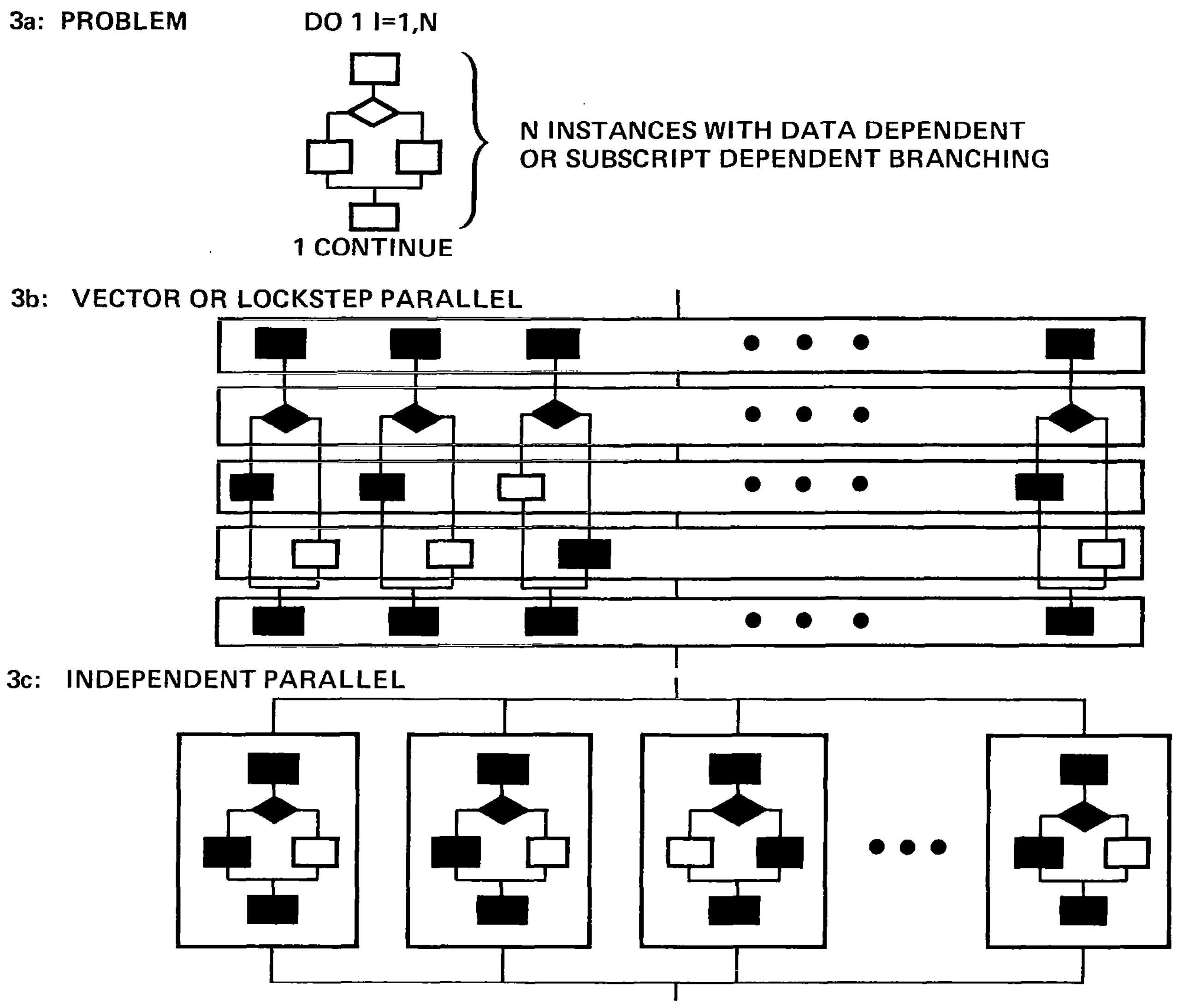

3a: PROBLEM
DO 1 I=1,N
N INSTANCES WITH DATA DEPENDENT OR SUBSCRIPT DEPENDENT BRANCHING
1 CONTINUE
3b: VECTOR OR LOCKSTEP PARALLEL
3c: INDEPENDENT PARALLEL

Figure 3.- Concepts of parallelism - branching.

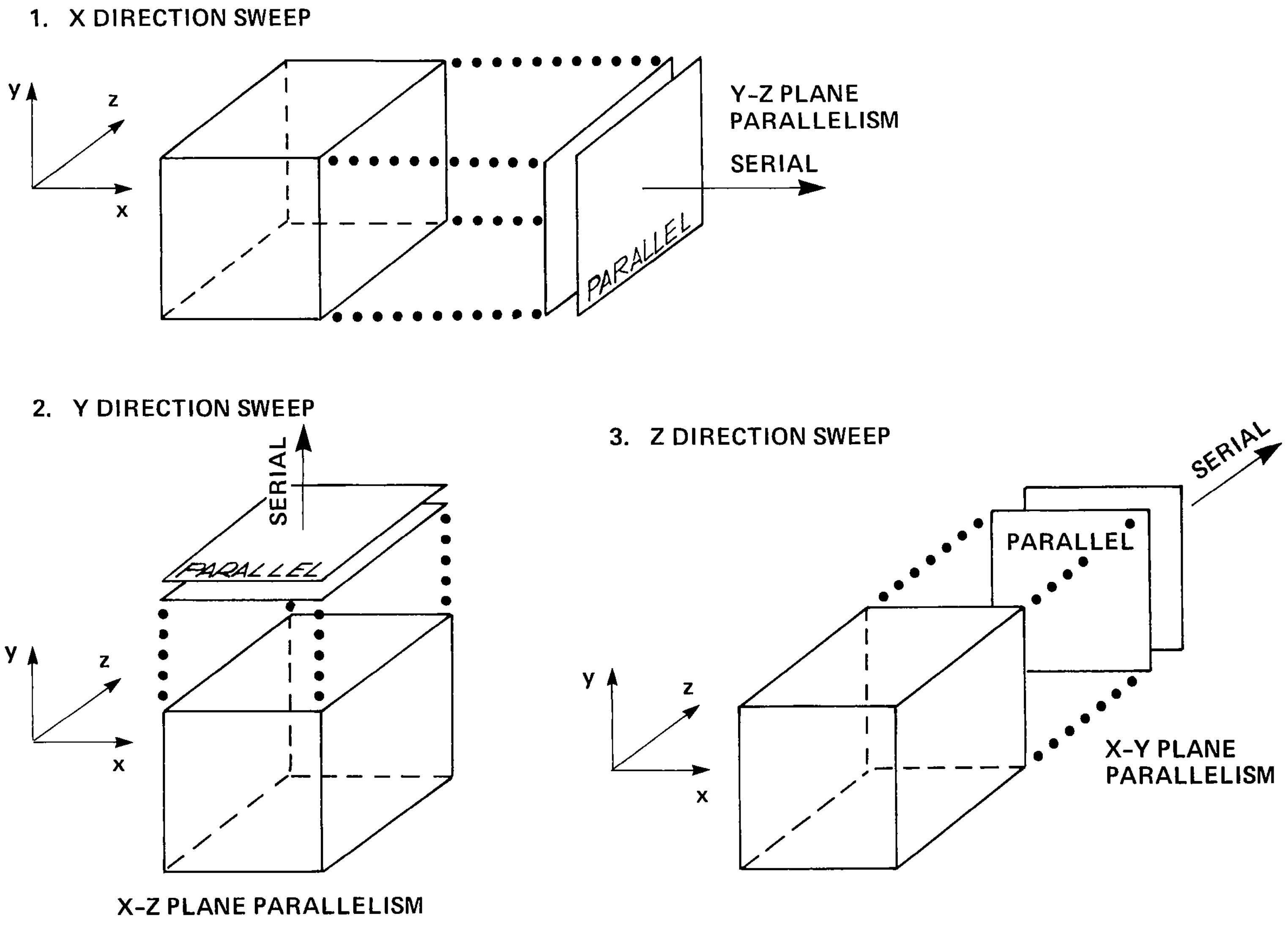


Figure 4.- The three sweeps of the three-dimensional implicit method.

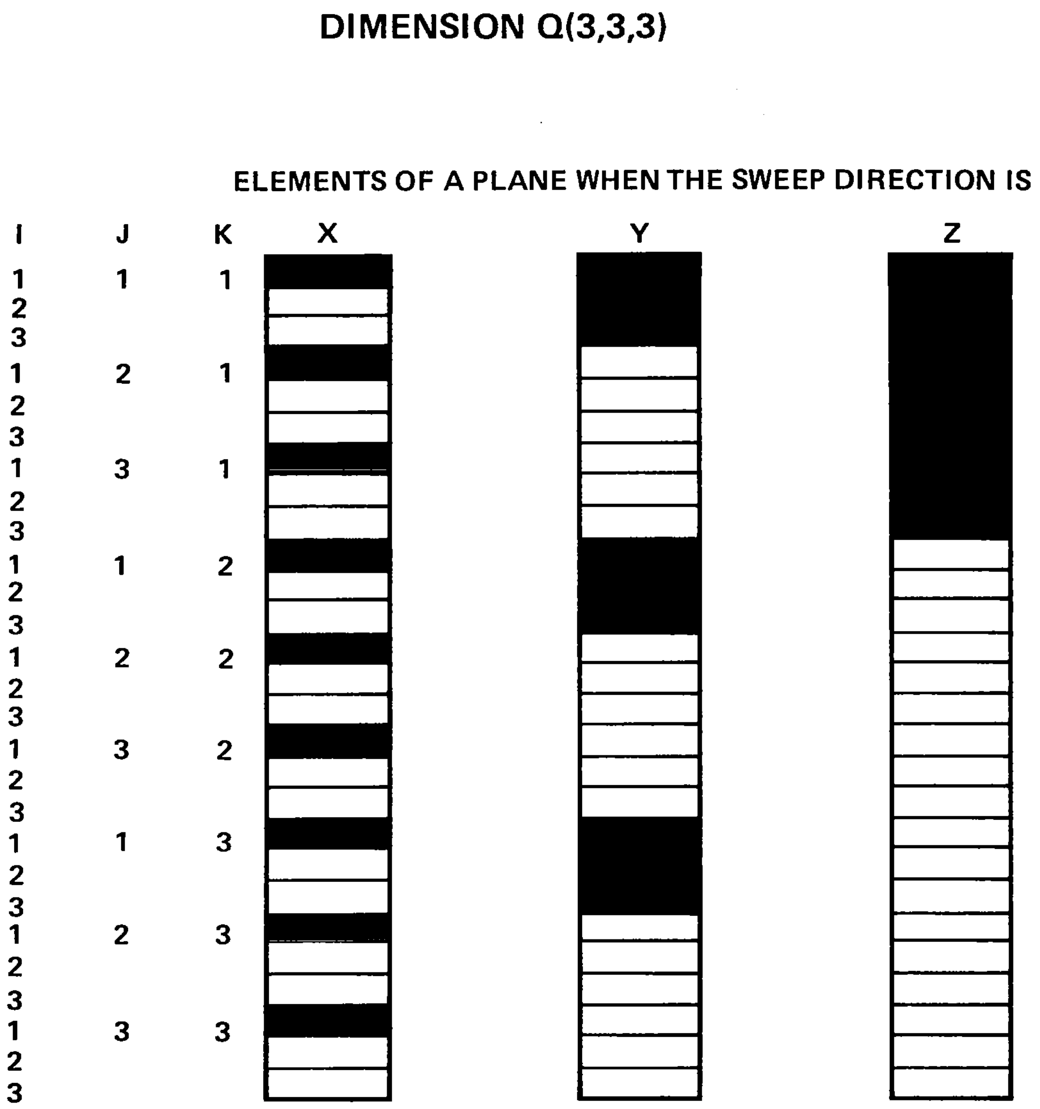


Figure 5. FORTRAN three dimensional array accessing.

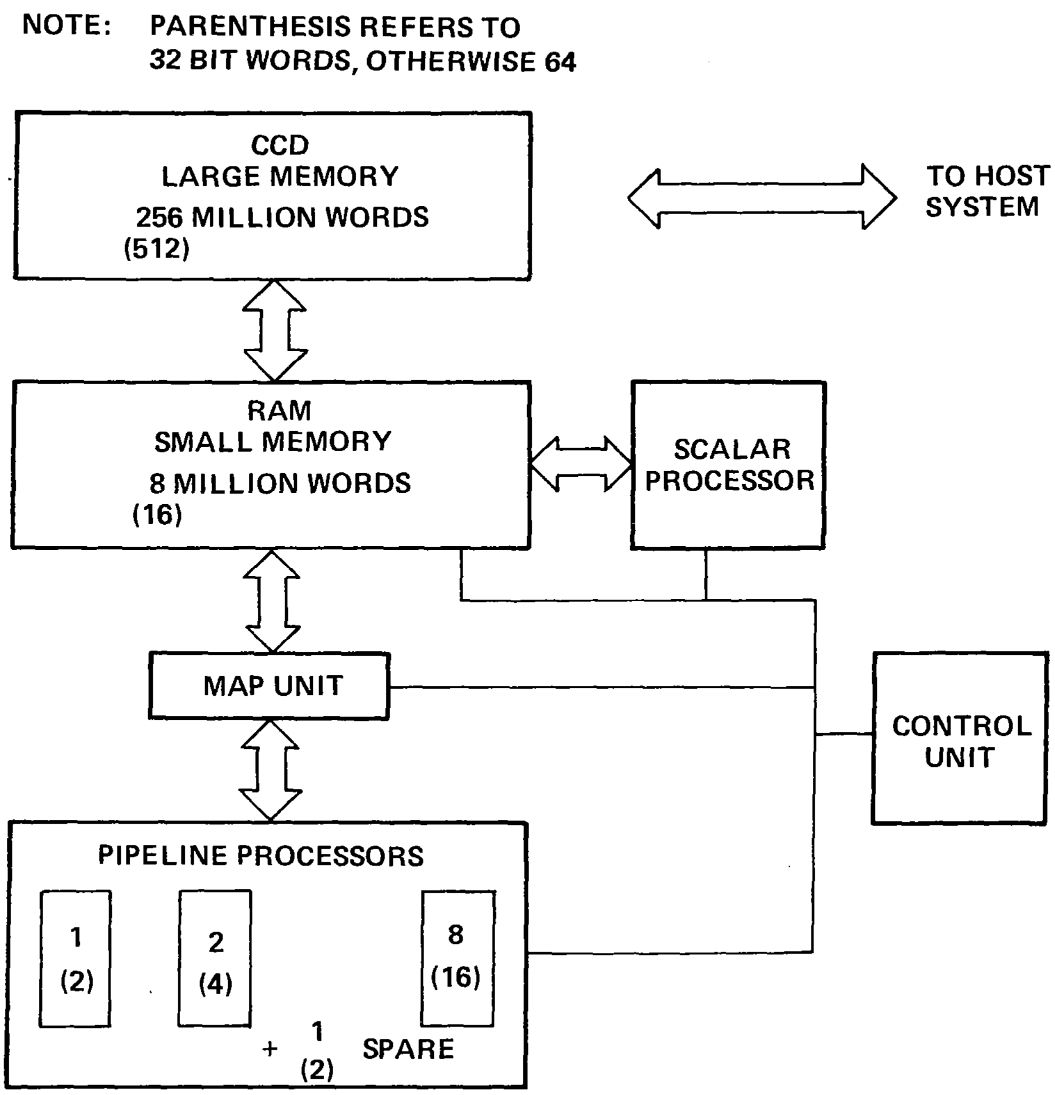


Figure 6.- Simplified block diagram of Control Data FMP. (RAM denotes random access memory.)

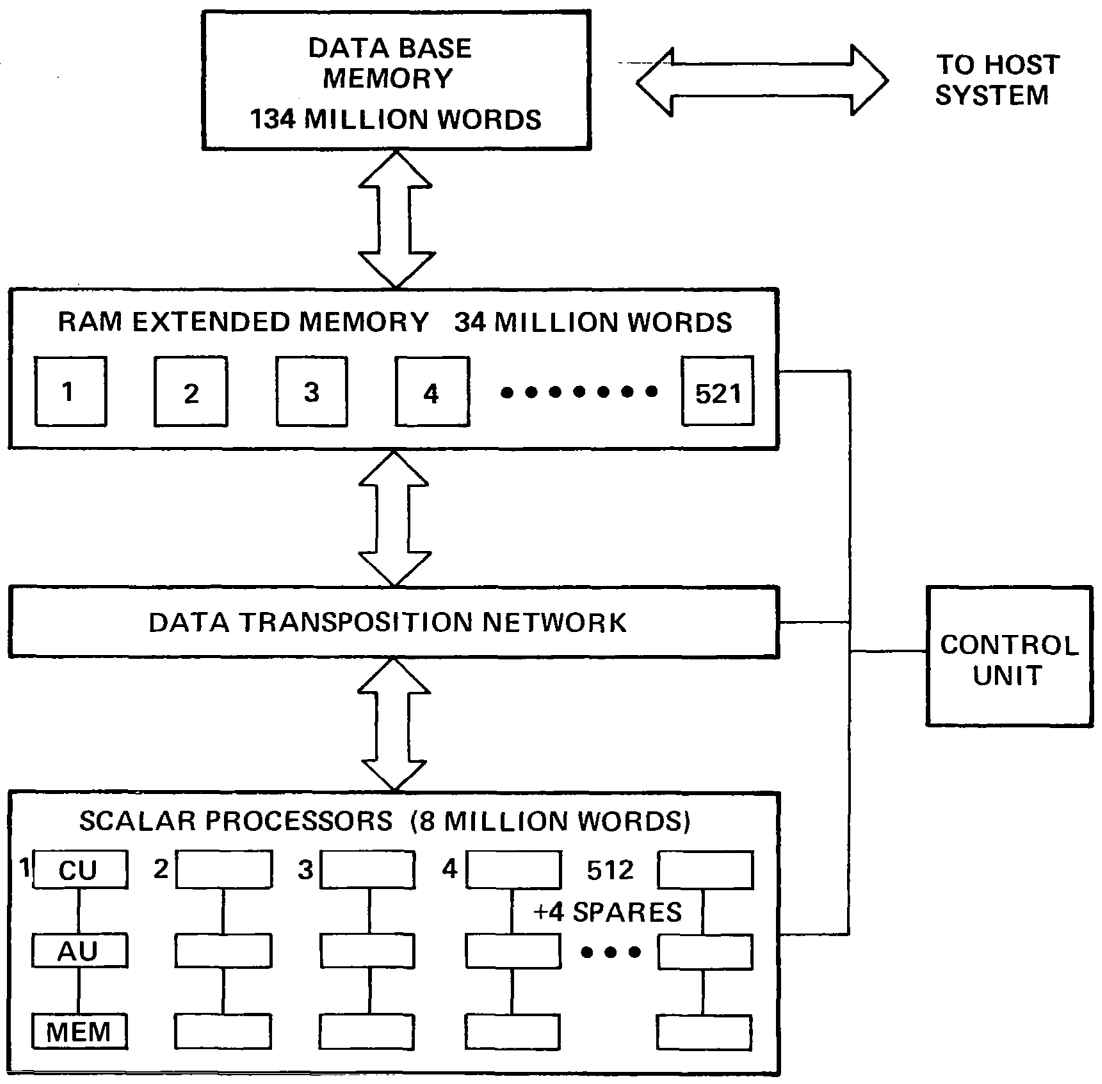

DATA BASE MEMORY
134 MILLION WORDS
TO HOST SYSTEM
RAM EXTENDED MEMORY 34 MILLION WORDS
1
2
3
4
521
DATA TRANSPOSITION NETWORK
CONTROL UNIT
SCALAR PROCESSORS (8 MILLION WORDS)
1
CU
2
3
4
512
+4 SPARES
AU
MEM

Figure 7.- Simplified block diagram of Burroughs FMP.

IN BILLIONS OF FLOATING POINT OPERATIONS PER SECOND

CODE	CONTROL DATA	BURROUGHS
MAX RATE (SIMPLE)	0.5 (AxB, 64–bit) 1.0 (AxB, 32–bit)	1.5 (AxB)
MAX RATE (COMPLEX)	1.5 (AxB+CxD, 64–bit) 3.0 (AxB+CxD, 32–bit)	2.3 (AxB+C)
HUNG-Mac CORMACK	0.9 (32–bit)	1.1
PULLIAM-STEGER	1.2 (32–bit)	1.2
GISS GCM	0.54 (32–bit)	0.8
MIT SPECTRAL	0.5 (32–bit)	0.7

Figure 8.- Performance estimates for alternate FMP concepts.

THE CURRENT STATUS OF MICROCOMPUTER ARRAYS

John C. Knight
NASA Langley Research Center

Robert G. Voigt
Institute for Computer Applications in Science and Engineering*

INTRODUCTION

In this paper we survey some current research on microcomputer arrays. There are a large number of projects currently under development, and some are at the stage of constructing hardware, while others are at the design and planning stages. We will not attempt to present an exhaustive discussion of these efforts, but rather a general overview of the topic followed by a more detailed discussion of three projects whose purposes are specifically scientific computation.

MOTIVATION

A microprocessor is the central processor of a small computer in the form of a single integrated circuit. By combining a microprocessor with a memory and a few other components, a complete computer of rather limited power can be constructed. At the present time, microprocessors typically use a word size of 8 or 16 bits, use 16-bit word addressing, and have a 1-word integer add time of a few microseconds. Often they do not provide multiply or divide instructions, and floating point operations must be explicitly programmed. The remarkable thing about microprocessors which makes them so attractive is their cost. At the time this paper was written, a microprocessor could be purchased for a few tens of dollars, and a complete microcomputer could be built for a few hundred dollars.

As the general level of semiconductor technology advances, microprocessors will become even less expensive, faster, and more sophisticated. In addition, a complete microcomputer on a single integrated circuit, a few of which are produced now, will become more widely available. Many projections from the semiconductor industry have appeared in the literature and indicate the probable availability of 32-bit microprocessors with built-in floating point hardware by the early 1980's.

*Research by the second author was performed under NASA Contract Number NAS1-14101 while he was in residence at ICASE, NASA Langley Research Center, Hampton, Virginia.

Despite the limited capabilities of a single microprocessor, it is clear that very significant computing power could be achieved by linking many of them (perhaps thousands) together in some way, and their very low cost now makes this financially feasible even though there remain technical problems.

From the point of view of the problem to be solved, there appear to be two clear motivations for considering arrays of microprocessors. For problems of moderate size, those that tax the resources of present day computers (but can be solved), the motivation is one of reducing both the cost and the time for obtaining a solution. For example, in structural analysis one routinely solves design problems that require on the order of 10 minutes of CDC 6600 computer time. However, to optimize that design, the engineer might like to solve the basic problem hundreds of times. The conflict is clear, and a compromise on the number of runs is usually made.

At the other end of the scale there are problems of significant importance that are simply too large for any present-day computers or those that will be available by 1980. For example, to model most of the details of the flow of a gas over a complex body, such as an aircraft moving through the atmosphere, would require a grid in excess of 10^6 points over which the three-dimensional Navier-Stokes equations would be discretized. It is predicted (ref. 1) that in order to model this flow in a few hours of computer time, a computer capable of performing at a sustained rate of 1 billion floating point operations per second would be required. The current super computers are capable of peak rates on the order of 100 million floating point operations per second. It is conceivable that the required raw performance could be obtained with a large array of microprocessors. Whether or not such an array could be used efficiently for scientific problems is another question with no clear answer at this point. A discussion of this problem is beyond the scope of this paper, but it may ultimately be the most difficult to solve.

PROCESSOR ORGANIZATIONS

If many microprocessors are going to be used in parallel in the solution of a large problem, then each processor will be working on a very small part of the problem at the same time. This leads to a requirement for some form of communication between the processors since in general each processor will need the results generated by others as it proceeds. For example, if many microprocessors are used to solve a discretized differential equation by Jacobi's method, they could be organized so that each processor corresponds to a node of the grid and is able to exchange data with its nearest row and column neighbors. The communications requirement is the most critical issue in the design of microprocessor networks. The expense of any completely general high performance interconnection scheme, such as direct connection of every processor to every other, becomes prohibitive as the number of processors increases. An inadequate interconnection scheme on the other hand either limits performance of the network or limits the class of problems which can be successfully tackled. A detailed discussion of interconnection schemes may be found in reference 2.

In practice, three different approaches to this problem have been used. They are

1. P processors, M memories, and a PxM electronic crossbar switch allowing any processor to access any memory.

2. P processors each with a local memory and all processors connected to a bus structure allowing transfer of data.

3. P processors each with a local memory and arranged into some form of regular lattice. Each processor is then permanently connected to a small subset of the others (usually its neighbors in the lattice).

Figures 1, 2, and 3 show examples of these organizations. Most microprocessor networks do not conform exactly to one of these but add to the basic structure based on application-specific requirements.

Within the basic structure, it is possible to arrange for all processors to be executing the same instruction at the same time in the sense of the Illiac IV, known as Single Instruction Multiple Data (SIMD), or for each processor to be working on its own program and then synchronizing with other processors as necessary, known as Multiple Instruction Multiple Data (MIMD). The choice of which scheme to use is very involved since each has its advantages and the intended application must influence the decision.

NAVIER-STOKES COMPUTER

A study by the RAND Corporation (ref. 3) has proposed a design for an array of microprocessors intended to be used specifically for the numerical solution of the Navier-Stokes equations on very large problems which cannot presently be tackled in realistic running times.

The proposed number of processors is 10 000 arranged as a 100x100 square and operating in the SIMD mode. Each processor is connected to its four nearest neighbors only. Processors on the edges are considered as neighbors of appropriate edge processors on the opposite side, forming what amounts to a "wrap-around" connection. This simple communication scheme reduces the complexity of the hardware and has been shown to be feasible for the intended application (ref. 4). The organization of the individual processors is unusual since a single integrated circuit microprocessor is not used. The intended application does not require all of the facilities of a microprocessor and so the individual processors will consist of an adder, some registers, and a small amount of memory. These may be combined in a single special integrated circuit or constructed from a small number of commercially available components. Since no sequencing capability is included, control information will be broadcast to all of the processing elements from a central control unit. If necessary, individual processors can be selected by a set of row and column lines. The extreme simplicity of the processors means that individual arithmetic operations may be quite slow, but since 10 000 will be performed in parallel, the overall performance will be substantial.

The amount of memory used is determined by the intended application. It is proposed to solve problems in three dimensions by using the array on a series of two-dimensional planes in sequence. Thus, data for one grid point from each plane have to be stored by each processor, and it is necessary to maintain several variables for each point. The memory will be large enough to ensure that no data transfers to or from peripheral equipment are necessary during problem solution.

The performance expected on the application of interest is very great. Assuming a 32-bit word, 500 nanosecond add time, and a total grid size of between 1/2 million and 10 million points, the solution time is predicted to be between two and three orders of magnitude faster than for a general purpose computer of the CDC 7600 class.

An interesting proposal in this design is the incorporation of a program accessible light-emitting diode (LED) on each processor. The purpose is to allow visual examination of certain aspects of the solution as it proceeds by arranging for each processor to switch on or off its LED based on local solution characteristics.

WISPAC

A group in the Department of Electrical and Computer Engineering at the University of Wisconsin has proposed an MIMD array for the solution of a wide range of partial differential equations (ref. 5). As with the previous design, the basic idea is to have computing power associated with each node of the discretized equations; however, the Wisconsin Parallel Array Computer (WISPAC) is considerably more general in its design. Figure 4 shows the overall structure.

WISPAC consists of a three-dimensional array of as many as 100x100x20 microcomputers, each connected to its six nearest neighbors and with edge nodes making "wrap-around" connections. The array is logically subdivided into sectors with from 5x5x5 to 25x25x20 nodes per sector. All of the nodes in a given sector are connected to a sector control computer, and all of these are connected to a master processor. A sector control computer is primarily responsible for overall program control, control of communication among microcomputers, and input/output to the outside through the master processor.

The microcomputer at each node of the array contains a full microprocessor and local data and program memory, making it possible for each processor to execute different instructions simultaneously. This also alleviates the possible bottleneck of a single controller attempting to distribute instructions to the entire array.

An interesting feature of the array design is an intra-array communication scheme in addition to the six nearest neighbor connections. Each processor can function as a switch by accepting data from any one of its six nearest neighbors and passing the data on to any other. The logic for handling this communication could be established for one particular problem by the sector control computer, or it could be changed during the solution of a problem by the individual node computers. This capability greatly increases the set of algorithms that one might expect to run efficiently on the array. Communication between non-neighboring nodes using this mechanism does represent an overhead which increases significantly if data must be moved over long distances or if a large number of nodes must communicate over short distances. In addition, for nonuniform problems, selection of efficient communication paths is a nontrivial problem. Reducing this overhead is perhaps the key to the effectiveness of the design.

DISTRIBUTED ARRAY PROCESSOR

International Computers Limited (ICL) has developed a machine called the Distributed Array Processor (DAP) (ref. 6) which is currently being marketed. The DAP consists of a large number of processors which are organized as a square array, with each processor connected to its four nearest neighbors. It is an SIMD machine and there is a control unit which broadcasts instructions to the processors in the array. Each processor contains an enable/disable bit which can be set under program control and determines whether that processor will execute the current instruction or no operation.

The processors are unusual in that they operate on single bits. A processor consists of a 1-bit adder, three 1-bit registers, and typically, a 4096 bit memory. The memories of all the processors taken together also constitute part of the memory of a conventional computer known as the host. This sharing of memory allows for very effective communication between the two systems and permits the host to operate on the data with no data transmission, for example, to set up the initial problem and to extract results. The host and the DAP may operate concurrently on separate tasks provided the host does not use the part of its memory which is shared. For example, the host may be compiling a program to be run on the DAP while the DAP is executing a different program.

As well as being able to communicate with its four nearest neighbors, each DAP processor is also connected to row and column "highways". These are data paths which allow data to be shared by all the processors in a row or column.

Since each processor can only perform single bit addition, floating point operations have to be explicitly programmed in terms of 1-bit operations. Multi-bit arithmetic can either be performed by one processor in bit serial mode or by a group of processors in parallel. In the first case, all of the bits of an operand are stored in one processor's memory, and in the second case, one bit from an operand is stored in each processor's memory. The precision of arithmetic operations and all the details of the arithmetic algorithm are therefore under program control. Thus, if a 29-bit floating point format is most appropriate for a particular problem, it can be used and will be more efficient in both speed and use of memory than a longer format. As well as flexibility in determining arithmetic, the bit level operations give the DAP considerable diversity in problem solving. For example, the maximum element of an array which has been stored with all of the bits of an element in one memory can be found by a bit level algorithm whose time is proportional to the number of bits in the number format, not the size of the array.

The manufacturer reports (ref. 7) the construction of a prototype DAP with 1024 processors organized as a 32x32 square. Each is equipped with 1024 bits of storage. The time reported for a matrix multiplication involving 32x32 matrices and using 32-bit floating point numbers was 16 milliseconds. This corresponds to approximately 4 million floating point operations per second. Inversion of such a matrix was found to take 29 milliseconds. Typical production models are expected to be arrays of 32x32, 64x64, or 128x128 processors.

PROJECTIONS

Although microcomputer arrays are in their infancy, it is possible to identify some areas where significant problems must be overcome in order to achieve any degree of success in solving real scientific problems. In this section, we will outline some of these problems and indicate some possible solutions that are currently under consideration.

Processor communication is still the major issue. No present design involving microprocessors has attempted to use the layout shown in figure 1. Some use the bus technique shown in figure 2, and most use a combination of the schemes shown in figures 2 and 3. The difficulty of connecting every processor to every other is that the complexity, and hence the cost, of the switch increases as the square of the number of processors. The 16x16 switch on the Carnegie Mellon University C.mmp computer (ref. 8) proved to be one of the most difficult problems in that design. There have been recent proposals (for example, refs. 9 and 10) which use less than full cross bar switches in a tree structure. This idea reduces complexity at the cost of reduced generality. Decisions must be made regarding the routing of information which may prove to be an expensive overhead.

At the other extreme, although the cost is minimal, the limitations of a nearest neighbor connection are clear, and it may be advantageous to augment this in some way, for example, with a global bus (ref. 11). In the end, experience with some configurations will be needed in order to determine the best approach for a given subset of problems.

Another critical issue that must be faced is that of control of the array. A control processor may be used to broadcast instructions to the array, or the computers in the array may execute their own sequence of instructions. The former approach requires that less hardware be invested at each node but reduces the algorithmic possibilities since any reasonably sized controller could support only a relatively small number of instruction streams. Again, the particular application should probably be a significant factor in the final design decision.

The question of reliability must be faced also. Although individual integrated circuits may have mean times to failure in the millions of hours, for a 100 000 node array with perhaps ten circuits per node, the failure rate may not be acceptable. Hardware will undoubtedly become more reliable, but array designers may have to supply error detection schemes which reduce the impact of a failure. This might involve switching a standby processor into the array where the failure occurs. The implication of this approach on intra-processor communication is profound.

Providing software for array computers is also a significant problem. Constructing an operating system is probably tractable. One solution would be to simply treat the array as a slave device to a host, allowing relatively routine modifications to the host operating system. The question of programming languages is more difficult. Potential users of array computers would prefer simply to run existing FORTRAN programs and to write new programs in standard FORTRAN. Experience with earlier parallel processors has shown that this usually produces very inefficient use of the array. The problem is that automatic detection and exploitation of parallelism are only marginally successful on real programs. New programming languages or modified versions of existing languages which give ready access to the unique features of the new hardware seem to be required.

The performance of a computer in the solution of a problem depends heavily on the particular algorithm being used. Algorithm development in the recent past has been dominated by the need to make effective use of serial computers. The result has been significant improvements in serial algorithm performance. Unfortunately, efficient algorithms which exploit the capabilities of array processors have received little attention. Considerable development in this area is needed if this class of machines is to reach its full potential.

REFERENCES

1. Peterson, Victor L.: Computational Aerodynamics and the Numerical Aerodynamic Simulation Facility. Proceedings of NASA Ames Research Center Workshop on Future Computer Requirements for Computational Aerodynamics, Oct. 4-6, 1977, pp. 5-30.

2. Sullivan, H.; and Bashkow, T. R.: A Large Scale Homogeneous Fully Distributed Parallel Machine. 4th Annual Symposium on Computer Architecture Proceedings, 1977, pp. 105-117.

3. Gritton, E. C.; et. al.: Feasibility of a Special Purpose Computer To Solve the Navier-Stokes Equations. Report No. R-2183-RC, RAND Corp., 1977.

4. Weiman, C.; and Grosch, C.: Parallel Processing Research in Computer Science: Relevance to the Design of a Navier-Stokes Computer. Proceedings of the 1977 International Conference on Parallel Processing, 1977, pp. 175-182.

5. Cyre, W. R.; et. al.: WISPAC: A Parallel Array Computer for Large-Scale System Simulation. Simulation, vol. 29, no. 5, Nov. 1977, pp. 165-172.

6. Parkinson, D.: Technical Description of the Distributed Array Processor. Document No. AP-2, International Computers, Ltd., 1976.

7. Flanders, P. M.; et. al.: Efficient High Speed Computing With the Distributed Array Processor. Proceedings of the Symposium on High Speed Computer and Algorithm Organization, 1977, pp. 113-128.

8. Wulf, W. A.; and Bell, C. G.: C.mmp - A Multi-Mini-Processor. Proceedings of AFIPS 1972 Fall Joint Computer Conference, 1972, pp. 765-777.

9. Lipovski, G. J.: On a Varistructured Array of Microprocessors. IEEE Trans. Comput., vol. C26, no. 2, Feb. 1977, pp. 125-138.

10. Nutt, G. J.: Memory and Bus Conflict in an Array Processor. IEEE Trans. Comput., vol. C26, no. 6, June 1977, pp. 514-521.

11. Jordan, H. F.: A Special Purpose Architecture for Finite Element Analysis. Report Number 78-9 (Contract NAS1-14101), Institute for Computer Applications in Science and Engineering, Mar. 29, 1978. (Available as NASA CR-158918.)

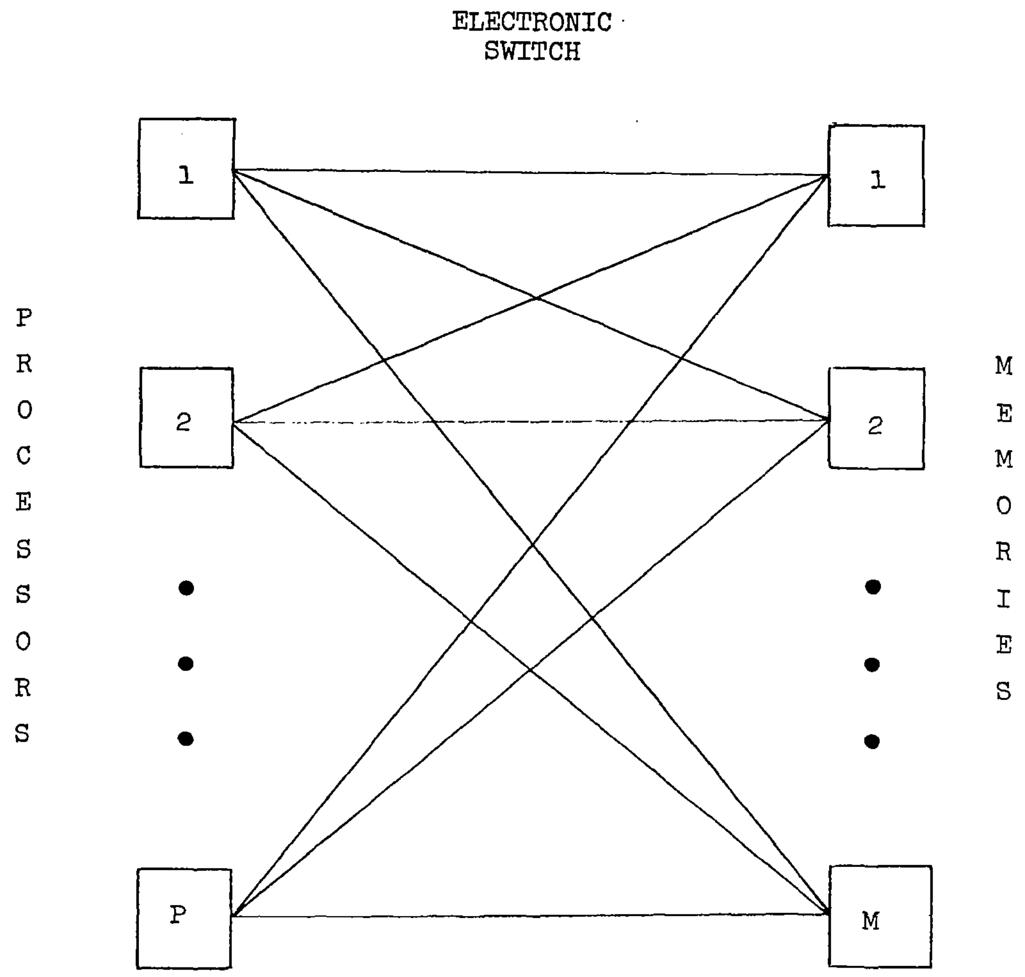


Figure 1.- General connection of processors and memories.

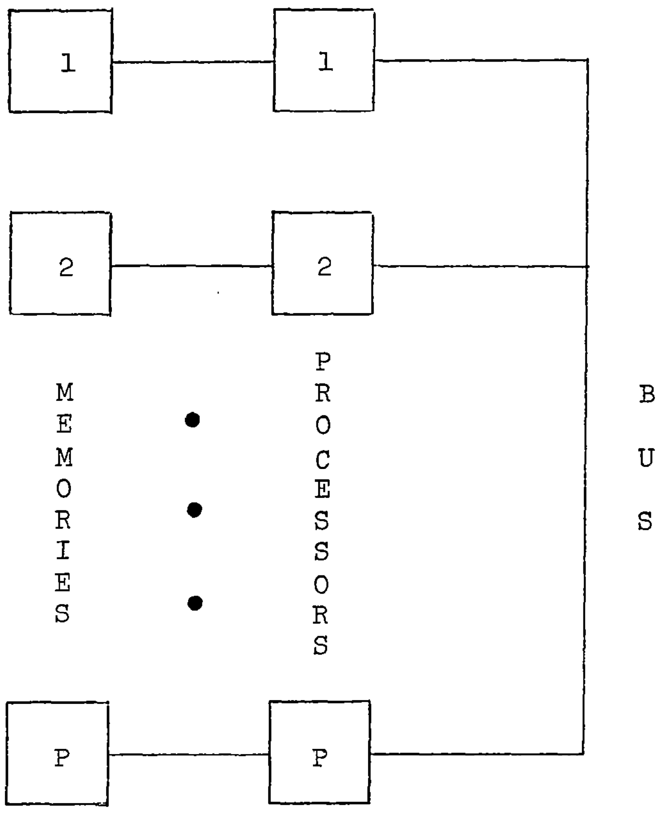


Figure 2.- Processors connected via a bus.

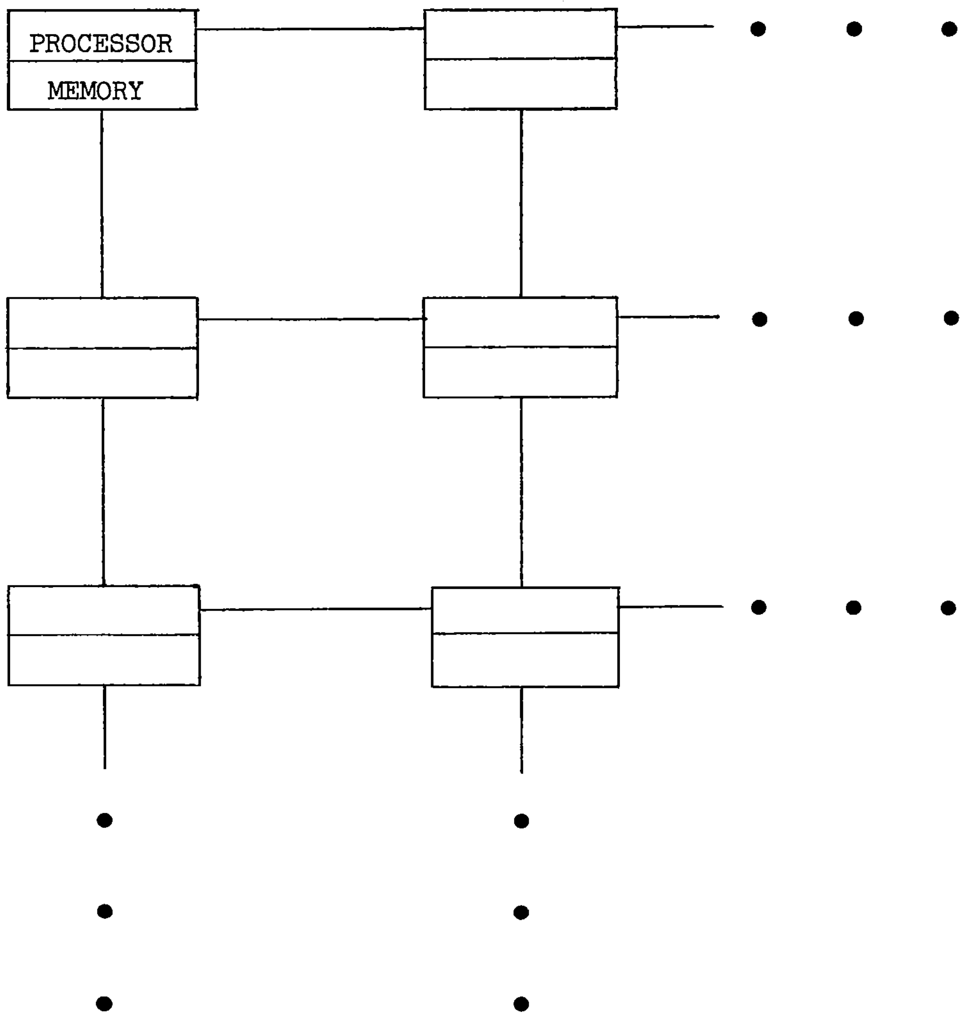


Figure 3.- Processors connected as a lattice.

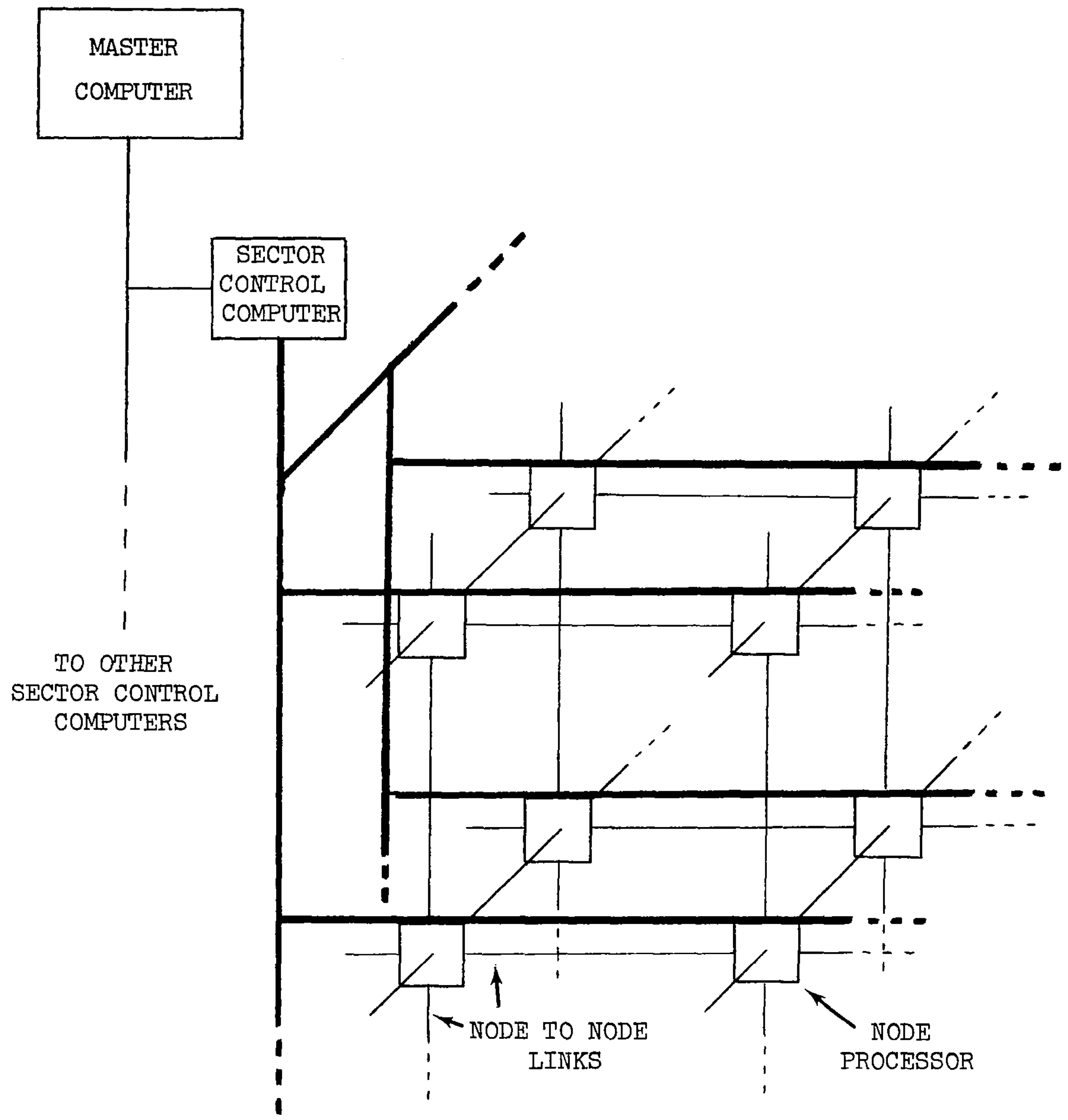


Figure 4.- WISPAC structure.

ADAPTIVE FINITE ELEMENT ANALYSIS BASED ON P-CONVERGENCE

B. A. Szabo, P. K. Basu, and M. P. Rossow
Washington University

SUMMARY

The results of numerical experiments are presented in which a posteriori estimators of error in strain energy were examined on the basis of a typical problem in linear elastic fracture mechanics. Two estimators were found to give close upper and lower bounds for the strain energy error. The potential significance of this is that the same estimators may provide a suitable basis for adaptive redistribution of the degrees of freedom in finite element models.

INTRODUCTION

One of the most important problems remaining in finite element research is the development of adaptive finite element software systems, i.e., finite element computer programs which have a local error estimation capability and a capability to increase the number of degrees of freedom selectively such that the quality of approximation is nearly uniform over the entire solution domain and the error does not exceed some pre-specified tolerance.

Research concerned with the development of adaptive finite element software systems has been underway at Washington University for several years. This work has resulted in the development of an approach for improving the quality of approximation without mesh refinement. In this approach the number of degrees of freedom is increased by increasing the polynomial orders (p) or introducing non-polynomial basis functions over fixed finite element mesh divisions. This process of reducing the error of approximation through the addition of new basis functions is called "p-convergence" to distinguish it from the conventional approach (called "h-convergence") in which the size of finite elements (h) is reduced while the number and type of basis functions for each element are fixed.

The efficiency of p-convergent adaptive procedures has been established in a series of numerical experiments, reported in references 1 and 2 and it has been shown that the rate of p-convergence cannot be slower than the rate of h-convergence (I. Babuska, private communication). In fact, for the vast majority of practical problems, p-convergence is substantially faster. This and other computational advantages suggest that adaptive finite element software systems should be based on p-convergence.

The key problem is to find suitable estimators of error which would indicate when and where the number of degrees of freedom should be increased over the solution domain. Some estimators have been proposed already:

1. Babuska and Rheinboldt developed a local, asymptotic, a posteriori error estimator for h-convergent approximations (ref. 3). The estimator requires element level computations only and measures the error in strain energy associated with an element. An optimal distribution of the degrees of freedom is obtained when this error measure is the same for all elements.

2. Melosh and Marcal (ref. 4) proposed to measure the specific energy difference (SED), defined as the largest difference over the element domain between the computed strain energy density function and the same function evaluated at the origin of the elemental coordinate system. This measures the effect of modes higher than those associated with the (generalized) constant strain states on the distribution of strain energy density within finite elements. The criterion for mesh refinement is that SED be approximately the same for each finite element. In practical computations SED is approximated by the largest of the strain energy density differences evaluated at quadrature points only.

3. Peano et al. (ref. 5) proposed a criterion for p-convergent approximations. This criterion is based on the rate of change of the total potential energy with respect to higher order displacement modes, evaluated <u>before</u> the higher displacement modes are actually introduced. When the rate of change of the potential energy exceeds a prescribed tolerance, the stiffness terms corresponding to the higher modes are assembled and the new system of equations is solved. The hierarchic structure of the elemental stiffness matrices permits efficient use of block relaxation procedures in obtaining improved solutions.

In this paper we examine the numerical characteristics of a criterion similar to that proposed by Babuska and Rheinboldt in ref. 3, but modified for p-convergent approximations. Our study is preliminary in nature and is restricted to one specific problem.

ERROR MEASURES

We have examined two measures for the error in strain energy on the basis of a problem in two-dimensional elasticity containing a geometric singularity. This problem is typical for a large class of problems in linear elastic fracture mechanics (fig. 1). The error measures were as follows. First we define the ith component of the residual vector, which represents the unbalanced body force, as

$$r_i = G\,\hat{u}_{i,jj} + (\lambda + G)\,\hat{u}_{j,ji} + X_i \qquad (1)$$

in which

G and λ	are Lame's constants;
$\hat{u}_i$	is the ith component of the displacement vector computed by the finite element method. The subscripts range over 1,2.
X_i	is the ith component of the body force vector.

One of the measures, to be called the "r-estimator," is defined for the kth finite element as

$$R_k(\alpha) = \frac{1}{p_k^{\alpha}} \int_{A_k} (r_1^2 + r_2^2)\, dA \qquad (2)$$

in which

α	is a constant, to be determined by numerical experiments;
p_k	is the polynomial order of the displacement approximation over the kth element;
A_k	is the area of the kth element.

The other measure, to be called the "t-estimator", is defined over interelement boundaries and external boundaries on which tractions are specified, as the square of the unbalanced tractions. Specifically, at the boundary of two elements, the vector of unbalanced tractions is:

$$t_i(s) = [\hat{\sigma}_{ij}^{(a)}(s) - \hat{\sigma}_{ij}^{(b)}(s)]\, n_j \qquad (3)$$

in which

$\hat{\sigma}_{ij}^{(a)}$	is the finite element approximation to the stress tensor for the ath element;
n_j	is the unit normal to the interelement boundary;
s	is the variable along the element boundary.

At external boundaries the unbalanced traction vector is the difference between the computed traction vector and the applied traction vector. When displacement vector components are specified, the corresponding unbalanced traction vector component is zero.

The t-estimator is defined as

$$T_k(\beta) = \frac{1}{p_k^{\beta}} \int_{\Gamma_k} (t_1^2 + t_2^2)\, ds \qquad (4)$$

in which

β	is a constant, to be determined by numerical experiments;
p_k	is the polynomial order of the displacement approximation at the kth interelement or external boundary segment;
Γ_k	denotes the kth elemental boundary.

Both the r and t measures were found to give close indications of the total error in strain energy in p-convergent approximations. The details are as follows.

The sample problem represented in fig. 1 does not have a known exact solution. For this reason it was necessary first to estimate the exact value of the total strain energy U. This was possible by utilizing the asymptotic relationship given in reference 6:

$$U = U_p + \frac{c}{NDF} \tag{5}$$

in which

U_p	is the computed (total) strain energy, based on pth order polynomial approximation;
c	is a constant;
NDF	is the (net) number of degrees of freedom.

Extrapolating on the basis of eq. (5), U was found to be 0.7702 $\sigma^2\ell^2/E$ (+ 0.0002 $\sigma^2\ell^2/E$) in computations involving two different finite element mesh divisions. The computed dimensionless strain energy values, the corresponding errors and the values of the two error estimators for parameter values $\alpha = 3$, $\beta = 2$ are given in Table I. The percent changes in the estimators as p is increased tend to bound the corresponding percent change in the strain energy error with increasing precision such that the change in the r-estimator is smaller and the change in the t-estimator is larger than that of the strain energy error. Furthermore, the values of these percentage changes are monotonically decreasing for $p > 2$. This suggests the possibility that two constants c_r and c_t could be found such that a relationship,

$$c_r \sum_k R_k(3) \leq U - U_p \leq c_t \sum_k T_k(2) \tag{6}$$

remains valid for all p-values and that the upper and lower bounds become progressively closer to the strain energy error as p is increased. This is, of course, highly speculative at the present because no theoretical justification exists, but consistent with the observations of this numerical experiment. For example, if we choose $c_r = 0.7008$ and $c_t = 0.2657$, the two estimators will give the value of the strain energy error at $p = 7$. The resulting relationship between the strain energy error and the two estimators is illustrated in fig. 2.

The question naturally arises whether the same estimators would bound the energy error at the element level as well. Clearly, this approach will be useful only if the indicators tend to zero with increasing p at about the same rate as the error in strain energy does, not only for the entire solution domain but for individual finite elements as well. The presently available information is sufficient only to indicate trends.

The problem chosen for study does not have a known exact solution; thus the energy error cannot be computed with precision. For the global solution the rate of convergence formula (eq. 5) provided a basis for extrapolation to the limit value of the computed strain energy values. Such formulas are not available for predicting convergence at the element level. For this reason an ad hoc procedure for extrapolation had to be devised. It was assumed that the strain energy converges at the element level as

$$u^{(k)} = u_p^{(k)} + \frac{\mu}{p^\nu} \tag{7}$$

where μ and ν are constants, $u^{(k)}$ is the strain energy of the kth element, and $u_p^{(k)}$ is the strain energy of the kth element computed on the basis of uniform pth order polynomial approximation over all of the finite elements. Taking two consequtive values of polynomial orders q and p, μ can be eliminated and the extrapolated value of $u^{(k)}$ is

$$u^{(k)} = \frac{p^\nu u_p^{(k)} - q^\nu u_q^{(k)}}{p^\nu - q^\nu} \tag{8}$$

where ν and $u^{(k)}$ were chosen such that for $q = 5$, $p = 6$ and $q = 6$, $p = 7$ the value of $u^{(k)}$ was constant. The resulting estimate of the strain energy for element 2 is $u^{(k)} = 0.0844\ \sigma^2\ell^2/E$. When the p-distribution is uniform over the entire mesh, the estimators vary over the finite elements by several orders of magnitude. As could be expected, their value is the greatest for the crack tip elements (element numbers 1,3,4) and least for the elements remote from the crack tip (element numbers 5,6,8). In those elements which are not on the crack tip, the estimators apparently approach zero faster than the error in strain energy.

When the distribution of p is altered such that the element to element variation in the estimators is reduced, then the estimators tend to become closer to the energy error. Specifically, letting $p = 7$ for the crack tip elements and $p = 3$ for the remote elements, the variation in the estimators is reduced somewhat but is still 5 orders of magnitude for the r-estimators and 4 orders of magnitude for the t-estimators. In this case we have complete third order polynomial approximation in the transition elements 2 and 7 with some additional shape functions ranging in order from 4 to 7. The estimators for element 2 were computed as $c_r R_2(3) = 0.2 \times 10^{-5}$, $c_t T_2(2) = 0.2 \times 10^{-4}$, and the dimensionless strain energy error as 0.4×10^{-4}. Thus the indications are that the bounding property of the estimators may be preserved at the element level only if the p-distribution is such that the estimators do not vary by more than 2 or 3 orders of magnitude.

In all computational experiments conducted to date the r- and t-estimators were found to be consistent indicators of the source of energy error, in this case the crack tip singularity.

CONCLUDING REMARKS

A preliminary numerical investigation has indicated that readily computable bounds may exist for the error in strain energy in p-convergent finite element analysis. These bounds would provide an indication of where the degrees of freedom should be increased over the solution domain in adaptive finite element analysis to achieve uniform quality of approximation.

REFERENCES

1. Chen, K. C., "High-Precision Finite Elements for Plane Elastic Problems", Doctoral Dissertation, Washington University, St. Louis, Missouri, 1972.

2. Mehta, A. K., "P-Convergent Finite Element Approximations in Linear Elastic Fracture Mechanics", Doctoral Dissertation, Washington University, 1978.

3. Babuska, I. and Rheinboldt, W. C., "A-Posteriori Error Estimates for the Finite Element Method", Technical Report TR-581, University of Maryland, September, 1977.

4. Melosh, R. J. and Marcal, R. V., "An Energy Basis for Mesh Refinement of Solid Continua", Int. J. Num. Meth. Engng., Vol. 11, pp. 1083-1091, 1977.

5. Peano, A. G., Fanelli, M., Riccioni, R., and Sardella, R., "Self-Adaptive Convergence at the Crack Tip of a Dam Buttress", Proc. Int. Conf. on Num. Meth. in Fracture Mechanics, Swansea, U.K., January 9-13, 1978.

6. Szabo, B. A. and Mehta, A. K., "P-Convergent Finite Element Approximations in Fracture Mechanics", Int. J. Num. Meth. Engng., Vol. 12, pp. 551-560 (1978).

TABLE I
ESTIMATORS FOR THE ERROR IN STRAIN ENERGY

[Centrally cracked panel, 8-element mesh, uniform p-distribution.*]

p	U_p[†]	$U-U_p$	$\sum_k R_k(3)$	$\sum_k T_k(2)$
2	0.6936	0.0766	0.0864	0.4662
3	0.7305	0.0397	0.0534	0.1989
4	0.7461	0.0241	0.0321	0.1042
5	0.7538	0.0164	0.0218	0.0658
6	0.7584	0.0118	0.0161	0.0452
7	0.7613	0.0089	0.0127	0.0335

*To convert entries into dimensioned values, multiply by $\sigma^2\ell^2/E$.
†Poisson's ratio: 0.3.

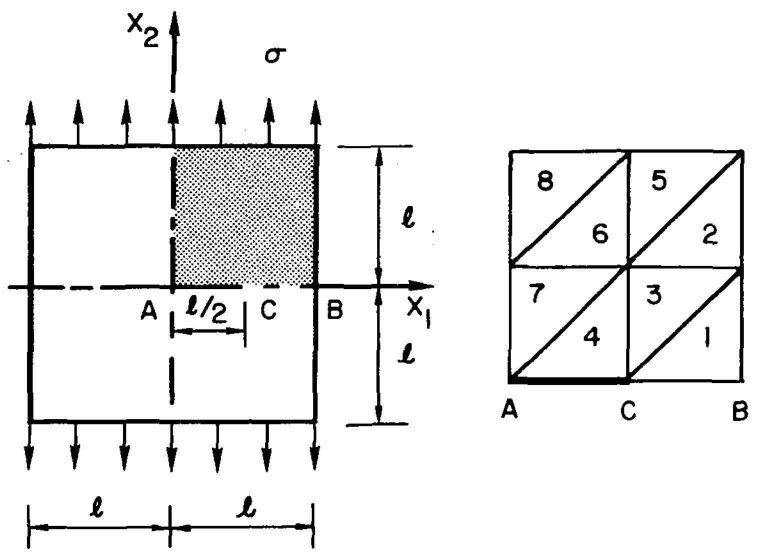


Fig. 1. Centrally cracked square panel mesh division and element numbering.

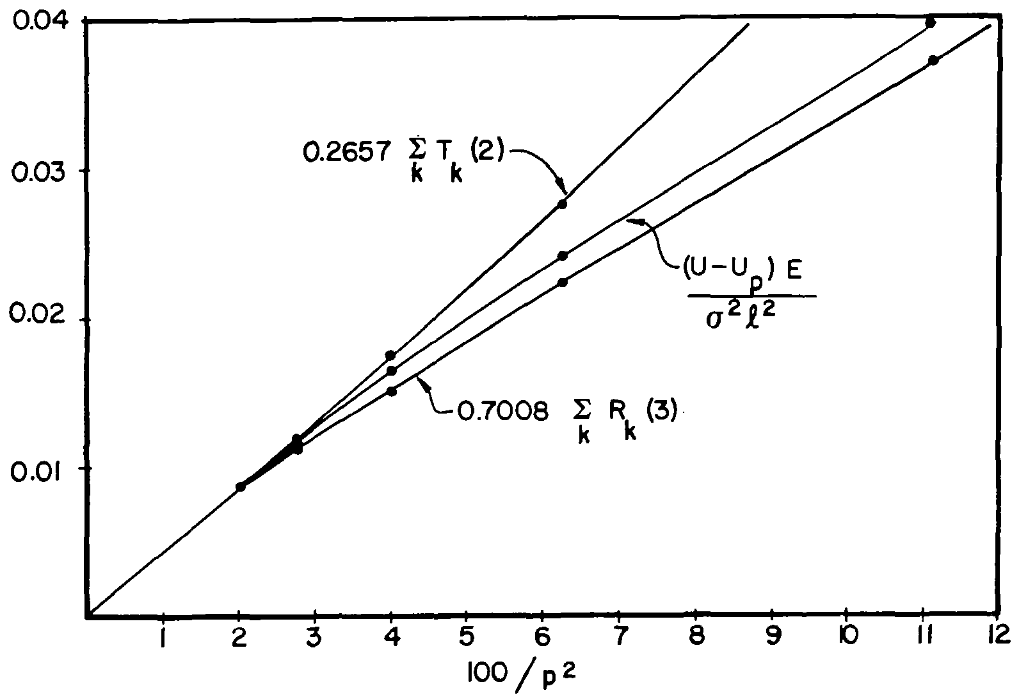


Fig. 2. Variation of the error indicators and the error in strain energy with the polynomial order (p).

FINITE ELEMENT ANALYSIS OF HELICOPTER STRUCTURES

Melvin J. Rich
Sikorsky Aircraft

SUMMARY

This paper presents the progress at Sikorsky Aircraft of the applications of finite element (F.E.) analysis for helicopter structures. The finite element analysis is now the standard method for helicopter airframe structures, and the use is now being expanded for 3D analysis of mechanical components. Examples of application are presented for airframe, mechanical components, and composite structure. Data are presented on the increase of model size, computer usage, and the effect on reducing stress analysis costs. Future applications for the use of finite element analysis for helicopter structures are projected.

INTRODUCTION

Prior to 1971 the major method for analysis of helicopter structures at Sikorsky Aircraft was the usual strength-of-materials approach. Semiempirical corrections were made to account for complex cutouts or stress concentration regions. Elastic energy methods were employed to a limited degree for some redundant structural areas, but mainly as a stress check for highly stressed parts. In the 1960's, an airframe was extensively strain gaged (about a thousand gages were used) to correlate stress analysis with test results. This correlation study showed that an appreciable weight reduction could be achieved if a more accurate analysis method was used to predict internal load paths. As a result a force method was used to reanalyze the airframe type structures, and appreciably improved correlation was obtained. However, the principal problem was the inability to use such improved methods to effect the structural design in a timely manner.

A review in 1970 indicated that the finite element technology had progressed to a level that this methodology could be employed as a design tool for helicopter airframes. The most promising features were that, with moderate instruction, the design stress engineer would be able to employ finite element analysis, and the more accurate analysis could be employed in the design iteration schedule. The expected results would be reduction in structural weight, with less risk of major design changes arising from not meeting final test verification. Table I illustrates the major milestones in the use of finite element technology at Sikorsky Aircraft.

A survey was made as to which of the many well developed finite element programs to use. NASTRAN® was selected since it met the selecting criteria (i.e., user oriented, analysis features, and, most important, the projected continued support for updating and development). The next step was to check out the use of NASTRAN to predict stresses and deflections. A static test airframe of the CH-53A helicopter was used and the test correlation was found to be highly superior to previous stress methods used. As a result, the confidence and experience was gained to employ the finite element technology to the subsequent airframes listed in Table I.

The first full finite element analysis was for the BLACK HAWK prototype airframe. Structural weight was reduced about ten percent and the accuracy was confirmed by subsequent airframe static tests. The same finite element technology was employed for the RSRA (rotor systems research aircraft), CH-53E (super stallion), and the S-76 commercial airframes.

The finite element analysis is now being employed for helicopter mechanical components. However, most of the mechanical components require a 3D finite element analysis to account for rapid changes in geometry as well as for accurate stress predictions. The availability of mesh generators has made 3D finite element analysis practical, and correlation of stress predictions has been within 10 percent accuracy for highly stressed regions.

This paper will present the progress of finite element technology, applications, benefits derived, and projection of future applications.

APPLICATIONS

The applications presented represent some of the typical structures where finite element analysis is important for helicopters.

As previously mentioned, finite element analysis was first started with the airframe structures. The BLACK HAWK helicopter shown in Figure 1 is typical of the complexity of helicopter airframes.

The airframe has large openings for troop entry on both sides, numerous openings for windows and the main rotor transmission support. Thus the center cabin has many structural discontinuities, for which the usual strength-of-materials approach would be questionable to predict accurate internal load distributions. In addition to the many usual flight and ground load conditions, the BLACK HAWK airframe was designed to meet the Army requirements for crashworthiness.

The finite element model of the BLACK HAWK is shown in Figure 2. The entire airframe, the vertical and horizontal tail surfaces, and the effect of the main rotor transmission were included in the finite element analysis with a resultant 9000 degrees of freedom. The use of finite element technology contributed significantly to a highly efficient airframe.

Another important area of helicopter applications is the design of mechanical components. These are usually of complex shape and designed for cyclic loadings. Fatigue testing is usually required to verify the component design. The goal is to "must pass" the fatigue requirements at the first test evaluation. Failure to do so involves costly changes and significant problems in schedule. Thus finite element technology, with its inherent improvement in stress prediction, is well warranted for use with helicopter mechanical components. Due to the complexity of shapes, a 3D analysis is required to achieve the accuracies desired. The 3D model shown in Figure 3 represents a segment of the CH-53A/D helicopter swashplate. By using cyclic symmetry, only the segment shown was required for the analysis, but even this segment represents 5500 degrees of freedom (equivalent to the center cabin required of the BLACK HAWK). The 3D finite element analysis provided stress results within ten percent of static test data in the highly stressed regions.

Another area of application is with the use of advanced composite materials. Sikorsky Aircraft is rapidly utilizing these advanced materials for primary structures. Since the composites are essentially linear elastic in the fiber dominated orientations, little or no plastic relief is obtained at stress concentration regions. The example shown in Figure 4 represents the finite element model of a composite bolted joint. With metal joints, the usual assumption is that bolt loads will redistribute due to local yielding. However, for composite materials, it is essential to account for accurate load distribution and concentrated stresses at the bolt regions. The finite element model shown in Figure 4 is made very fine at the bolt locations to obtain accurate local loadings for a subsequent laminate stress analysis.

ASSESSMENT OF FINITE ELEMENT ANALYSIS

The accuracy of the finite element methods as used at Sikorsky Aircraft has been well proven with the correlations obtained for airframe and mechanical components. Developments will include reduced computer time, use of improved elements, and reduced schedule time with interactive modeling.

The increasing size of the model is beginning to become a problem. As shown in Figure 5, the general number of degrees of freedom (D.O.F.) was maintained at the 5000 number level for a long period. As more of the structure is analyzed, the size grows rapidly. A 9000 D.O.F. model has been used for the production version of the BLACK HAWK fuselage. Substructuring will help in reducing some of the work load. However, 3D analysis, even on the swashplate segment, has already reached the 5000 D.O.F. level and can be expected to grow rapidly, perhaps to levels of 20,000 in the next few years. Improvements in interactive modeling, substructuring, and others will be required to handle models of this size.

Computer time is now becoming an appreciable cost factor. Figure 6 illustrates the rapid rise in total CPU time in the past five years. CPU time can be expected to rise even more rapidly in the next decade. It would appear that the finite element work load is increasing to a level which requires fully dedicated, higher speed computers.

The question arises as to what benefits the finite element technology has brought forth. First of all the increased accuracy is estimated to have resulted in a 5 to 10 percent reduction in the weight of airframe structures. We can expect even greater reductions in the weight of helicopter mechanical components when the technology is employed to the same degree as for airframe structures.

Most important is the effect on the engineering costs to analyze structures. A best estimate of the savings in engineering hours is illustrated in Figure 7. One measure is the number of hours for stress analysis per square foot of structure. In 1977 it took only one sixth the stress hours per square foot used in 1962 for airframe structures. Thus, the engineering cost effectiveness appears to be more than ample to trade off against increased computer charges.

FUTURE APPLICATIONS OF FINITE ELEMENT ANALYSIS

We can expect increased usage of finite element analysis for weight reduction, reducing engineering costs for analysis of complex structures, and the probable reduction of development costs for helicopter components. In addition, the complexity of 3D anisotropic composite structures will require the use of finite element methods because of the inability to accurately analyze by simpler strength-of-materials methods.

The damage tolerant aspect of structural design will require the use of fracture mechanics and will be integrated with finite element methods. It will be necessary to have such a combination to enable rapid design iteration and arrive at design solutions rather than completely relying on the static or fatigue test evaluation to make design changes.

Another area of application will be to permit rapid design iteration using interactive modeling. The goal would be to permit on-the-spot stress analysis and design changes to arrive at an optimum design. This goal is a long range objective and may be reached as the finite technology is improved and the capability of the computer is increased.

CONCLUDING REMARKS

Finite element analysis is now established as the standard method for helicopter airframe analysis and is now required by the military services. Figure 8 illustrates the many Sikorsky airframe applications.

It is expected that the 3D finite element analysis will soon be a standard requirement for mechanical components, and the technology is progressing in this area.

Finite element analysis will be a requirement for primary advanced composite components because of the need to account for the complex layups and anisotropic behavior of these materials. The increasing model size from fuller use of finite element analysis and the 3D applications will put further requirements on efficient modeling and increased computer capabilities.

We can also expect further finite element applications to be required for damage tolerant design and optimization of the helicopter structures.

TABLE I - FINITE ELEMENT MILESTONES SHOW INCREASING APPLICATIONS

. Survey Selected NASTRAN	1971
. Test Correlation on Airframe	1972
. First Full Application, BLACK HAWK Prototype	1973
. RSRA Airframe	1974
. First Application to Mechanical Components	1974
. CH-53E (Super Stallion) Airframe	1975
. Canopy and Airframe	1976
. Commercial S-76 Airframe	1976
. First 3D Analysis and Correlation	1976
. Production BLACK HAWK Airframe	1977

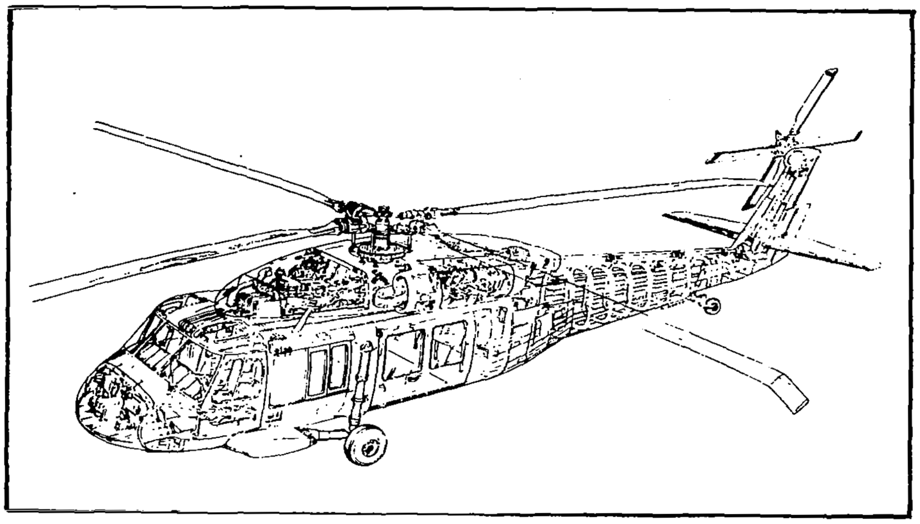


Figure 1.- Finite element technology provided highly efficient BLACK HAWK airframe.

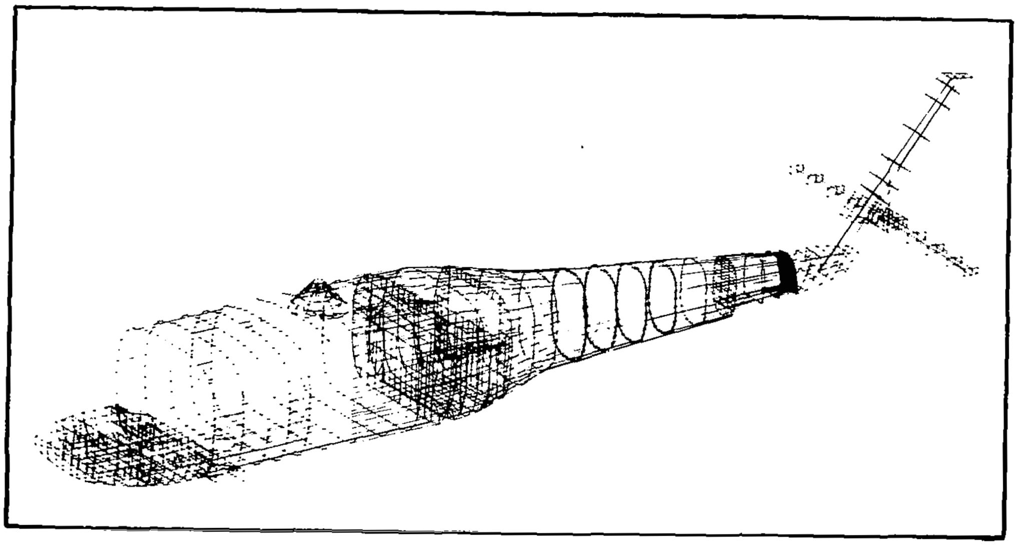


Figure 2.- Finite element model of BLACK HAWK.

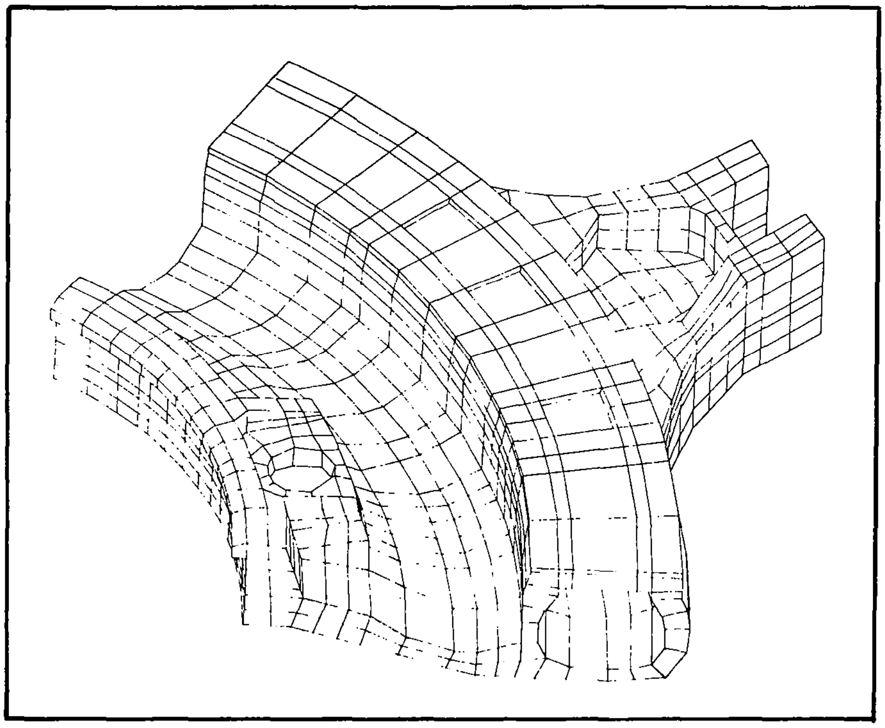


Figure 3.- NASTRAN 3D model CH-53A/D rotating swashplate.

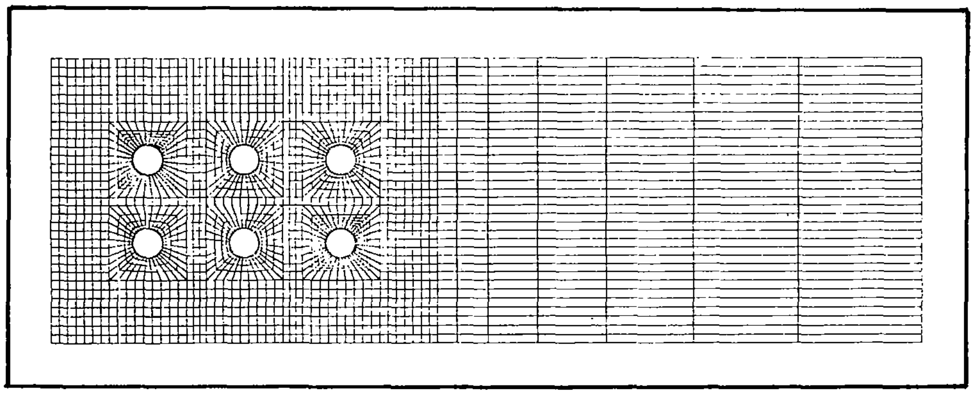


Figure 4.- Finite element analysis of composite materials bolted joint.

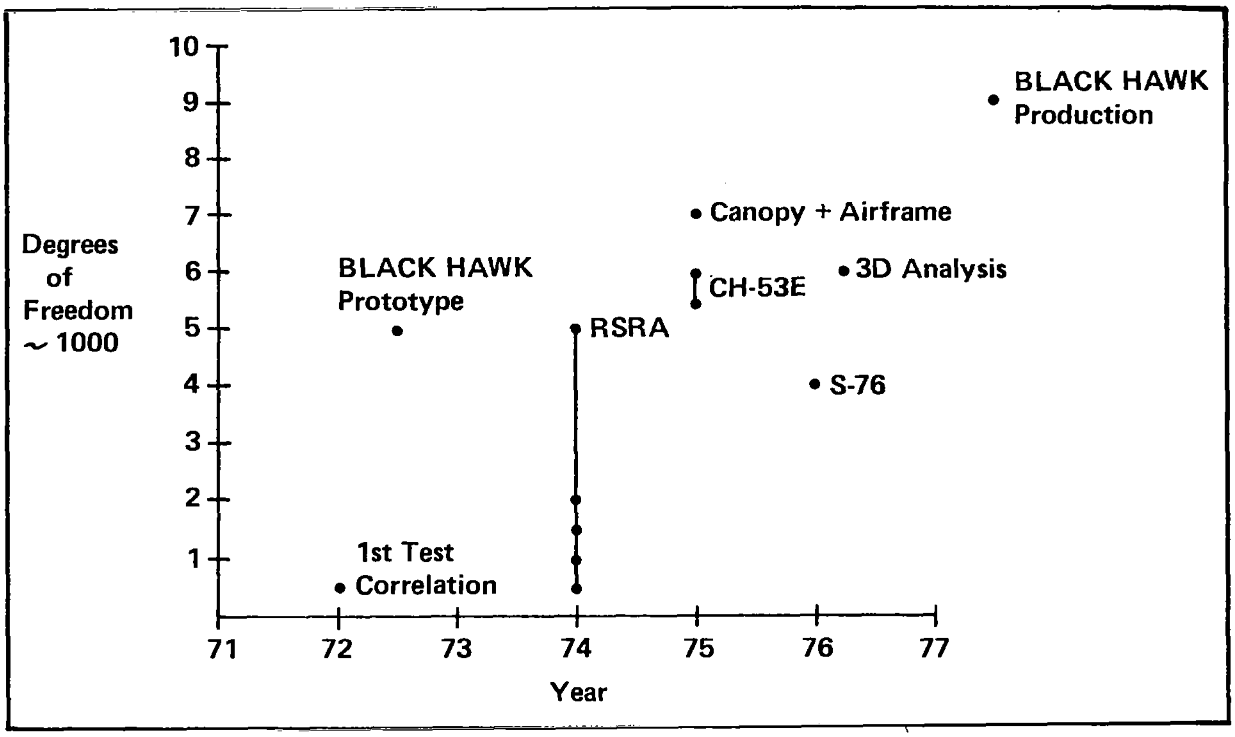


Figure 5.- Model size is increasing.

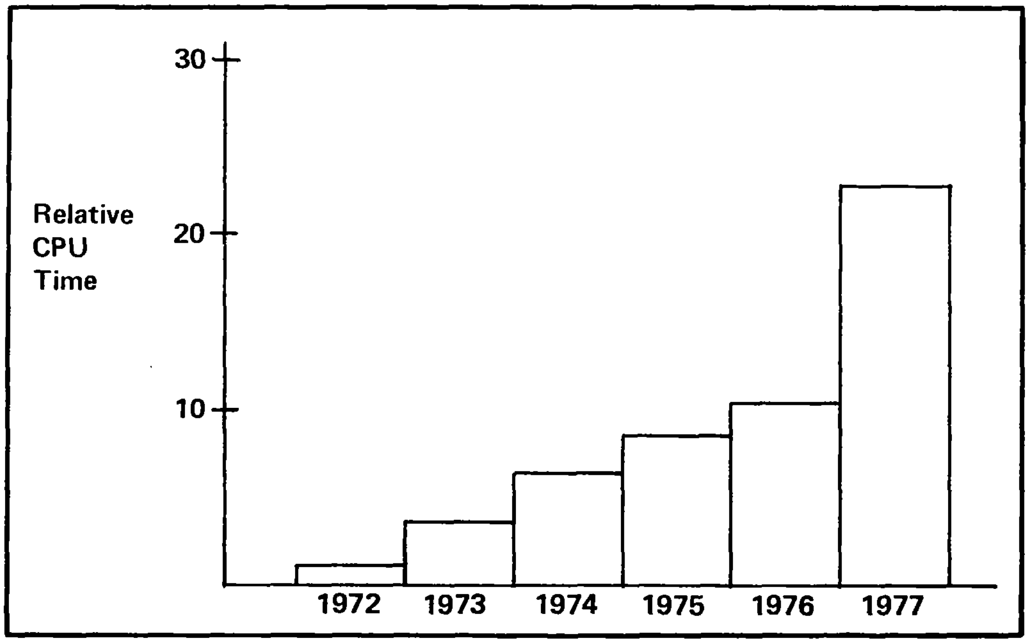


Figure 6.- Computer usage steadily increases.

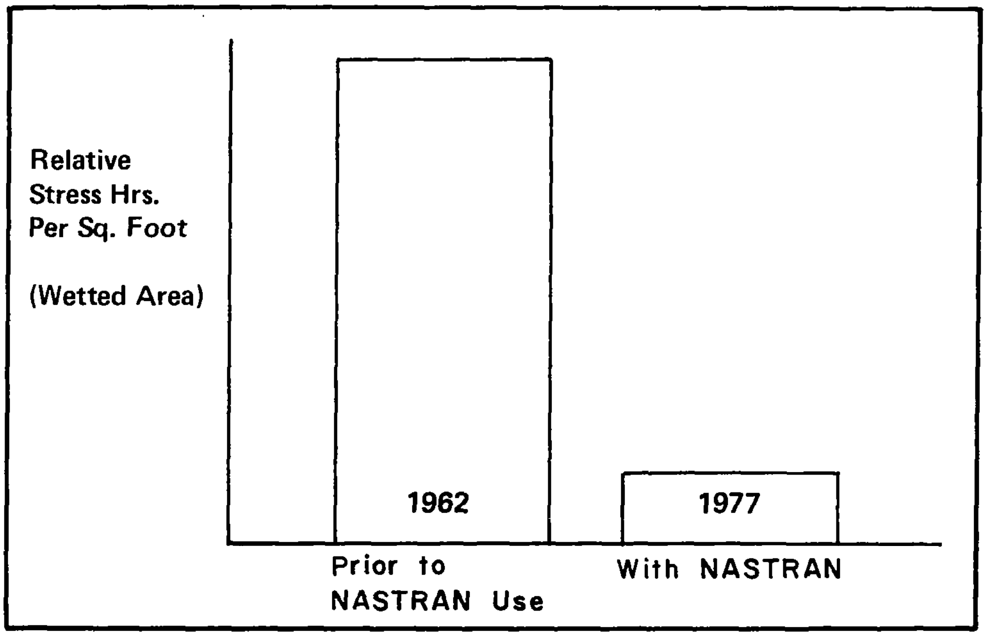


Figure 7.- NASTRAN has reduced stress hours by a factor of 6.

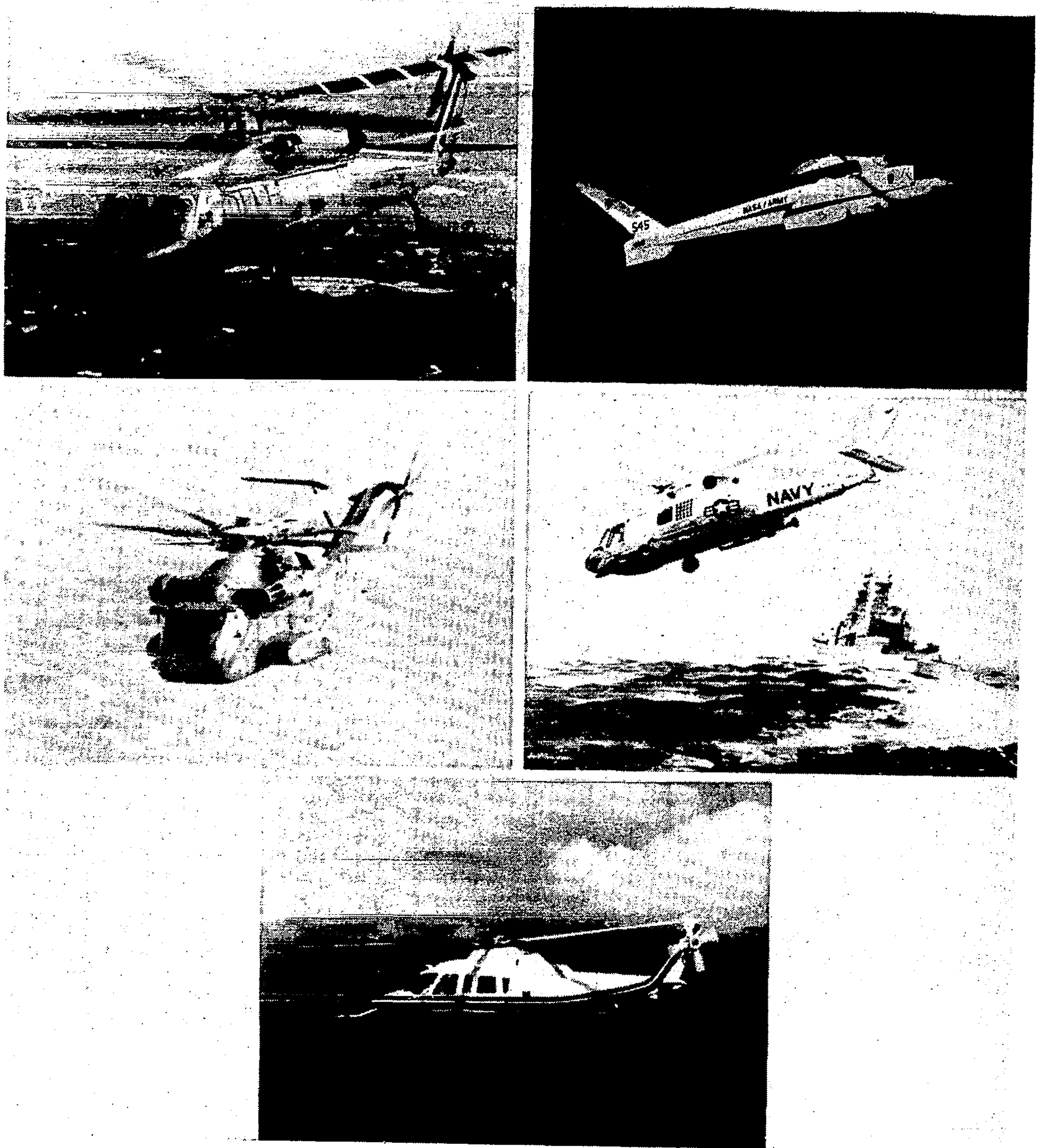


Figure 8.- Finite element analysis now established as standard for helicopter structures.

SYNTHESIS OF AIRCRAFT STRUCTURES USING INTEGRATED DESIGN AND ANALYSIS METHODS - STATUS REPORT

Jaroslaw Sobieszczanski-Sobieski and Robert C. Goetz
NASA Langley Research Center

SUMMARY

This paper gives a status report and describes the future work directions of a systematic research effort to develop and validate methods for structural sizing of an airframe designed with the use of composite materials and active controls. This research program includes procedures for computing aeroelastic loads, static and dynamic aeroelasticity, analysis and synthesis of active controls, and optimization techniques. Development of the methods is concerned with the most effective ways of integrating and sequencing the procedures in order to generate structural sizing and the associated active control system, which is optimal with respect to a given merit function constrained by strength and aeroelasticity requirements.

INTRODUCTION

Aircraft design depends on the strong coupling of a multitude of technical disciplines. For example, wing structural sizing is strongly influenced by aerodynamic loads and load control devices which are in turn defined by the structure's static and dynamic aeroelastic characteristics. Conventionally, these interdependencies are included in the design process through a sequence of analyses and iterative reanalyses. An example of one such conventional iterative process used is illustrated in figure 1 (ref. 1). In this process the structure is first sized for strength and is then resized to add stiffness, if necessary, to satisfy flutter requirements. This process becomes more complex when the vehicle includes active control systems, such as flutter and loads control which must be designed concurrently with the structure itself.

A recognized deficiency of the conventional approach, which amounts to a series of suboptimizations, is the inability to optimize the final configuration (refs. 2-5). This inability is because the conventional approach does not maximize a single merit function (objective function) for the total system. That is, an assembly of coupled subsystems, even if each is optimized individually, does not constitute an optimum for the whole system. This deficiency is usually aggravated by economic and time constraints which in practice often preclude closure of all of the iteration loops.

A mathematically consistent alternative approach is needed which leads to an optimal system with all constraints satsified and all couplings accounted for. This approach would integrate all of the analyses into one iterative loop with a formalized optimization algorithm as illustrated in figure 2. This approach, although conceptionally simple, cannot be readily implemented today for complex aircraft configurations because of the prohibitive computational costs associated with the large number of repetitive analyses required by the very large number of design variables and constraints.

Langley Research Center has undertaken an activity to develop the analysis and synthesis techniques needed to implement the integrated optimization method shown in figure 2. This activity is a Program to Integrate Controls, Aerodynamics, Structures, Software and Optimization for Vehicle analysis and synthesis (PICASSO) (fig. 3). The long-term goal of PICASSO is to develop an integrated multidisciplinary analysis and synthesis methodology for a wide range of aerospace vehicles. Emphasis is focused on developing the methodology necessary to include composite structures, advanced technology aerodynamics, and active controls. The purpose of this paper is to describe the status and future plans of that part of the total effort which is associated with structural synthesis including active controls.

INTEGRATED APPROACH

Several computer codes have already been integrated into a versatile, modular system (ref. 6) to explore various structural synthesis and optimization methods. This system consists of a data base and executive software as depicted in figure 4. The data base includes computer codes to do specific calculations, e.g., a finite element code to do structural analysis (SPAR, ref. 7), numerical data describing mathematical models of specific vehicle configurations, and sets of executive commands in which each set is a procedure carrying out a typical engineering task, e.g., structural resizing for given loads.

Executive software is entirely provided by the Control Data Corporation (CDC) Network Operating System (NOS 1.2). This software consists of a command language and auxiliary utilities which permit storing and retrieving files containing codes, data, execution procedures, and file modification.

Codes currently available in the system are listed in figure 4. Information about these codes is provided in reference 8. In order to utilize existing codes and those which will be developed in the future, a guideline in the development of the integrated system is the ability to incorporate existing codes without internal code changes. This is accomplished by suitable pre- and post-processors. These processors are usually small, stand-alone FORTRAN codes, which perform data conversions to bridge differences between a given code input-output format and the system data storage format. The computer system enables several users to experiment simultaneously on a time-sharing basis with the computations sequenced in various ways within typical engineering tasks, as shown in the right-hand side portion of figure 4.

To develop new synthesis methodology, the integrated system is being used to study various aircraft configurations. Included are several configurations of a supersonic transport and a subscale model of a fuel efficient subsonic transport. These configurations were selected to represent diverse types of subsonic and supersonic aircraft with both low and high aspect ratio wings. As an example of the complexity of the mathematical model available in the data base, a finite element model of one supersonic transport configuration is shown in figure 5 (ref. 9). One half of the symmetric model contains 750 grid points, 2140 elastic degrees of freedom, and 2400 elements representing construction details as shown in the inserts. The number of variables used to size the structure varies from 720 for the metal wing to over 1900 for the composite wing.

ANALYSIS AND SYNTHESIS METHODOLOGY

Through application studies, the system of integrated computer codes described in the preceding section provides a tool for investigating and developing analysis and synthesis methodology. It is also useful in defining missing technical capability and to identify operations which are now impractical and need further development.

Structural Sizing for Strength

Current capability.- Conventional iterative analyses of aerodynamic loads on a flexible aircraft and iterative resizing of structural components are illustrated in the left of figure 6, as a sequence of two iterative loops, nested in a third overall loop. The sequence of these operations when combined into a common iterative loop is shown in the right of figure 6. Detailed discussion of this new iteration approach is given in reference 10. Resizing of the structure is accomplished by using a nonlinear mathematical programming technique designed to minimize weight of each individual structural component separately. Each component is subject to constraints of strength and local buckling due to internal forces acting from the surrounding structure. These forces are held invariant in the optimization of each component and are updated by analysis of the complete structure. The update analysis is carried out after all components have been optimized. The resizing requires several repetitive iterations (usually 3 to 7). Figure 7 gives an example of a wing resized by this approach. Indicated in the figure is the level of detail of the approach where very localized component dimensions are included as design variables. A more detailed description of the approach is presented in references 9, 11, and 12.

Development direction.- As pointed out in reference 11, this approach does not guarantee a minimum weight design since there is no system level objective function to which all structural components would contribute. To remove this shortcoming, a method similar to that proposed in reference 4 (a systematic, multilevel optimization) is to be implemented and evaluated for metal and composite structures.

Experience to date suggests that improved structural analysis is a key element to future system synthesis. Improvements are necessary to trade, in a controlled manner, analysis accuracy for computational cost and to generate, as part of the solution, sensitivity (gradient) information needed for optimization. Two concepts for trading analysis accuracy and computational cost (and producing gradient information) are proposed in references 13 and 14, while decomposition of a large structural analysis problem into smaller subproblems by substructuring is discussed in reference 15. These concepts are broadly referred to as dimensionality reduction and extrapolation methods. Implementation of these concepts are enhanced by several techniques reported in the last decade, which originated from the need to correlate mathematical models with experimental data (refs. 16-18). Seven of these promising techniques are outlined on figure 8 and are currently being evaluated.

Another key need lies in the broad area of computational aerodynamics, because of its obvious impact on the accuracy of the structural analyses results. Improvements are critically needed for predicting loads at transonic speeds, high angle of attack, and for supercritical airfoils. These requirements are for complete wing-body-empennage configurations and include accuracy versus computational cost and data on sensitivity to changes in the configuration geometry and dynamic characteristics.

Structural Sizing for Flutter

Current capability.- Using conventional design methodology a strength resized airframe is analyzed for flutter and, if necessary, stiffened by adding layers of new material as shown for a representative metal wing in figure 9. The amount of new material added is minimized using a nonlinear programming method. The methodology is innovative in two ways: the new material is added to strength sizing as a new minimum gage (ref. 9); the flutter analysis is carried out on a simplified finite element model as compared to the one used in strength resizing (ref. 19).

Development direction.- The next step is to simultaneously combine flutter and strength optimization to realize the benefits of reduced weight as described in reference 2. These benefits are particularly large if the directional properties of composite material are to be exploited (aeroelastic tailoring) as reported in references 20-22.

Structural Sizing for Gust Load Response

Current capability.- Strength sized and flutter free flexible airframes are subjected to a comprehensive dynamic gust response analysis by methods described in reference 23. These methods are computationally expensive for structural synthesis procedures. Therefore, resizing is carried out by a well-known quasi-static gust method which reduces the gust to another steady state maneuver.

Development directions.- For highly gust sensitive aircraft configurations, it will be necessary to include gust as another constraint in the strength-flutter optimization. Therefore, the dynamic response methods will have to be modified for more efficient repetitive use in the optimization loop. A mathematically rational way of combining statistically defined gust stresses with the deterministic stresses due to maneuvers is being developed. Two candidate approaches are under consideration. One is a combinatorial approach to define the worst possible combination of statistical stresses to be superimposed on the deterministic stresses. The second approach replaces the worst combination with the equal probability combinations.

Structural Sizing Including Active Controls

Current capabilities.- Analytical techniques for active flutter suppression analysis and synthesis are defined in references 24-26. Synthesis capability for flutter suppression exists using modern control theory (ref. 24), the "aerodynamic energy method" (ref. 25), and classical control theory (ref. 26). The results of modern control theory are being practicalized, without having to measure the states of each variable, through the use of nonlinear programming techniques.

An example of active flutter suppression for a strength sized supersonic transport is shown in figure 10. The figure shows the vehicle flutter boundaries with respect to the flight envelope. The dashed line boundary indicates a flutter deficiency of the airframe sized for strength. By using active flutter suppression the flutter boundary is shifted to a position indicated by the solid line, which is outside of the flight envelope. The weight of the active control system is estimated according to reference 27. This weight is about five times smaller than the weight requirement for a structural fix defined in the studies reported in reference 9.

Development directions.- In addition to flutter suppression, capability is being developed to synthesize control systems for gust load alleviation, maneuver load control, and relaxed static stability, using the methods mentioned above. Once these capabilities are developed, studies to determine the maximum benefits on structural sizing and weight by integrating active controls into the initial design stage will be undertaken. The use of formal optimization techniques is to be further expanded to include the active control surface size and location as design variables. Additionally, the optimum manner of controlling aeroelastic behavior will be explored. This will include combining the passive control benefits of composite aeroelastic tailoring and the benefits of active controls.

Improvements are also needed for predicting distributed loads on oscillating control surfaces, especially on supercritical wings. These improvements are needed for the determination of control surface effectiveness and control system authority.

CONCLUDING REMARKS

The coupling between structures, aerodynamics and active controls points to a need for a mathematically rational unified optimization methodology to shape and size the airframe structure. This structural synthesis methodology should be based on a given merit function (e.g., weight) subject to realistic constraints (e.g., strength and stiffness).

A systematic research program has been established to achieve the development of such an optimization capability. Research on unified strength and flutter optimizations and resizing for gust response is in progress. A parallel effort to improve analysis and synthesis techniques for active controls is also underway. Structural and active control synthesis development is intended to provide the capability to predict the optimum control of static and dynamic aeroelastic behavior of airframes by passive and active means.

Missing elements which need further development for a totally integrated optimization method are

(1) Analytical formulations that give sensitivity results and permit trades between accuracy and computational speed for static and dynamic structural behavior, for definition of steady and unsteady aerodynamic loadings, and for description of active control systems.

(2) Multilevel optimization procedures which permit addressing of global constraints (e.g., flutter) and local constraints (e.g., buckling of a wing cover panel stiffened with stringers).

(3) Advanced computational aerodynamics for conventional and supercritical wings with controls in the transonic and high angle of attack regimes.

(4) Active control synthesis techniques for relaxed static stability and the control of loads.

REFERENCES

1. Sakata, I. F.; and Davis, G. W.: Advanced Structures Technology Applied to a Supersonic Cruise Arrow Wing Configuration. Proceedings of the SCAR Conference, NASA CP-001, Part 2, November 1976, pp. 603-636.

2. Stroud, W. J.; Dexter, C. B.; and Stein, M.: Automated Preliminary Design of Simplified Wing Structures to Satisfy Strength and Flutter Requirements. NASA TN D-6534, December 1971.

3. Sobieszczanski, J.: Sizing of Complex Structure by the Integration of Several Different Optimal Design Algorithms. AGARD-LS-70, Sept. 1974.

4. Schmit, L. A., Jr.; and Ramanathan, R. K.: Multilevel Approach to Minimum Weight Design Including Buckling Constraints. AIAA Journal, vol. 16, no. 2, Feb. 1978, pp. 97-104.

5. Lasdon, L. S.: Optimization Theory for Large Systems. McMillan Co., New York, 1970.

6. Sobieszczanski, J.: Building a Computer-Aided Design Capability Using a Standard Time Share Operating System. Integrated Design and Analysis of Structures, Proceedings of ASME Winter Annual Meeting, Houston, TX December 1975.

7. Whetstone, W. D.: SPAR Structural Analysis System Reference Manual. NASA CR-145098-1, February 1977.

8. Giles, G.: Computer-Aided Methods for Analysis and Synthesis of Supersonic Cruise Aircraft Structures. Proceedings of the SCAR Conference, NASA CP-001, Part 2, Nov. 1976, pp. 637-658.

9. Sobieszczanski, J.; McCullers, L. A.; Ricketts, R. H.; Santoro, N. J.; Beskenis, S. D.; and Kurtze, W. L.: Structural Design Studies of a Supersonic Cruise Arrow Wing Configuration. Proceedings of the SCAR Conference, NASA CP-001, Part 2, Nov. 1976, pp. 659-684.

10. Giles, G. L.; and McCullers, L. A.: Simultaneous Calculation of Aircraft Design Loads and Structural Member Sizes. AIAA 1975 Aircraft Systems and Technology Meeting, Los Angeles, CA, Aug. 1975. AIAA No. 75-965.

11. Sobieszczanski, J.; and Loendorf, D.: Automated Sizing of Large Structures by Mixed Optimization Methods. Second Symposium on Structural Optimization, Milan, Italy, April 1973. AGARD Conference Proceedings No. 123.

12. Sobieszczanski-Sobieski, J.: An Integrated Computer Procedure for Sizing of Composite Airframe Structures. NASA TP-1300, 1978.

13. Sobieszczanski, J.; and Storaasli, O. O.: On the Accuracy of the Taylor Approximation for Structure Resizing. AIAA Journal, vol. 12, no. 2, Feb. 1974, pp. 231-233.

14. Noor, A. K.; and Lowder, H. E.: Approximate Techniques of Structural Reanalysis. Computers and Structures, vol. 4, 1974, pp. 801-812.

15. Noor, A. K.; Kamel, H. A.; and Fulton, R. E.: Substructuring Techniques-Status and Projections. Computers and Structures, vol. 9, no. 1, 1978.

16. Chen, J. C.; and Wada, B. K.: Matrix Perturbation for Structural Dynamic Analysis. AIAA Journal, vol. 15, no. 8, August 1977.

17. Berman, A.; and Flannelly, W., G.: Theory of Incomplete Models of Dynamic Structures. AIAA Journal, vol. 9., no. 8, August 1971, pp. 1481-1487.

18. Chen, J. C.; and Wada, B. K.: Criteria for Analysis - Test Correlation of Structural Dynamic Systems. Journal of Applied Mechanics, Transactions of ASME, vol. 42, no. 2, June 1975, pp. 471-477.

19. Ricketts, R. H.; and Sobieszczanski, J.: Simplified and Refined Structural Modeling for Economical Flutter Analysis and Design. AIAA 18th Structures, Structural Dynamics and Materials Conference, San Diego, Cal., March 1977. AIAA Paper No. 77-421.

20. Aeroelastic Tailoring of Advanced Composite Structures for Military Aircraft. Vol. 3, User's Guide for Procedure TSD, AFFDL-TR-76-100, Feb. 1978.

21. McCullers, L. A.: Automated Design of Advanced Composite Structures. ASME Winter Annual Meeting, Structural Optimization Symposium, N. Y., Nov. 1974, pp. 119-134.

22. Starnes, J. H.; and Haftka, R, T.: Preliminary Design of Composite Wings for Buckling Strength and Displacement Constraints. Proceedings of AIAA/ASME 19th Structures, Structural Dynamics and Materials Conference, Apr. 1978. AIAA Paper No. 78-466.

23. Miller, R. D.; Kroll, R. I.; and Clemmons, R. E.: Dynamic Loads Analysis System (DYLOFLEX) Summary. NASA CR-2846, May 1978.

24. Newsom, J. R.: Synthesis of Active Flutter Suppression Systems Using Optimal Control Theory. AIAA Guidance and Control Conference, Palo Alto, Cal., Aug. 1978. AIAA Paper No. 78-1270.

25. Nissim, E.; and Abel, I.: Development and Application of an Optimization Procedure for Flutter Suppression Using the Aerodynamic Energy Concept. NASA TP 1137, Feb. 1978.

26. Visor, O. E.; and Sevart, F. D.: Preliminary Design Study of a Flutter Suppression Control System for the BQM-34E/F Drone Aircraft with a Supercritical Wing-Final Report. NASA CR-145208, 1976.

27. Anderson, R. D.; Flora, C. C.; Nelson, R. H.; Raymond, E. T.; and Vincent, J. H.: Development of Weight and Cost Estimates for Lifting Surfaces With Active Controls. NASA CR-144937, 1976.

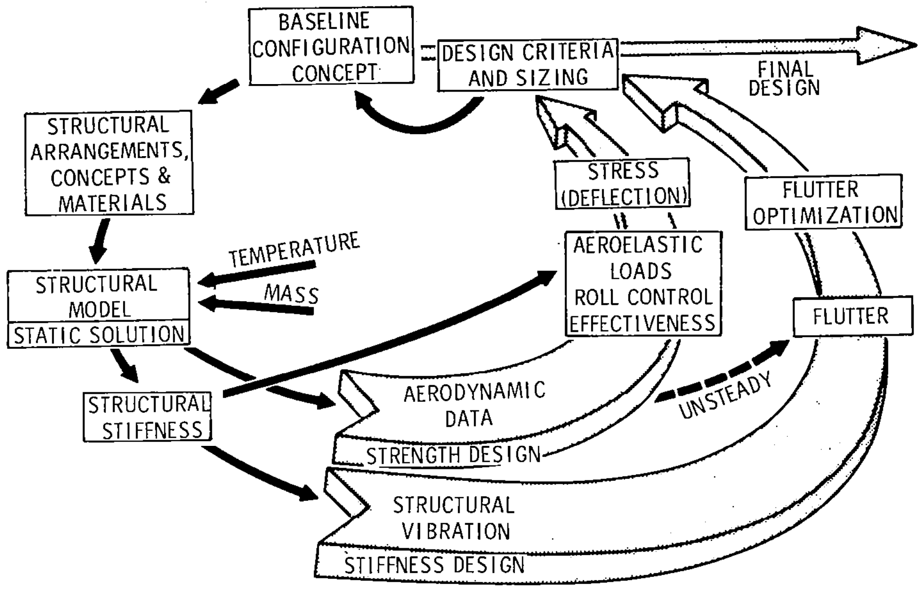


Figure 1 Strength and flutter resizing - typical application (ref. 1).

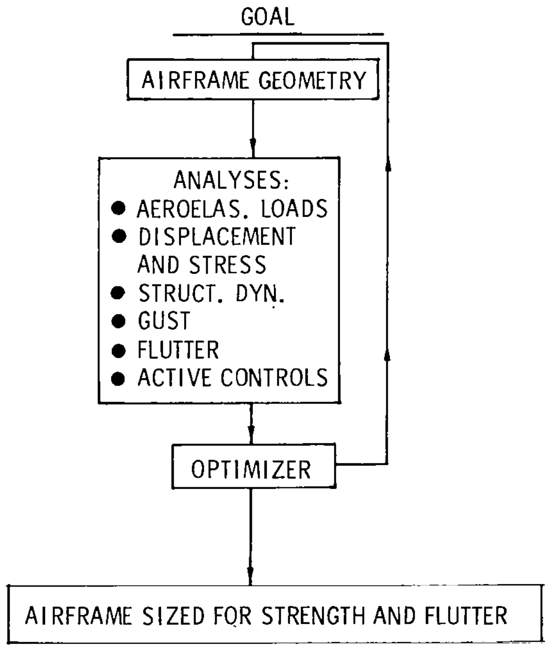


Figure 2 Integrated optimization method.

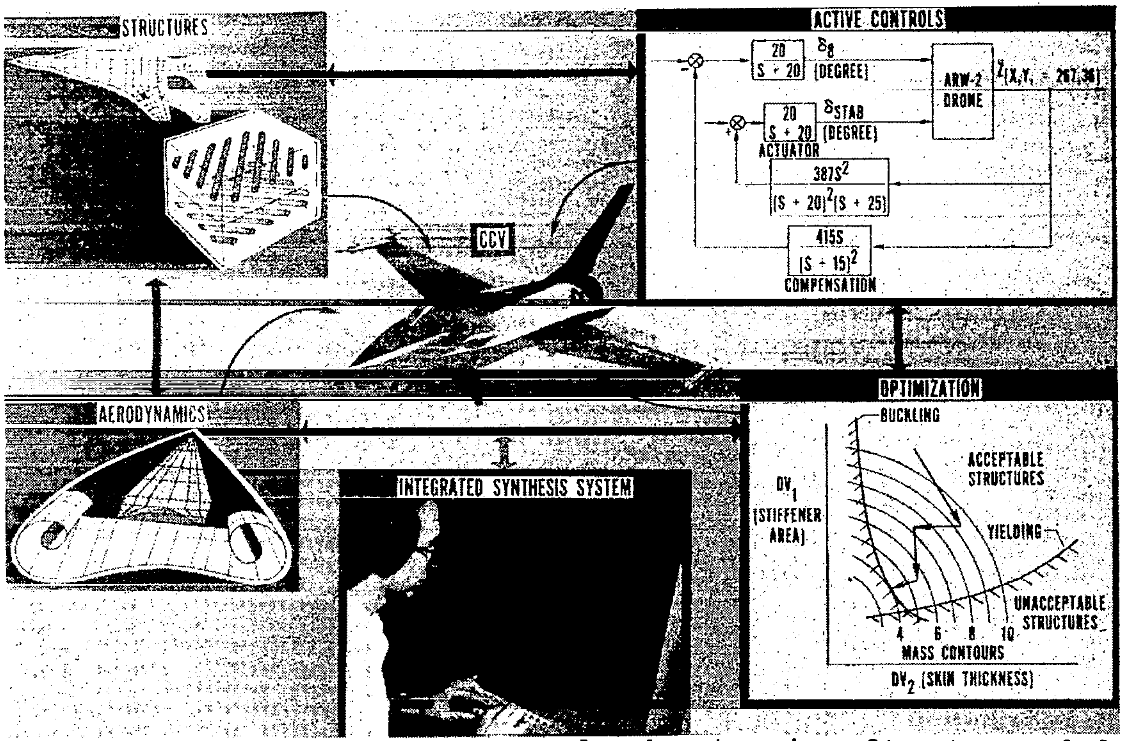
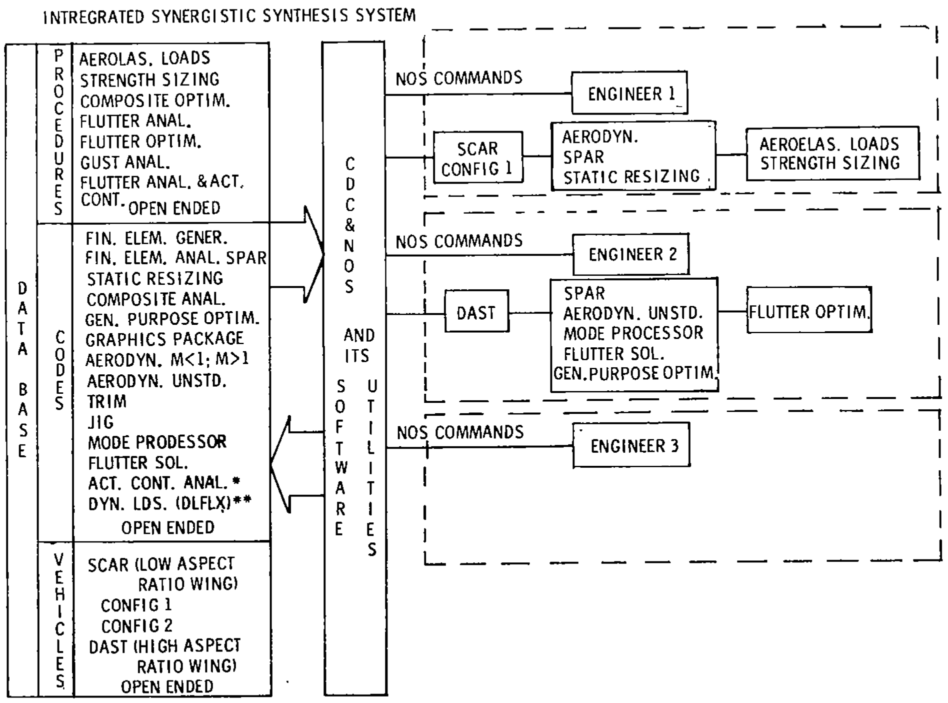


Figure 3 Program to Integrate Controls, Aerodynamics, Structures, Software, and Optimization (PICASSO).



* EXTERNALLY LINKED
** IN THE INSTALLATION PROCESS

Figure 4 Present integrated system of programs and data.

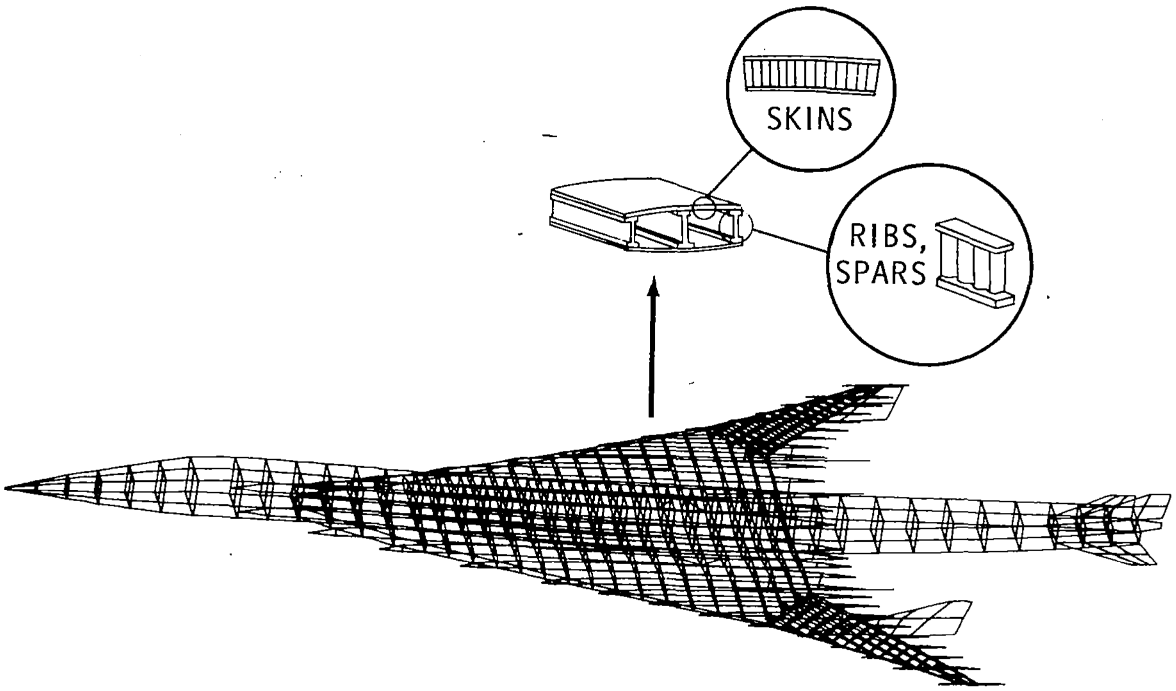


Figure 5 Finite element model of a supersonic transport aircraft (ref. 9).

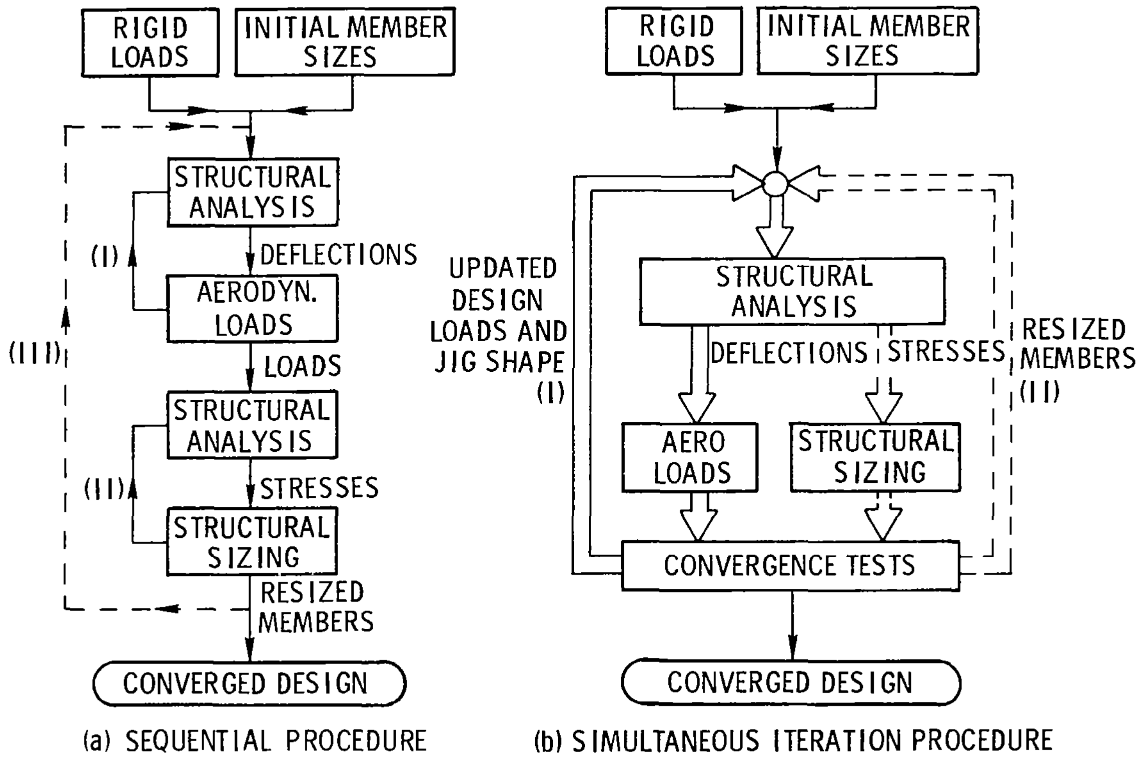


Figure 6 Conventional and improved procedures for aeroelastic load and structural resizing (ref. 10).

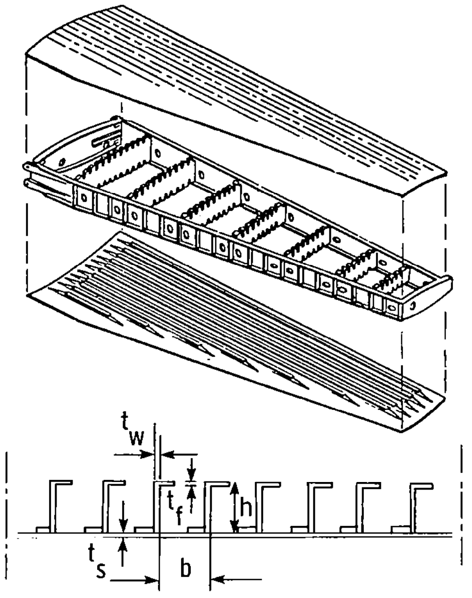
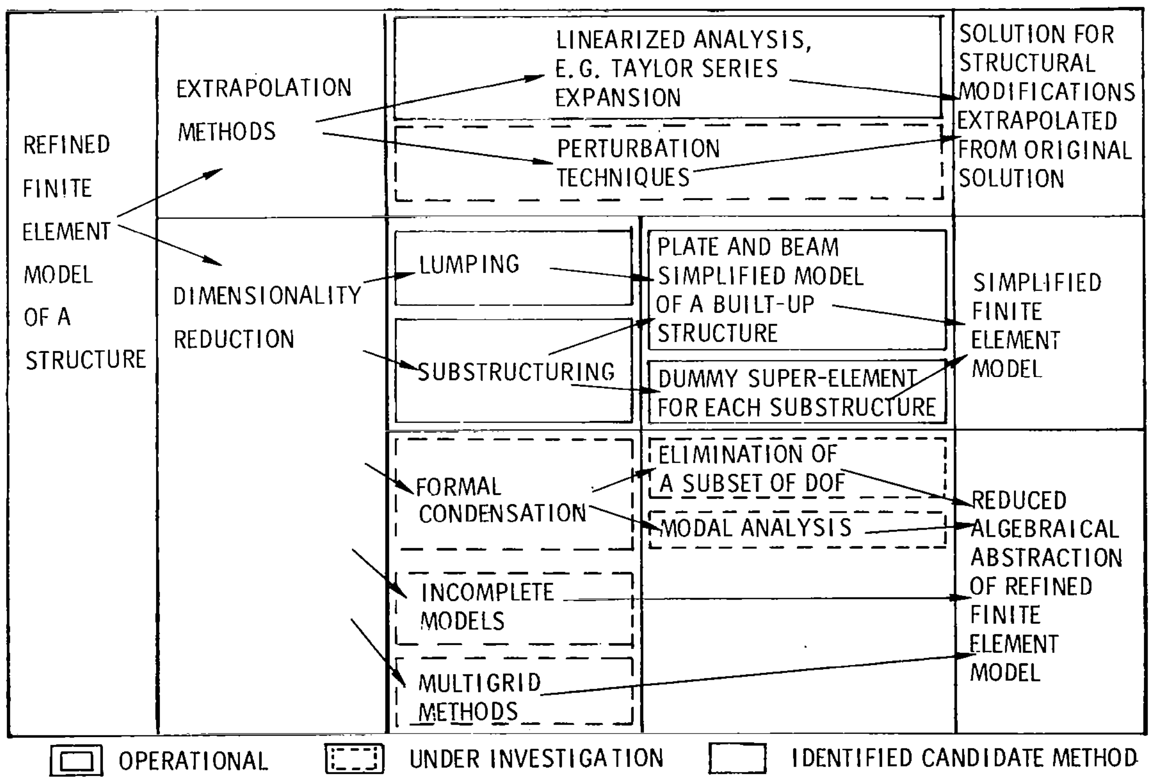
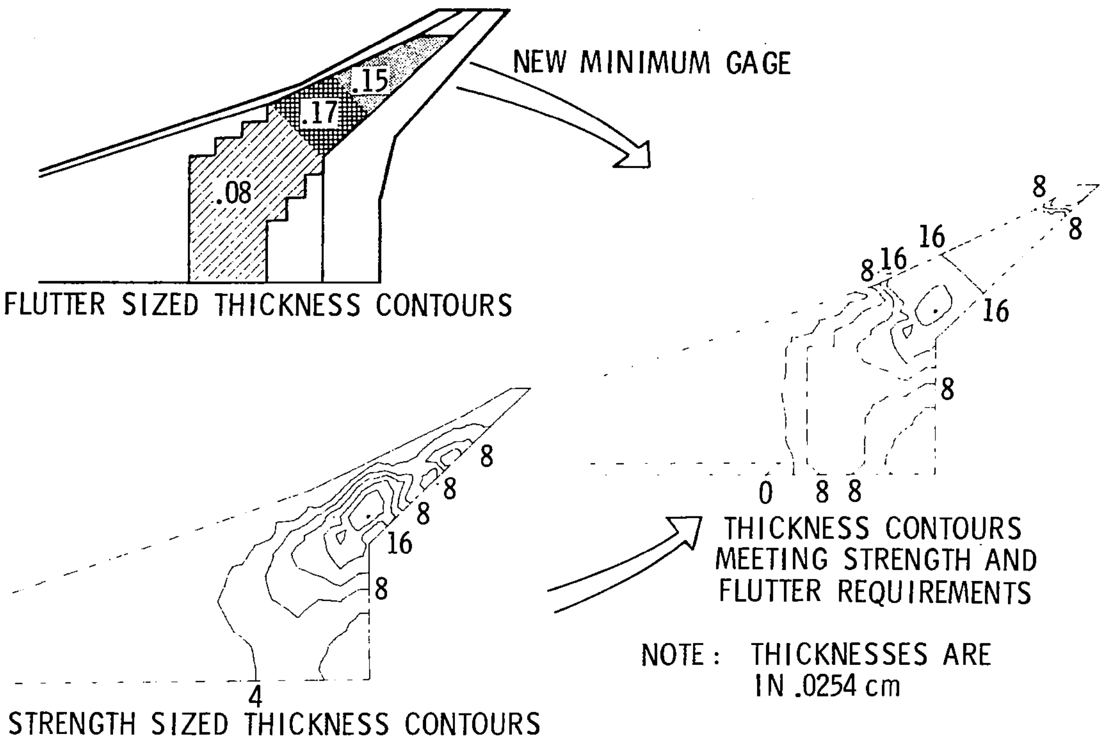


Figure 7 Example of the optimization of the wing structural components.



REFINED FINITE ELEMENT MODEL OF A STRUCTURE
EXTRAPOLATION METHODS
DIMENSIONALITY REDUCTION
LINEARIZED ANALYSIS, E.G. TAYLOR SERIES EXPANSION
PERTURBATION TECHNIQUES
SOLUTION FOR STRUCTURAL MODIFICATIONS EXTRAPOLATED FROM ORIGINAL SOLUTION
LUMPING
SUBSTRUCTURING
PLATE AND BEAM SIMPLIFIED MODEL OF A BUILT-UP STRUCTURE
DUMMY SUPER-ELEMENT FOR EACH SUBSTRUCTURE
SIMPLIFIED FINITE ELEMENT MODEL
FORMAL CONDENSATION
ELIMINATION OF A SUBSET OF DOF
MODAL ANALYSIS
INCOMPLETE MODELS
MULTIGRID METHODS
REDUCED ALGEBRAICAL ABSTRACTION OF REFINED FINITE ELEMENT MODEL
OPERATIONAL
UNDER INVESTIGATION
IDENTIFIED CANDIDATE METHOD

Figure 8 Candidate methods to accelerate structural analysis.



NEW MINIMUM GAGE
.15
.17
.08
FLUTTER SIZED THICKNESS CONTOURS
8
16
16
8
16
8
0
8
8
THICKNESS CONTOURS MEETING STRENGTH AND FLUTTER REQUIREMENTS
NOTE: THICKNESSES ARE IN .0254 cm
8
8
8
16
8
4
STRENGTH SIZED THICKNESS CONTOURS

Figure 9 Example of flutter stiffening by imposing a new minimum gage on the strength design (ref. 9).

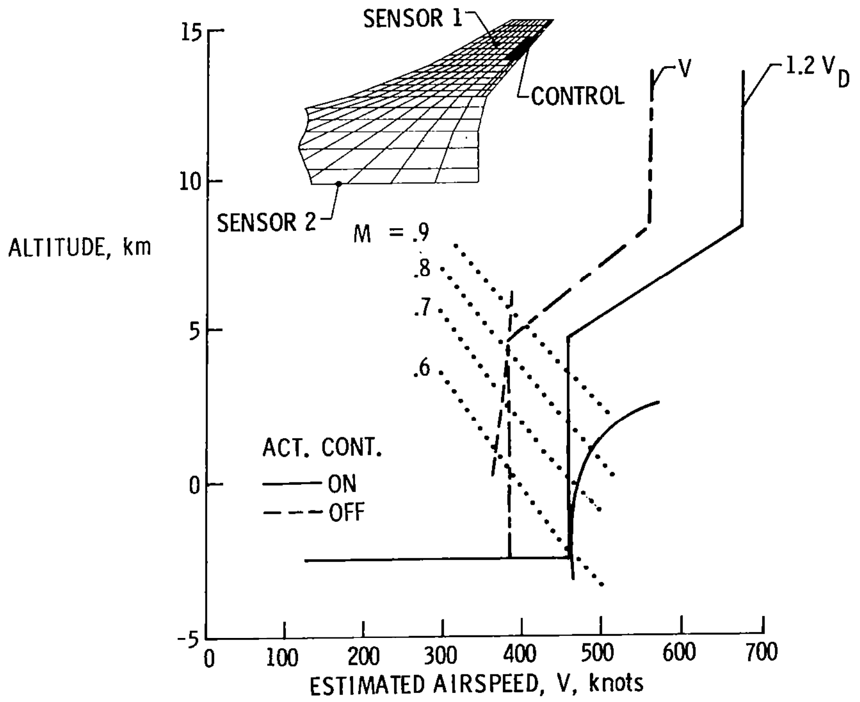


Figure 10 Flutter suppression by means of active controls. M is Mach number and V_D is diving velocity.

FINITE ELEMENT ANALYSIS IN A MINICOMPUTER/MAINFRAME ENVIRONMENT

Olaf O. Storaasli
NASA Langley Research Center

Ronald C. Murphy
Joint Institute for Advancement of Flight Sciences
The George Washington University

INTRODUCTION

The advent of modular finite element systems provides the opportunity for engineers to solve a broad range of structures problems in a distributed computing environment. To maintain versatility in this changing computing environment (ref. 1), changes may be appropriate in the design concepts of future finite element systems. Recent exploratory studies (refs. 2-3) have shown that minicomputers offer great potential for solving structures problems. The purpose of this paper is to investigate design considerations for general purpose finite element systems to maximize performance when installed on distributed computer hardware/software systems. This paper explores how the features of current minicomputers complement those of a modular implementation of the finite element method. Central to this investigation is increasing the control, speed, and visibility (interactive graphics) by structural engineers in solving a broader range of structural problems at reduced cost. The approach used is to implement a finite element system in a distributed computer environment to solve structural problems and to explore alternatives in distributing finite element computations.

THE FINITE ELEMENT METHOD

To implement the finite element method on computers for typical static, dynamic, buckling, and thermal analyses, two approaches are commonly used. The first approach (fig. 1(a)), and the one which dominated software design concepts prior to the advent of virtual memory, is to use an executive program to connect and communicate with analysis overlays in a fixed, serial fashion. This method is known to lack modularity, portability, and efficiency and often requires significant effort to make minor software changes. The second approach (fig. 1(b)) is to implement each analysis activity of the finite element method as an independent processor and have all processors communicate through a common data base. The key to implementing such a modular approach is to have system and/or data base utility software (ref. 4) to open, read, and write to named files from within finite element processors. In the finite element procedure, the function of each processor can be selected to minimize computing time and memory. Such a modular approach is the basis of the implementation of the SPAR (ref. 5) finite element system on the PRIME computer* and is well suited for use in the current investigation.

It is important to identify how large a problem (degrees of freedom) may be solved conveniently on a minicomputer and which processors are the bottleneck with regard to computation time. Figure 2 shows minicomputer solution times for a range of problems vs. problem size (e.g., number of degrees of freedom). Also shown are projected times based on planned enhancement which should reduce solution times by a factor of two to four. Thus, to achieve solutions to static analysis problems on the minicomputer in less than 30 minutes, a reasonable problem size is about 2500 degrees of freedom. While large problems may be solved conveniently on the minicomputer mostly in a background mode, the 30 minutes shown in figure 2 (horizontal line) is probably a reasonable upper limit for engineering users to maintain thought continuity and work on other activities while background computations are in progress.

By analyzing the solution time for specific components (SPAR processors) of the finite element process, it is possible to identify functions which may be suited for mainframe or array processor calculations (see Results). A high-speed data link connecting the minicomputer to a CDC 6600 (roughly eight times the computation speed of the minicomputer) was explored to transfer "number crunching" activities. Use of this link is to minimize overall computation time and yet preserve the advantages of quick response and high-speed user interface provided by the minicomputer for structural engineering activity which involves interaction.

COMPUTER HARDWARE AND SOFTWARE

Today's minicomputers have similar capabilities to large mainframe computers, except the cost and CPU speed are about an order of magnitude less. (Table 1). Table 1 shows results of a benchmark test run on fifteen computers to simulate structures calculations (double precision matrix operations using nested DO loops). On the left are times to run the benchmark for seven large mainframes, and on the right are times to run the same benchmark for eight minicomputers. The table illustrates variations in both performance and cost

*The SPAR-minicomputer version, in use at NASA Langley Research Center and on several NASA contracts, is now available from COSMIC, the distribution center for NASA software.

for both mainframes and minicomputers. For interactive engineering calculations, many of the "mainframe" peripherals (high-speed card readers, line printers, punches, etc.) are not required.

TABLE 1.- CPU TIME FOR STRUCTURES BENCHMARK

MAINFRAME	TIME (SEC)	MINICOMPUTER	TIME (SEC)
CDC CYBER 175	2	DEC VAX 11/780	23
IBM 360/95	3	DEC PDP 11/70	42
IBM 370/168	4	PRIME 500	47
CDC 6600	8	SEL 32/75	52
IBM 360/75	10	PRIME 400	65
CDC CYBER 173	12	SIGMA V	71
UNIVAC 1108	15	SEL 32/55	72
		MODCOMP IV	85
Cost Range ($2-6 Million)		Cost Range ($50-150 Thousand)	

Figure 3, upper left, shows one of the minicomputers at Langley Research Center on which the SPAR finite element code has been implemented. This PRIME 400 minicomputer contains 32-bit arithmetic registers, 192 000 16-bit words of real memory, and 80 million words of disk storage, and costs about $150 000. The virtual memory on the minicomputer permits each time-share user a working space in excess of 1 million words. Currently, seven high-speed (4800-9600 baud) data lines link seven Tektronix 4014 graphics terminals to the minicomputer.

Figure 3 also shows a 4800 baud intercomputer data link which permits lengthy iterative ("number crunching") activity to be transferred from the minicomputer to the large mainframe computer by entering a simple command from any Tektronix terminal. The primary reason for the minicomputer as the user interface (see refs. 2-3) was the increased capability available to users (through both hardware and software advances) at a significantly reduced cost when compared to time-shared computing on a large mainframe.

RESULTS

Approximately twenty smaller problems (less than 2500 degrees of freedom) were solved entirely on the minicomputer and each result was obtained within 30 minutes. For these problems, obtaining solutions on the minicomputer in a stand-alone mode was satisfactory with no need to off-load portions of calculations to faster computation devices. However, for three larger problems (figs. 4-6) the trends indicate that large finite element problems should be solved in a distributed computing environment which contains high-speed computing capabilities.

Figure 4 is a minicomputer plot of a finite element model of a current NASA flight project vehicle typical of a problem whose solution time on the minicomputer is less than 30 minutes. The model has 1120 degress of freedom and 450 structural nodes and consists of 728 two-node and 374 four-node elements. Symmetry constraints about the aircraft center line and y-constraints on the wing leading and trailing edges were imposed, and rigid masses were used in the fuselage. Load cases simulated were a fuel inertia relief maneuver condition, a cruise condition, and a taxi condition. The wing model has three degrees of freedom at each structural node point.

Figure 5 shows a finite element model of a launch umbilical tower with 2208 degrees of freedom, 372 structural nodes, 944 two-node elements, and six degrees of freedom per node. The model was subjected to a downward prestress load of 1 g unit.

Figure 6 is a minicomputer plot of a 4708-degree-of-freedom finite element model of the National Transonic Facility currently under construction at Langley. This cryogenic wind tunnel model is the largest finite element problem attempted thus far on a minicomputer at Langley; it has six degrees of freedom per node and requires 3 CP hours for the static solution.

The finite element models shown in figures 4-6 have distributions of solution time shown in figures 7-9, respectively. Shown on the abscissa of figures 7-9 are components of the finite element process for SPAR as they occur for static analysis. Shown on the ordinate of the figures are the central processor times (CP) in minutes. The processors TAB, ELD, TOPO, and E process the node point, element, topology, and elemental stiffness matrices. Figures 7-9 show that the model generation activity requires little CP time and is well suited for a minicomputer. Formation of the element data packets (EKS) involves significant CP time where a large number of three- and four-node elements are involved. The tower (fig. 8) requires less CP time for EKS (since it contains simple bar elements) even though it has more degrees of freedom than the wing model (fig. 7). Assembly of the global stiffness matrix (K) is the dominant CP activity for the wing model (fig. 7), while the decomposition processor (INV) is dominant for the larger tower and tunnel models (figs. 8-9). For the SPAR finite element system, the decomposition time (INV) is proportional to the cube of the degrees of freedom allowed at a structural node point. Thus, for large models, care should be exercised to include only those freedoms actually required. The remaining loads (AUS), static solution (SSOL), and stress (GSF) processors are less important from the standpoint of CP time for all models. Not shown in the figure are results obtained for free vibration analysis (EIG) which is a major CP user for large complex models. Recent improvements in solution time due to use of a virtual memory loader are shown by the dashed lines in figures 7-9.

Figures 7-9 show that for static analysis of large structures on a minicomputer, the EKS, K, and INV components of the finite element process dominate CP requirements and are prime candidates for being relegated to a mainframe or array processor. The EKS and K processors are less time consuming where two-node elements are used in the finite element process, and the decomposition (INV) processor is dominant for such models with more than 2500 degrees of freedom. Although the results shown in figures 7-9 are a function of the particular finite element system and its implementation on the PRIME 400 minicomputer, the general trends should be representative. In particular, they suggest some advantage to conducting finite element calculations in a distributed computer environment.

ISSUES INVOLVED IN DISTRIBUTING COMPUTATIONS

The above results suggest that, ideally, an automated selection of computer hardware for the EKS, K, and INV processors based on problem size and element complexity should be initiated to minimize the solution cost and time. However, for distributed structural computations, there is still a long way to go before such an automated system is achieved.

The current distributed capability (fig. 3) consists of both hardware and software. The hardware used in the data transfer is a disk on both the minicomputer and the mainframe, a modem on each, a synchronous multi-line controller (SMLC), and a telephone line. The software used includes the protocol supported by the mainframe computer (UT200), communications software on the minicomputer (COMET), and special software written to permit the transfer of SPAR binary data base files between computers. Use of this procedure soon exposed a basic deficiency in that excessive time was spent formatting data into 80-column card images and then reconstructing the data again after data transmission. This excessive formatting time will soon be overcome with the replacement of current protocol (UT200) by a better protocol (HASP) in the new release of the mainframe operating system, which supports the direct transfer of binary data at 9600 baud.

Another alternative being considered is to adapt the finite element software to permit the connection of an array processor directly to the minicomputer to overcome these hardware and software restrictions (i.e., 9600 baud, data transfer, and formatting). This approach looks promising from a technical point of view at present, as do increased CPU speed on minicomputers and the use of certain advanced computer linking devices (i.e., HYPERchannel, ref. 6), with transfer rates of 50 million bytes/sec. The current distributed configuration (fig. 3) at Langley permits computations on both the minicomputer and mainframe by using communications software to transfer SPAR data between the two computers at 4800 baud. However, the transfer process takes longer (in many cases) than the equivalent time for the minicomputer to perform the computations. Future software and hardware enhancements currently planned should, however, remove some of these restrictions.

The authors have already introduced several performance and efficiency improvements in the SPAR minicomputer version and it is clear that sometimes small subtle changes can lead to reductions in cost and time by factors of

2 to 3 or more. The mainframe version of SPAR has been optimized with judicious use of machine code (CDC COMPASS) and such improvements will continue as software packages are tuned to take advantage of specific hardware features. The distributed computations made thus far indicate that the above hardware and software configuration accomplishes the distribution of tasks with a moderate degree of success. However, a 50 million byte/sec computer communication link or judicious use of an array processor on the minicomputer could significantly improve the distributed solution of large finite element problems.

The modularity of the SPAR system made the combination of both mainframe and mini computing environments possible. Future finite element systems should have this feature, modularity in their design, so that time consuming number crunching tasks may be readily distributed to appropriate computing devices (i.e., mainframe computers, array processors, or specially tailored microprocessors) which are better suited for such tasks.

CONCLUDING REMARKS

This paper presents results of exploratory studies on how the modularity of the finite element process can complement the advantages of low-cost, quick-response minicomputers. The finite element process is separated into its basic building blocks (processors) for the SPAR finite element system, and minicomputer central processing (CP) times of each processor are shown for three finite element models. Results are then discussed for the case of a minicomputer linked to a remote mainframe host. It is shown that for problems up to about 2500 degrees of freedom, the performance of the minicomputer in solving the problem in a stand-alone mode is acceptable. While the virtual memory of the minicomputer removes any restriction on problem size, its slower CPU speed tends to place a practical limit on the size of interactive finite element solutions (approximately 2500 degrees of freedom). An initial distributed system is discussed in which computations are performed on both the minicomputer and the mainframe and data transferred between them. The deficiencies of this system are identified and a computer linking system is discussed which makes this distributed system practical. Array processors on minicomputers to carry out high-speed vector calculations may also be viable alternatives which, in many cases, may decrease the need for a high-speed link to the mainframe. Such strategies or combinations thereof should be developed and updated in future finite element systems. Most important, however, future finite element systems should be sufficiently modular to allow the interactive user the opportunity to take advantage of the capability offered by a wide variety of advanced computer hardware, either currently available, or likely to evolve in the near future.

REFERENCES

1. Noyce, R. N.: Microelectronics, Sci. American, vol. 237, no. 3, Sept. 1977, pp. 62-69.

2. Storaasli, O. O.: On the Role of Minicomputers in Structural Design. Comput. & Struct. vol. 7, Feb. 1977, pp. 117-123.

3. Storaasli, O. O.; and Foster, E. P.: Cost Effective Use of Minicomputers To Solve Structural Problems. AIAA Paper No. 78-484 presented at the 19th AIAA/ASME Structures, Structural Dynamics and Materials Conference, Apr. 1978.

4. Giles, G. L.; and Haftka, R. T.: SPAR Data Handling Utilities. NASA TM-78701, 1978.

5. Whetstone, W. D.: SPAR Structural Analysis System Reference Manual, Vols. 1-3. NASA CR-145098-1-3, 1977.

6. McCusker, Tom: Talk Is Cheap Over Thornton's Channels. Datamation, vol. 22, no. 3, Mar. 1976, pp. 166-170.

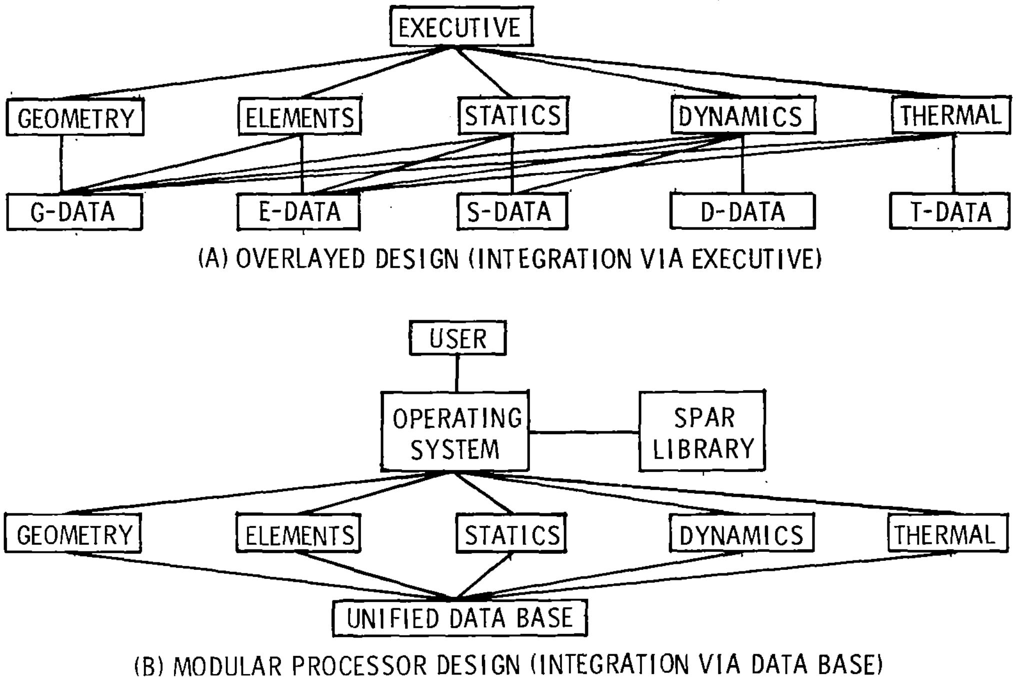


Fig. 1 Finite element software architectures.

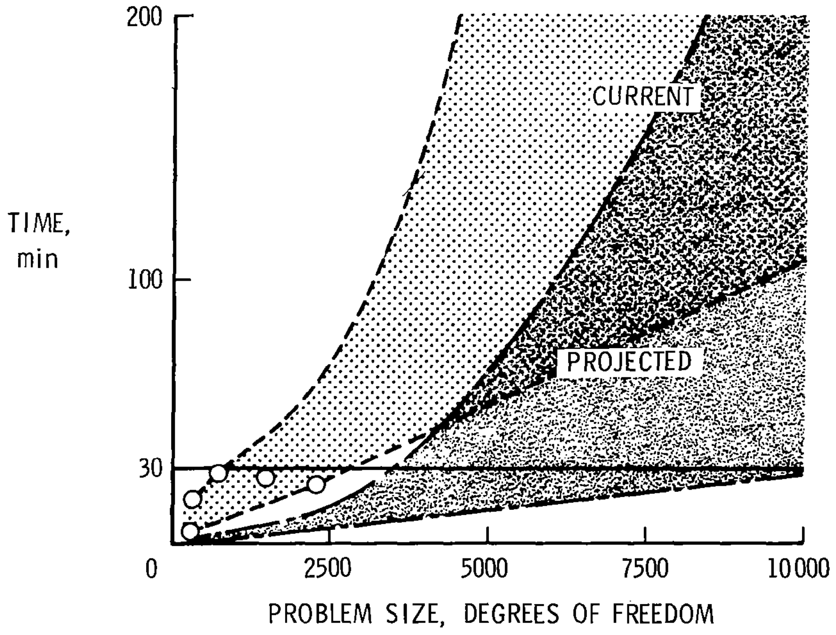


Fig. 2 Minicomputer solution time vs. problem size.

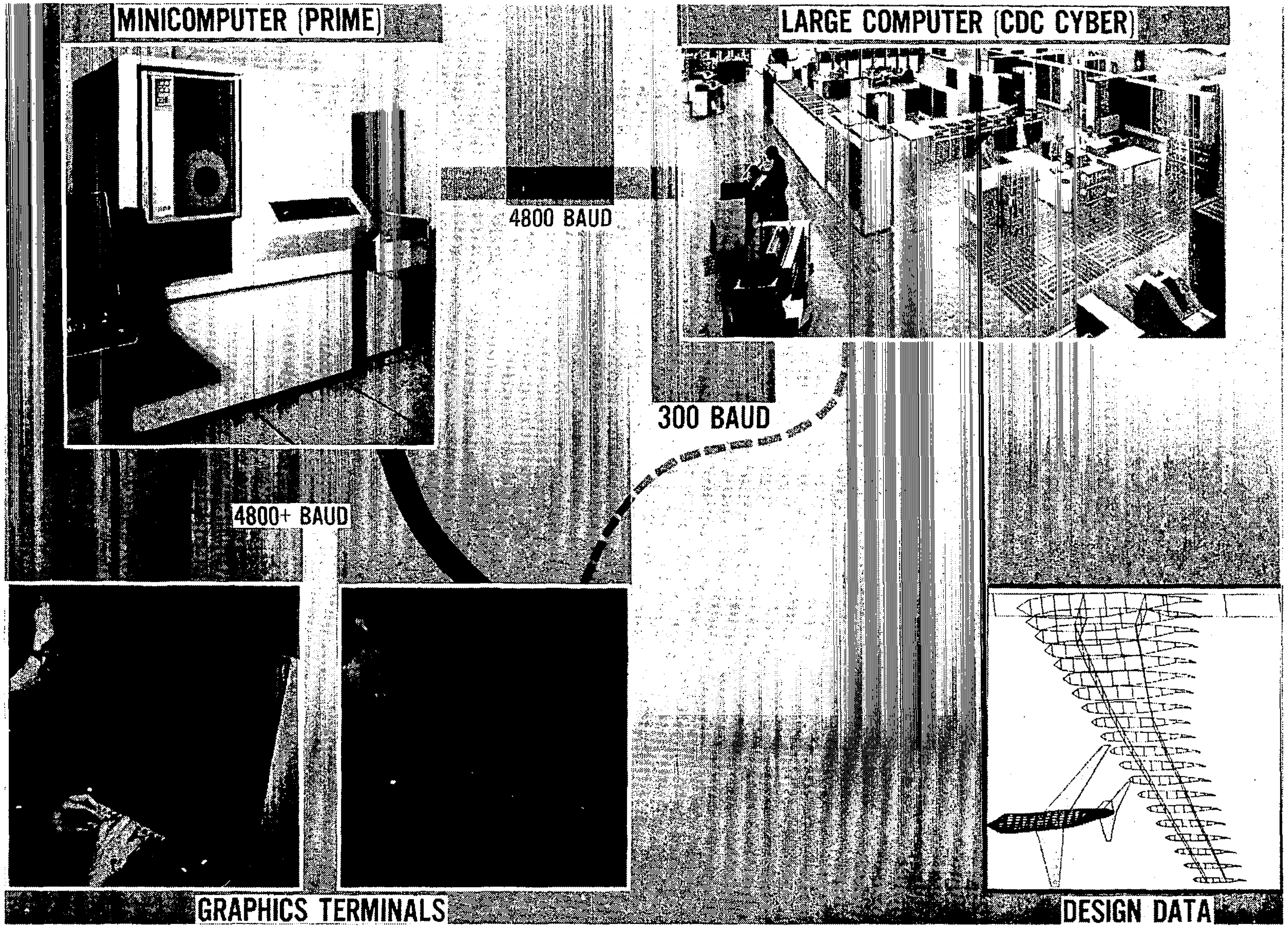
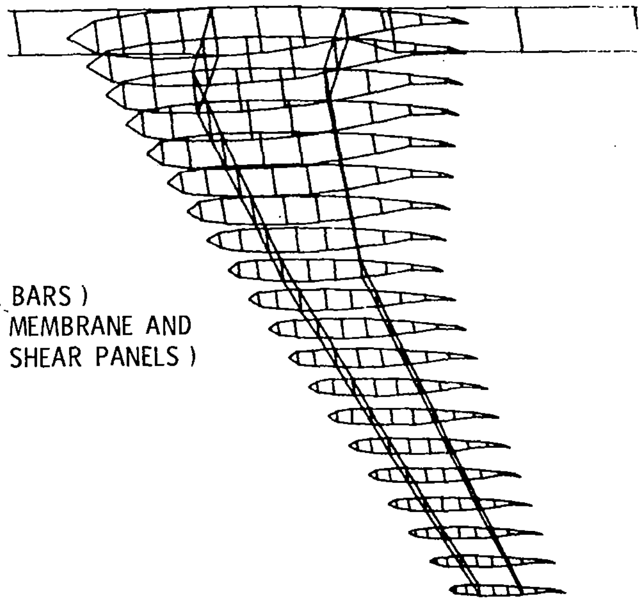


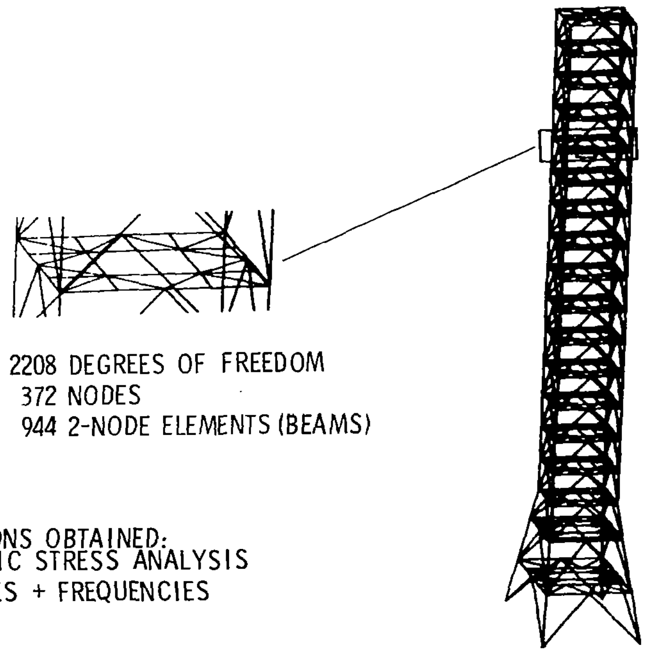
Fig. 3 Interactive terminal access to minicomputer and mainframe computer.



1120 DEGREES OF FREEDOM
450 NODES
1102 ELEMENTS
{728 2-NODE ELEMENTS (BARS)
{374 4-NODE ELEMENTS (MEMBRANE AND SHEAR PANELS)

SOLUTIONS OBTAINED:
STATIC STRESS ANALYSIS

Fig. 4 Plot (bars shown) of finite element wing model.



2208 DEGREES OF FREEDOM
372 NODES
944 2-NODE ELEMENTS (BEAMS)

SOLUTIONS OBTAINED:
STATIC STRESS ANALYSIS
MODES + FREQUENCIES

Fig. 5 Plot of launch umbilical tower model.

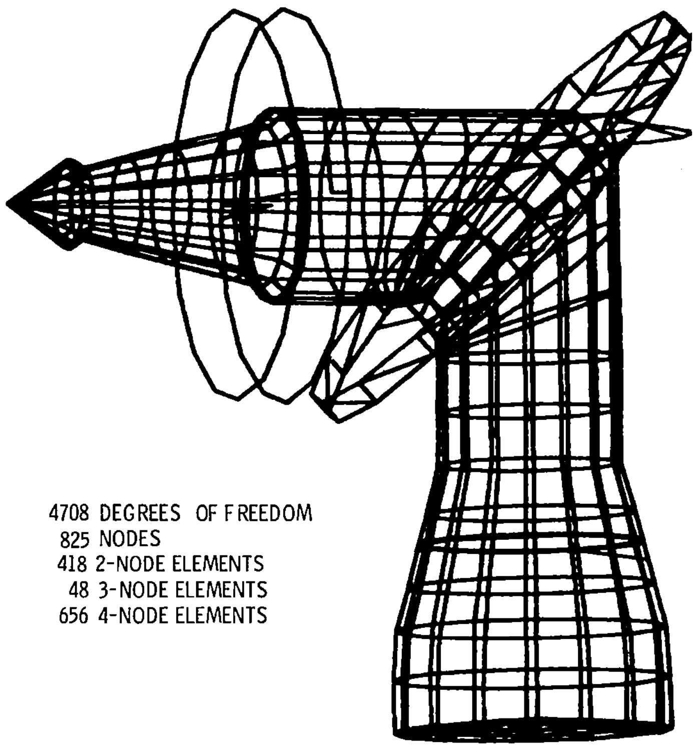


Fig. 6 Plot of National Transonic Facility (cryogenic wind tunnel) model

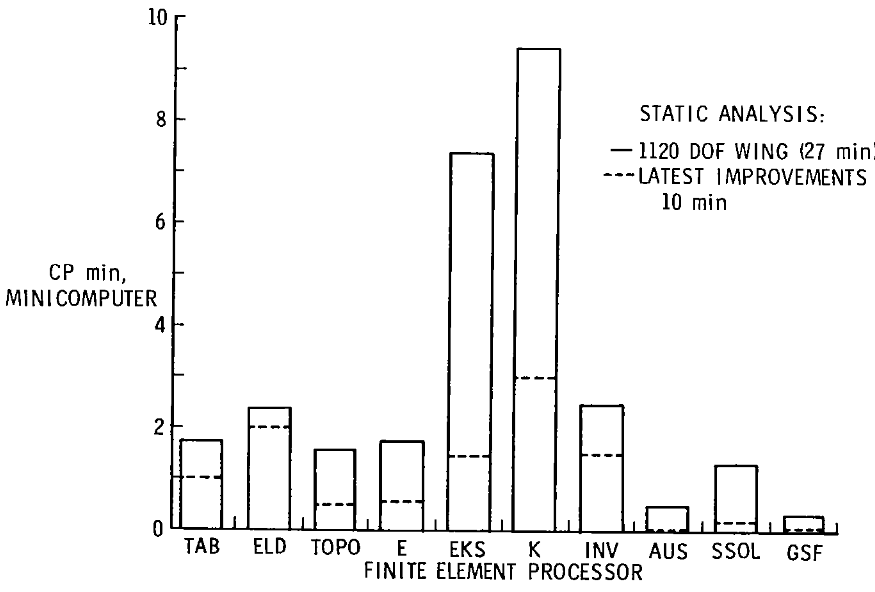


Fig. 7 Finite element processor time distribution (wing model).

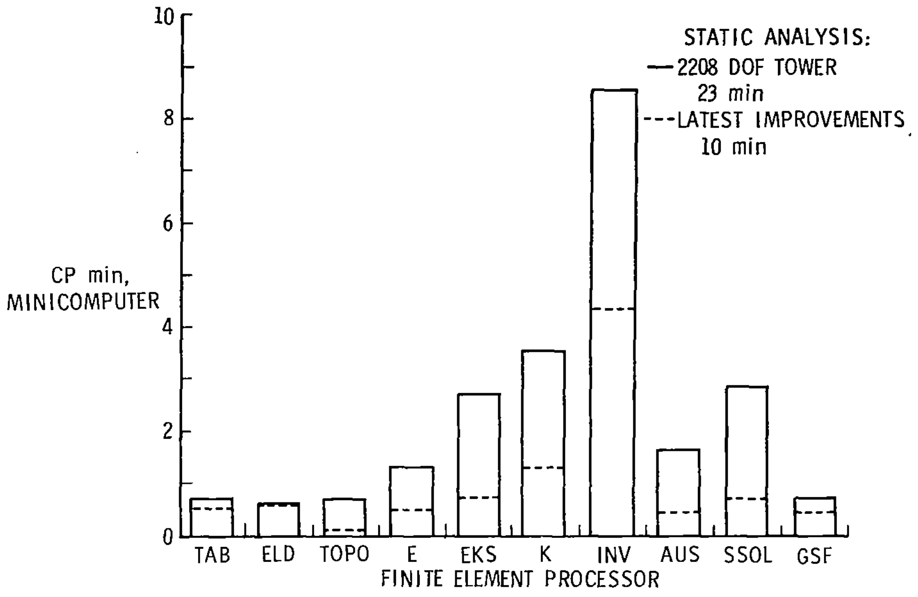


Fig. 8 Finite element processor time distribution (tower model).

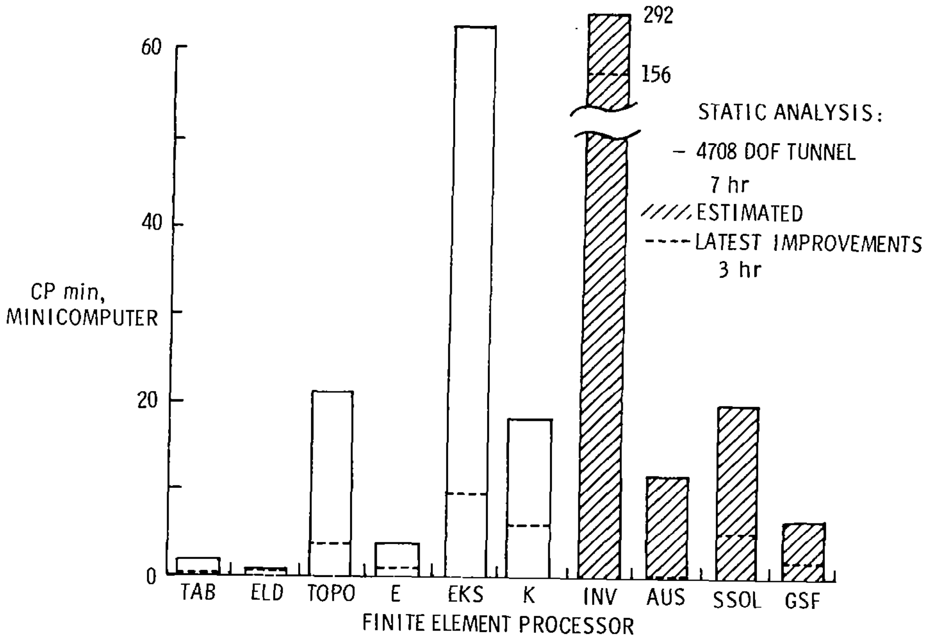


Fig. 9 Finite element processor time distribution (tunnel model).

FINITE ELEMENT MESH CONFIGURATIONS USING ISOENERGETICS AND EQUALIZED ENERGY LEVELS

David J. Turcke
Queen's University, Kingston, Ontario, Canada

SUMMARY

The concept of equalizing energy levels was shown to be a viable additional criterion in assisting the analyst in laying out finite element grids according to the isoenergetic discretization technique. In addition it is clear that similar problems specifically with respect to mesh refinement in piecewise approximation theory are being researched.

Further, common criteria in both areas are being developed in an effort to cope with the question of discretization for improved piecewise approximations.

INTRODUCTION

One of the most fundamental decisions which a finite element analyst must make is how to discretize the continuum. Research on optimum mesh configurations has been primarily based on the minimization of the potential energy functional with respect to both the nodal displacements and nodal co-ordinates [refs. 1,2,3,4,5]. This leads to the following set of non-linear equations

$$[k]\ \{\Delta\} - \{F\} = 0 \tag{1}$$

and $$\tfrac{1}{2}\ \{\Delta\}^T \frac{\partial [k]}{\partial x_j} \{\Delta\} - \frac{\partial \{F\}^T}{\partial x_j}\ \{\Delta\} = 0 \tag{2}$$

where $\{\Delta\}$ is in general a column vector of unknown values of the displacement field and its derivatives at the nodal points

$[k]$ is the stiffness matrix

$\{F\}$ is a column vector of nodal forces

and x_j is the co-ordinate of node j

A solution of these equations will yield the best possible approximate solution for a given number of elements and initial topology. However, the computational effort in solving these equations is excessive and there is no guarantee that the global minimum has been achieved. As a result two alternative approaches have been employed.

One procedure is to approximately satisfy equation (2), that is until the left hand side is less than some prescribed tolerance ε [refs. 4,5]. This still involves the calculation of the derivatives of [k] and {F} with respect to the nodal co-ordinates.

The other approach is to develop a set of guidelines such that "near optimum" grids can be obtained [refs. 1,2,3,6,7] without explicitly dealing with equation (2). The fundamental concept in this method is based upon the observation that optimal grids align themselves along lines of constant strain energy density - so-called isoenergetics [refs. 2,8]. These contours provide the stress analyst with a picture of the location of stress concentrations, relative density of elements to be allocated to various regions and proper element orientation with respect to the critical stress gradients. This latter aspect is of major importance for the so-called simplex elements which are available in almost all finite element software packages and are used in complex non-linear analyses to reduce the computational costs.

The primary objective of this paper is to present an interactive isoenergetic discretization technique which incorporates the concept of equalizing energy levels as an additional criterion in assisting the analyst in laying out finite element grids. In addition, a mesh refinement approach employing a similar criterion in the piecewise approximate solutions of differential equations is presented. This suggests a possible refinement strategy in finite element analysis.

ISOENERGETIC DISCRETIZATION PROCEDURE

This procedure for generating efficient mesh configurations incorporates not only the necessary topological considerations but also the response characteristics of the problem. The steps involved are as follows:

Step (1) An initial course grid is sketched on a digitizer or produced by automatic mesh generation schemes whichever is appropriate.

Step (2) The grid is analyzed and the strain energy density values are calculated at the nodes and/or the integrating points if applicable. The number of contours to be plotted is selected by the user based upon the maximum energy density differences. Subsequently, the isoenergetics and initial grid are superimposed or displayed separately on the CRT screen.

Step (3) Given this display of information the stress analyst can

(a) Modify the initial grid such that the element gradation reflects the density of isoenergetics and element orientation, in the case of triangles, is directed along lines orthogonal to the contours, that is, in the direction of the maximum strain energy density gradient.

or (b) Select an arbitrary number of isoenergetic contours within which a new mesh can be defined consistent with the above characteristics of the isoenergetics.

Both these operations can be done at a display terminal or with a hard copy of the isoenergetics on a digitizing table.

Clearly, steps (2) and (3) can be repeated until two successive configurations, whose modification usually only involves a shifting of the isoenergetics, provide little or no improvement in the strain energy content.

EQUALIZING ENERGY LEVELS

One of the most interesting observations during the study of optimal grid configurations was that for a class of one dimensional problems the energy content in all the elements was equal [refs. 3,1]. This notion of equal energy levels for optimal meshes was arrived at independently by Prager [ref. 9]. Subsequently Masur (private communication) examined bars under varying axial forces and beams subjected to lateral loads, both with arbitrary variations of the cross section. He concluded that it appears to be futile to search for a universally valid optimality criterion in terms of the average energy in the entire element.

However, for all the various one and two dimensional problems studied, there was a very definite trend for the total energy content between the isoenergetic contour levels in the high strain gradient regions of the optimal grid to be equalized. Figure 1 shows clearly that for the special case of a linearly varying tapered bar subjected to a constant load P the optimized grid yields an equal amount of energy in each element. This is not exactly true for the parabolic taper; however there is clearly a tendency for the optimal element energy contents to be equalized relative to the unoptimized grids. Figures 2 and 3 demonstrate the same general trend, but in these two dimensional plane stress examples rather than dealing with a specific element the energy content between contours is calculated and then compared to the other bands of elements or contour levels of elements. The calculation of the energy content between contours is simply found by summing the product of the element strain energy density and the corresponding volume for each element within the particular contour. The results show that there is a general trend to equalizing the energy content in each contour level of the optimal grid relative to the unoptimized mesh.

It should be noted that this equalization effect is most noticeable in the immediate vicinity of the high strain gradients both for the one and two dimensional problems.

This result has proven extremely useful in providing an additional criterion for evaluating the effectiveness of the already established isoenergetic discretization procedure.

Several examples were re-examined after using the above mentioned isoenergetic discretization procedure to see if the near optimum grids had relatively equalized energy levels. A typical result is shown in figure 4 which clearly indicates the above trend. Consequently, this observation was introduced into the discretization procedure as an additional indicator along with the shifting of the isoenergetic contours and the variation in the total strain energy content to assist the analyst in establishing when further mesh modifications are required.

SIMILARITIES IN APPROXIMATION THEORY

In reviewing the mathematical literature dealing with splines having variable knots the following interesting results were found.

In a paper by De Boor [ref. 10] on the topic of good approximations by splines with variable knots it is suggested that in approximating a function f by elements of S_N^k, the N knots t_1, t_2 ... t_N should be chosen such that

$$\int_{t_i}^{t_{i+1}} | f^k (r) |^{1/k} dr$$

is approximately constant as a function of i, where

f is the function being approximated or a solution of a boundary value problem

S_N^k is a spline of order k (or, degree < k) with N knots

This concept has been tried by Dodson [ref. 11] in a scheme for the adaptive solution of ordinary differential equations. The procedure is as follows: using a current piecewise polynomial approximation of order less than k to the solution f, a piecewise constant approximation g to $f^{(k)}$ is assumed. Then a new knot is selected so as to equalize

$$\int | g(r) |^{1/k} dr$$

over the subintervals.

Subsequently Sewell [ref. 12] extended Dodson's results to piecewise polynomial approximations in two dimensions. In particular Dodson gave an algorithm for the automatic partitioning of an interval which provided the basic ideas for an automatic triangulation algorithm proposed by Sewell. From an error bound theorem developed in this work it is recommended that for a good approximation the integral

$$\iint \max_{i+j=k} \left| D_x^{\,i} D_y^{\,j} f \right|^{\sigma} dA$$

should be distributed as evenly as possible among the grid of triangles, where

$$\sigma = \frac{2}{k+2}$$

Throughout the presentation a polynomial of degree less than k is assumed in each triangle; consequently an estimate of the above integrand must be made. This is accomplished by approximating each kth derivative within a triangle by a piecewise constant function G_j.

As a result the following elemental product

$$G_j^{\sigma} \; A_j$$

should be distributed as evenly as possible among the elements having areas A_j. In order to achieve this a continual mesh refinement process was carried out by subdividing those elements with the highest values of $G_j^{\sigma} A_j$. This clearly ensures the necessary grid refinement near the high gradients or singularities of the solution or function being approximated.

Referring back to the notion of equalizing energy levels, a similar criterion has been suggested in this paper for mesh modification; specifically the energy within each contour should be distributed as evenly as possible between successive contour levels in the vicinity of the high gradients of the solution field; that is,

$$U_1 \simeq U_2 \simeq U_3 \ldots$$

where $$U_m = \sum_j U^0_{m\,j} \; V_{m\,j}$$

in which $U^0_{m\,j}$ is the piecewise constant strain energy density function of element j in level m

$V_{m\,j}$ is the volume of element j in level m

and U_m is the total energy in contour level m

An important point to note here is that the above criterion is concerned with mesh modification rather than mesh refinement which Dodson and Sewell presented.

Further, from the research studied to date this criterion is best applied to contour levels rather than an examination of the individual elements. This suggests that future mesh modification and refinement schemes should consider bands of elements to be repositioned rather than individual finite elements.

CONCLUSIONS

The concept of equalizing energy levels was shown to be a viable additional criterion in assisting the analyst in laying out finite element grids according to the isoenergetic discretization technique. In addition it is clear that similar problems specifically with respect to mesh refinement in piecewise approximation theory are being researched.

Further, common criteria in both areas are being developed in an effort to cope with the question of discretization for improved piecewise approximations.

REFERENCES

1. Turcke, D.J. and McNeice, G.M., "A Variational Approach to Grid Optimization in the Finite Element Method", Conf. on Variational Methods in Engineering, Southampton University, England, 1972.

2. Turcke, D.J. and McNeice, G.M., "Guidelines for Selecting Finite Element Grids Based on an Optimization Study", Int. J. of Computers and Structures, Vol. 4, 1974, pp. 499-519.

3. Turcke, D.J., "Optimum Mesh Configurations in the Finite Element Method", Doctoral Dissertation Civil Engineering, University of Waterloo, Waterloo, Canada, 1974.

4. Carroll, W.E. and Barker, R.M., "A Theorem for Optimum Finite Element Idealizations", Int. J. of Solids and Structures, Vol. 9, 1973.

5. Carroll, W.E., "Ramifications of Optimum Idealization Geometry in Discrete Element Analysis", Proc. World Cong. on Finite Elements in Struct. Mech., Bournemouth, England, 1975.

6. Turcke, D.J., "On Optimum Finite Element Grid Configurations", AIAA J., Vol. 14, No. 2, Feb., 1976, pp. 264-265.

7. Tang, J. and Turcke, D.J., "Characteristics of Optimal Grids", Comp. Methods in Appl. Mech. and Eng., Vol. 11, No. 1, April, 1977, pp. 31-37.

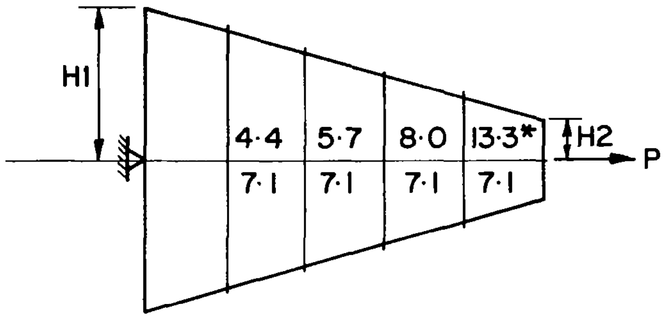
8. Oliveira, E.R. Arantes, "Optimization of Finite Element Solutions", Proc. of the Third Conf. on Matrix Methods in Struct. Mech., Wright-Patterson Air Force Base, Ohio, Oct., 1971, pp. 750-769.

9. Prager, W., "A Note on the Optimal Choice of Finite Element Grids", Comp. Meths. Appl. Mec. Eng. 6, 1975.

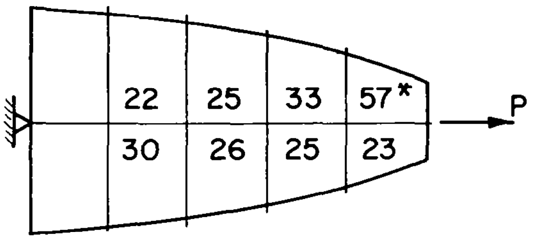
10. De Boor, C., "Good Approximation by Splines with Variable Knots", ISNM Vol. 21, Springer-Verlag, Berlin, 1974.

11. Dodson, D.S., "Optimal Order Approximation by Polynomial Spline Functions", Ph.D. Thesis, Purdue Univ., Lafayette, Indiana, Aug., 1972.

12. Sewell, E.G., "Automatic Generation of Triangulations for Piecewise Polynomial Approximation", Ph.D. Thesis, Purdue University, Lafayette, Indiana, Dec. 1972.



LINEAR TAPER H1/H2=6/1



PARABOLIC TAPER K = 0·8

* UNOPTIMIZED STRAIN ENERGY LEVEL VALUES

Figure 1.- Strain energy levels for 2 one-dimensional problems.

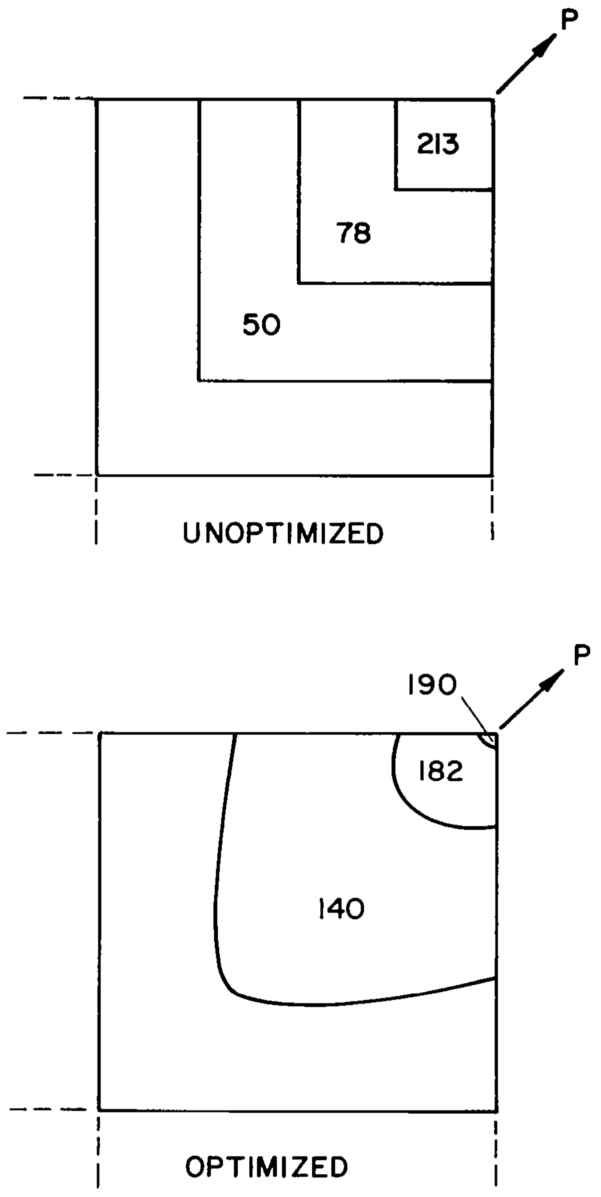


Figure 2.- Strain energy levels for a two-dimensional corner load proglem.

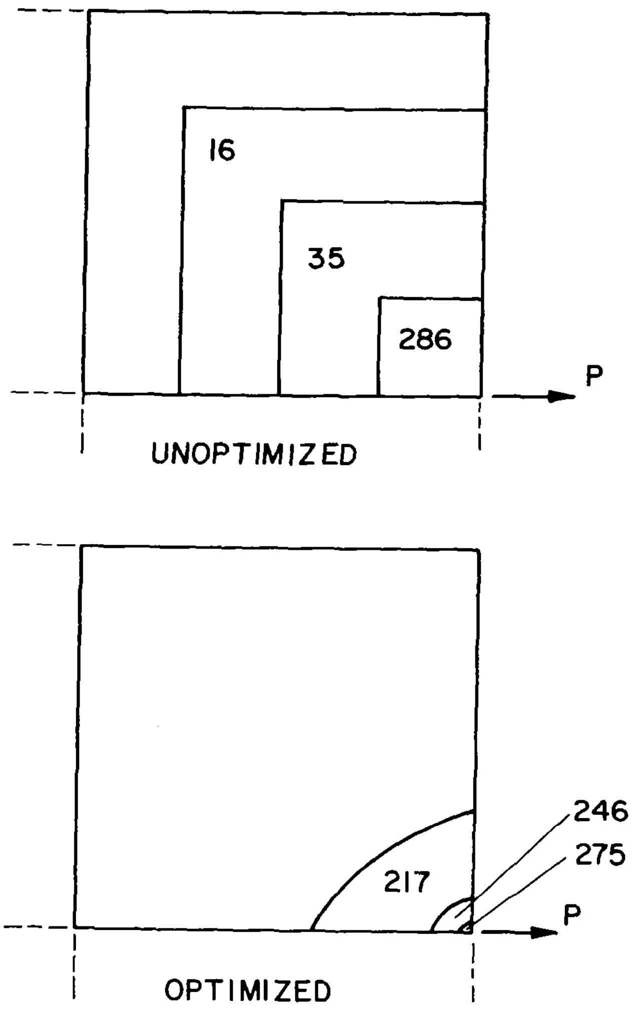


Figure 3.- Strain energy levels for a two-dimensional mid-edge load problem.

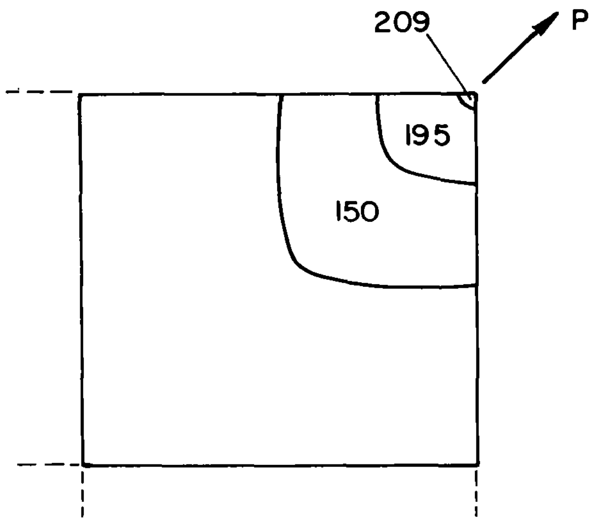


Figure 4.- Strain energy levels in a near optimized grid using isoenergetics.

DYNAMIC PROGRAMMING AND DIRECT ITERATION FOR THE OPTIMUM DESIGN OF SKELETAL TOWERS

G.C. Howell and W.S. Doyle

University of Cape Town, Rondebosch, South Africa

ABSTRACT

A computer technique is proposed for a simple practical method of automatically designing tower structures. Dynamic programming is used to find the optimum geometric configuration of the structural members, while the member sizes are proportioned by direct iteration.

Tower structures are particularly suited to this method of automatic design since the rapidity of the analysis and design depends primarily upon substructuring. Substructuring of towers is comparatively simple because interaction between adjacent substructures can be simulated with reasonable accuracy. Typical examples are presented to illustrate the method.

INTRODUCTION

Dynamic programming has long been recognised as an extremely powerful optimization technique, particularly for problems of a discontinuous nature. High-dimensional problems, however, result in a large amount of computation, but this can be reduced by a successive approximation method.

A 3-dimensional Dynamic Programming Successive Approximation technique is used here to obtain an optimum (least weight) geometrical configuration for the design of tower structures. Concurrently, a simple direct iteration procedure is used to select the optimum member sizes from any list of section properties provided.

If the structure were to be designed as a whole, a change of geometric configuration would need a re-analysis. This means that a large amount of computation would be required.

Substructuring has been introduced to reduce the overall problem into smaller stages so that the analysis can be performed more rapidly.

If any sections recommended by the computer are not desirable for practical reasons the re-input of a complete new set of data is unnecessary. Only the list(s) of sections and the data relating to section types need be changed.

Three examples of tower structures are given in this paper:

1. A sample plane-truss tower
2. A comparison of the optimum weight of a triangular tower referred to by Kuzmanovic, Willems and Thomas (ref. 1)
3. A typical transmission tower used in South Africa

DYNAMIC PROGRAMMING - BASIC CONCEPTS

The computational effort required to find all the possible solutions for large problems can become unmanageable without the use of Dynamic Programming. This technique bypasses this problem by considering the possible decisions to be made at each stage of the solution.

Dynamic Programming can be described as a technique for methodically selecting an optimum solution of a multi-stage decision problem. This mathematical technique can be used for problems where a sequence of decisions are dependent upon one another and each decision influences the system's response to future decisions. A set of solutions can be categorized so that one may be judged to be better than another in some pre-defined manner.

In a sequence of decisions, the current state of the sequence is assessed as $f(x(k-1))$ and the succeeding one as $f(x(k))$. Without considering the whole chain of past and future decisions, except that they contribute to $f(x(k-1))$, the best decision can be found from:

$$f(x(k)) = \min[t(x(k); x(k-1)) + f(x(k-1))]$$

where t is the assessment between stages $k-1$ and k and
x is a state variable

Dynamic Programming is based on a repeated use of this idea.

A simple network problem posed by Bellman and Dreyfus (refs. 2 & 3) and discussed by Palmer (ref. 9) demonstrates the process excellently. In the course of the solution of this problem, two central ideas are used. The first is the idea of imbedding; this means that the overall optimization problem consists of a number of smaller problems imbedded within the whole, which can be solved independently. In the second idea, an optimum solution can be found from a sequence of decisions by imagining that the final solution is broken up into a series of simpler decisions.

Problems of this type become very tedious when the order of the state variables $(x(k))$ becomes large. For example, various state variable vectors (denoted $x_n(k)$) may be present at any particular stage k of the calculation, which complicates the matter. For this reason, the dimension (n) of the problem is decreased to a single variable vector by the use of the Dynamic Programming Successive Approximation (DPSA) technique.

The geometric design of a tower structure can be degenerated into a series of simple decision processes as prescribed by the DPSA technique by using substructuring.

SUBSTRUCTURING

The benefit of substructuring in the design of towers is that the optimum section sizes can be determined rapidly within each substructure. The interaction between adjacent substructures is comparatively simple and can be simulated with reasonable ease which makes this technique particularly attractive.

The geometry at the interface of each substructure is uniquely defined by the coordinates of the member ends which are 'cut' at the interfaces. Hence, in Figure 1, the coordinates at the right hand side of substructure 1 are defined by x_o, y_o; x_1, y_1 and for substructure 2 by x_1, y_1; x_2, y_2 and so on for the other substructures.

The assumption is made that the forces transferred from one substructure to the next can be calculated from statics. For example, the interaction forces at the interface between substructures 1 and 2 are found by applying equivalent loads and moments at the central point A; i.e., the vertical force P_v is resolved into a load P_v and a moment $P_v \times h$, while the horizontal force P_h is resolved into a load P_h and a moment $P_h \times (y_4 - y_1)$.

Each pin-jointed substructure is analysed by the displacement method, which requires the equivalent loads and moments at A to be applied as point loads at the interface nodes (x_1, y_1) and $(-x_1, y_1)$. The substructures are now analysed and designed independently assuming that the lower interface nodes are constrained. The member sizes are selected by a simple direct iteration procedure.

DIRECT ITERATION

A list of sections is provided so that the actual member sizes required in each substructure can be selected automatically by the direct iteration procedure. The calculated stresses are compared with permissible stresses so that the ratio between them is a minimum. The substructure is reanalysed each time the member sizes are revised until the results from successive iterations are identical.

THE DYNAMIC PROGRAMMING SUCCESSIVE APPROXIMATION (DPSA) TECHNIQUE

The coordinates of the interface nodes in Figure 1 can be changed at any stage and the members designed accordingly by the direct iteration method.

Consequently, the weight of each configuration of members in all substructures can be calculated. The optimum solution is then the combination of possible configurations which results in the least weight of the total structure. The dynamic programming technique sets about this calculation in an organized methodical way.

Let us consider a tower similar to that in Figure 1. The interface nodes, which define the configuration of the tower, can be positioned in space by a set of three-dimensional coordinate values. Let the structure be symmetrical about the XZ and YZ planes; therefore a node placed in the first octant will define the shape of the tower at that level. Let the coordinates of the interface node be $(x_1(k), x_2(k), x_3(k))$. This corresponds to the x, y and z coordinates of the primary node of substructure 1, where k also denotes the upper interface level of that substructure and k = 0 represents the ground level. The n-dimensional DPSA method is therefore well suited to structures of this type, where n = 3. For example, at level k = 2 some possible values of the state variables are shown in Table 1. The control variables $u_n(k)$ shown correspond to the identification number in each set.

The values of u(k) (i.e. $u_1(k)$, $u_2(k)$, $u_3(k)$) and hence the values of $x_1(k)$, $x_2(k)$, $x_3(k)$ at each level k must be found so that the overall weight of the structure is a minimum.

The DPSA method requires an initial solution to the problem. The average value of the state variables at each interface is a suitable initial solution. To proceed, only one state variable is altered, while the remainder stay at their initial values. In this way a single-dimensional dynamic programming procedure with respect to this variable is carried out. The process is then continued, the new value of the first variable is retained and one of the other state variables is altered. All the variables are processed in this way; the first cycle is complete when all the state variables have been altered. Subsequent cycles follow the same procedure as above. The DPSA method has converged to its optimum solution when no weight difference is recorded between successive cycles. This method is found to converge rapidly to an optimum weight solution. The structure comprises a number of substructures. The geometric configuration between interfaces is regarded as a substructure.

The direct iteration method is used to find the most satisfactory structural design and hence the weight for every geometric configuration of each substructure. A least weight path is followed through all the substructures to determine the optimum configuration of the total structure. The explanation of the DPSA method can be simplified by the following elementary example.

Example: Consider a simple 2-dimensional tower which consists of 2 substructures - Figure 2.

Let the state variables x(1) and x(2) vary in three steps on each side of the vertical axis of symmetry and the other state variables i.e. y(k) be kept constant. In addition, let the state variable x(k) at k = 0 be constrained to a single value. The accumulated cost (or weight) of the substructures up to level k for each position of x can be denoted by:

$$C_i(k)$$

where i denotes the position of x(k) on interface k.
Let the cost of a single substructure k be denoted by:

$$t_{ij}(k)$$

where i and j denote the positions of the state variables x(k) for substructure k at the upper and lower interfaces respectively.

Figure 3 shows all the possible geometrical configurations within the constraints given.

The costs of each of the 3 configurations in the first substructure, due to the varying values of x(1), are calculated by:

$$C_1(1) = t_{11}(1)$$
$$C_2(1) = t_{21}(1)$$
$$C_3(1) = t_{31}(1)$$

and are shown in Figure 4.

Similarly the values of $t_{ij}(2)$ can be obtained for the second substructure. The least accumulated weight at point i on interface k can be found from:

$$C_i(2) = \min[t_{ij}(2) + C_j(1)] \quad \text{for } j = 1 \text{ to } 3$$

The optimum configuration of the structure at level 2 is determined by choosing the least value of $C_i(2)$. The optimum path can then be traced back through the calculations to find the optimum configuration of the entire structure.

A computer program, which uses the ideas discussed above, has been written to design general tower structures. Three examples included are:
a. A simple 3-cell plane-truss tower. b. A 3-legged transmission tower.
c. A practical, rectangular plan transmission tower.

<u>Example 1.</u> 3-substructure plane-truss tower
The tower shown in Figure 5 supports two loads at the upper level of -15 kN in the y-direction and -10 kN in the x-direction at nodes 7 and 8 respectively. The possible dimensions at the 4 levels are, in x-direction:

	1	2	3	4	5
Base	1,6	1,8	2,0	2,2	2,4
Level 1	1,1	1,3	1,5	1,7	1,9
Level 2	0,8	0,9	1,0	1,1	1,2
Level 3	0,5	-	-	-	-

and in z-direction:

	1	2	3	4	5
Base	0,0	-	-	-	-
Level 1	1,8	1,9	2,0	2,1	2,2
Level 2	3,8	3,9	4,0	4,1	4,2
Level 3	6,0	-	-	-	-

Each substructure is to be designed using the following sections:
1. Upright member - channel sections. 2. Diagonal member - pipe sections
3. Horizontal member - angle sections.

Results: The final shape is shown in Figure 6.

Structural Design:

Substructure	Members	Size - mm	Type
1	a,d	100 x 50 x 6	Channel
	b,c	38,1ϕ x 2	Pipe
	e	40 x 40 x 3	Angle
2	f,i	100 x 50 x 6	Channel
	g,h	38,1ϕ x 2	Pipe
	j	25 x 25 x 3	Angle
3	k,n	76 x 38 x 5,1	Channel
	ℓ,m	48,5ϕ x 2	Pipe
	o	25 x 25 x 3	Angle

Total weight of structure, 1,461 kN; total computer time, 28,39 sec UNIVAC 1106.

Example 2. A three-legged transmission tower shown in Figure 7 is considered. It contains 56 nodes and 175 members. The loads used correspond to those used by Kuzmanovic et al. (ref. 1).

Load case 1: Basic wind free in transverse direction

Position	Load kN	Direction
A,B	26,7	- z
	8,9	- x
C	80,0	- z
	31,0	- x
D,E	15,6	- x
	40,0	- z

Load case 2: 0,707 Basic wind in the transverse direction
0,707 Basic wind in the longitudinal direction

Position	Load kN	Direction
A,B	16,0	- z
	5,4	- x
C	48,0	- z
	18,7	- x
D,E	9,4	- x
	11,6	+ y
	24,0	- z

Load case 3: No wind

Position	Load kN	Direction
A,B	32,0	- z
C	160,0	- z
D,E	80,0	- z

Member Groups: Table 2 shows the member groups used in each substructure.

Angle sizes: Table 3 shows the list of angles which were used to produce a structural design.

Results: The structural design for each load case is shown in Figure 8. The horizontal axis represents, in ascending order, the sizes of angle available for the design. The vertical scale represents the member groups within each substructure. Computer times and structure weights are:

Load case	No. of state variable positions at each level		Weight kN	Computer time UNIVAC 1106
	Horizontal (radial)	Vertical (elevation)		
1	5	1	23,256	6 min 27 sec
2	5	1	16,786	4 min 12,17 sec
3	5	1	17,428	3 min 36,89 sec

The optimum weight found by Kuzmanovic et al. (ref. 1) was 23,51 kN. The weights above compare favourably with this value.

Example 3. A practical transmission tower similar to those currently in use in South Africa is considered. It contains 115 nodes and 336 members. The shape of tower is shown in Figure 9. An equivalent set of loads have been calculated from those used in Example 2. Angle sizes used in the design are shown in Table 3.

Member Groups: Table 4 shows the member groups used in each substructure.

Results: The structural design for each load case is shown in Figure 10. Computer times and structural weights are:

Load case	No. of state variable positions at each level			Weight kN	Computer time UNIVAC 1106
	x-dir.	y-dir.	z-dir.		
1	3	3	1	57,877	46 min 04,252 sec
2	3	3	1	48,544	31 min 15,069 sec

CONCLUSION

This method of automatically determining the optimum geometric configuration and member sizes substantially reduces the effort involved in the design of tower structures. The dynamic programming successive approximations technique is effective for structures of this type, where substructuring provides the necessary static variables. The displacement method is admirably suited to the analysis of these structures. Direct iteration is the simplest method of selecting the optimum member sizes from any given list of section properties. Three examples have been discussed. In the 15 member plane-truss example considered as 3 substructures, the computational time taken on a UNIVAC 1106 is 28,39 seconds. A waisted tower shape of structure is the optimum geometric solution. Example 2 is a three-legged tower with 175 members and 56 nodes. The computational time with two state variables is 6 minutes 27 seconds. The weight compares favourably with that given in reference 11 by Kuzmanovic et al. Example 3 is a practical transmission tower similar to those currently used in South Africa. This structure consists of 336 members and 115 nodes and the computational time required for an optimum feasible solution is 46 minutes per load case.

REFERENCES

1. Kuzmanovic, B.O.; Willems, N.; and Thomas, F.M.: Automated Design of Three-Legged Steel Transmission Towers. Computer & Structures, vol. 7, 1977, pp. 171-182.

2. Bellman, R.E.: Dynamic Programming. Princeton University Press (Princeton, N.J.), 1957.

3. Bellman, R.E.; and Dreyfus, S.E.: Applied Dynamic Programming. Princeton University Press (Princeton, N.J.), 1962.

4. Palmer, A.C.: Dynamic Programming and Structural Optimization. Int. Symposium on Computers in Optimization of Structural Design (University of Wales, Swansea), January 1972.

BIBLIOGRAPHY

Larson, R.E.: A Survey of Dynamic Programming Computational Procedures. IEEE Transactions of Automatic Control, vol. AC-12, 1967, pp. 767-774.

Larson, R.E.; and Korsak, A.J.: A Dynamic Programming Successive Approximation Technique With Convergence Proofs. Automatica, vol. 6, 1970, pp. 245-260.

Larson, R.E.: State Increment Dynamic Programming. American Elsevier (New York), 1968.

Packia Raj, P.; and Olani Durrant, S.: Optimum Structural Design by Dynamic Programming. J. Struct. Div. ASCE, vol. 102, no. ST8, 1976, pp. 1575-1589.

Palmer, A. C.: Optimum Structure Design by Dynamic Programming. J. Struct. Div. ASCE, vol. 94, no. ST8, 1968, pp. 1887-1906.

Palmer, A. C.; and Sheppard, D.J.: Optimizing the Shape of Pin-Jointed Structures. Proc. of Inst. of Civ. Eng., vol. 47, 1960, pp. 363-376.

Sheppard, D.J.; and Palmer, A.C.: Optimal Design of Transmission Towers by Dynamic Programming. Computers & Structures, vol. 2, 1972, pp. 455-468.

Weaver, W.; and Patton, F.W.: Automated Design of Space Trusses. AISC Engineering Journal, vol. 5, 1968, pp. 26-36.

TABLE 1 - A POSSIBLE SET OF STATE AND CONTROL VARIABLE VECTORS

STATE VARIABLES $x_n(k)$			IDENTIFICATION NO $u_n(k)$ $n = 1$ to 3
$x_1(2)$	$x_2(2)$	$x_3(2)$	$u(2)$
0,8	0,7	0,9	1
0,9	0,75	0,95	2
1,0	0;8	1,0	3
1,1	0,85	1,05	4
1,2	0,9	1,1	5

TABLE 2 - MEMBER GROUPS

	Substructures					
Type	1	2	3	4	5	6
Main leg members, a	1					
Horizontal leg members, b	2					
Diagonal leg members, c	3					
Main leg members, d,g,j,m,p		1	1	1	1	1
Horizontal members, e,h,k,n,q		2	2	2	2	2
Diagonal members, f,i,ℓ,o,r		3	3	3	3	3

TABLE 3 - AVAILABLE ANGLE SIZES

Number	Size - mm	Number	Size - mm
1	25 x 25 x 3	11	80 x 80 x 8
2	30 x 30 x 3	12	90 x 90 x 8
3	40 x 40 x 3	13	100 x 100 x 8
4	45 x 45 x 3	14	100 x 100 x 10
5	45 x 45 x 5	15	120 x 120 x 10
6	50 x 50 x 5	16	120 x 120 x 12
7	60 x 60 x 5	17	150 x 150 x 12
8	60 x 60 x 6	18	150 x 150 x 18
9	70 x 70 x 6	19	200 x 200 x 16
10	80 x 80 x 6	20	200 x 200 x 24

TABLE 4 - MEMBER GROUPS

	Substructures			
Type	1	2	3	4
Main leg members, a,f,j	1	1	1	
Secondary leg members, b,g,k	2	2	2	
Leg horizontal members, c	3			
Diagonal members, d	4			
Interface members, e,i,m	5	4	4	
Leg bracing members, h,ℓ		3	3	
Main arm members, n				1
Secondary arm members, o				2
Bracing, p,q,r,s,t,u				3,4,5,6,7,8
Boom members, v,w				9,10

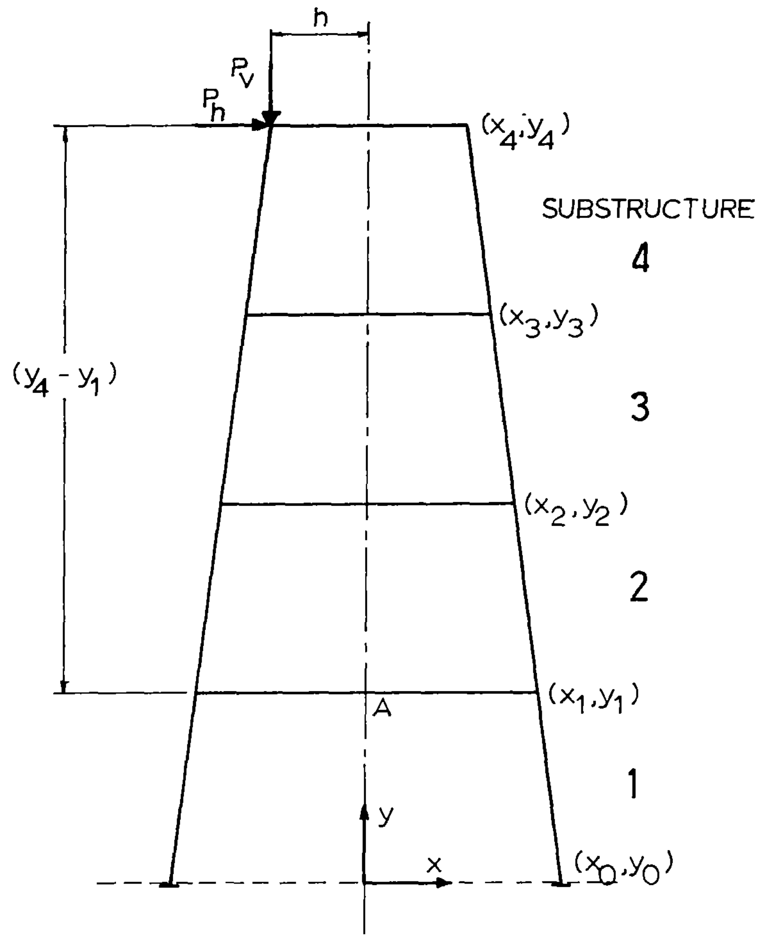


Figure 1.- Tower structure with substructures.

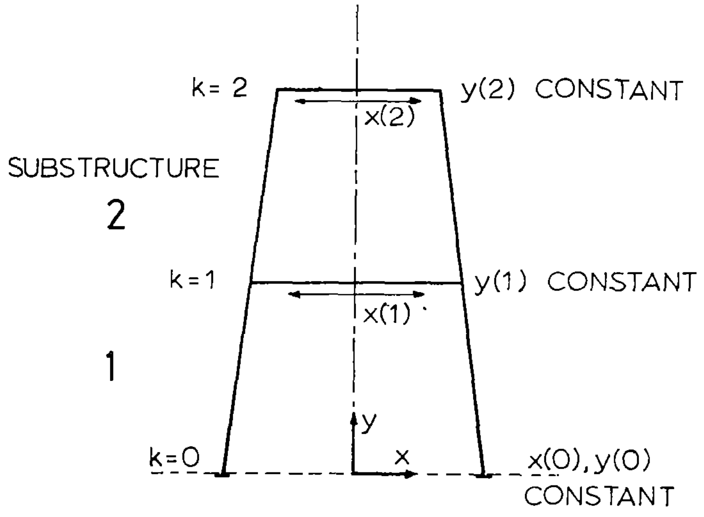


Figure 2.- Tower structure (example).

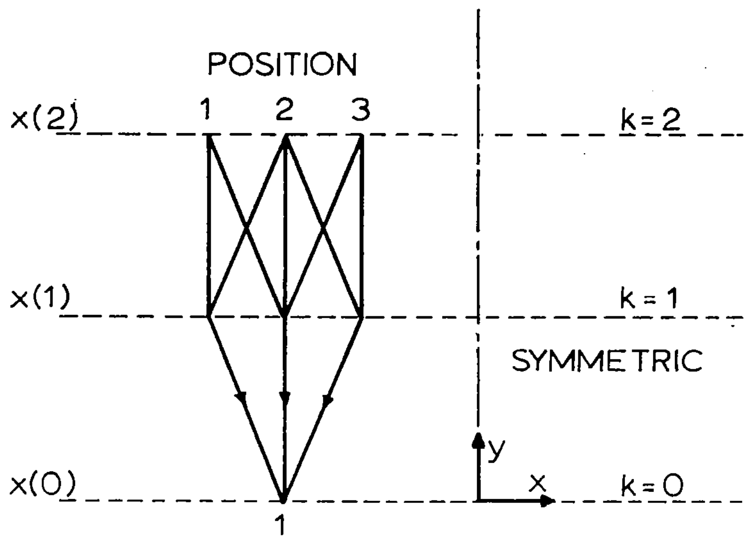


Figure 3.- Possible configurations.

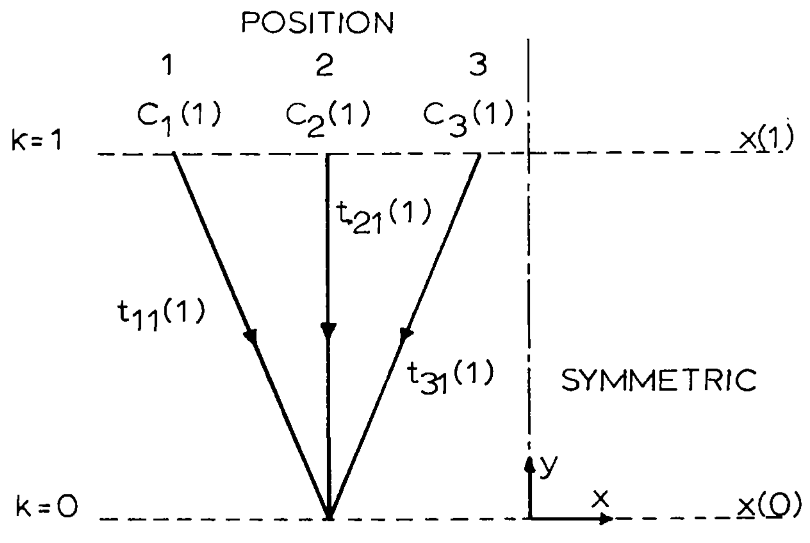


Figure 4.- Three cost values.

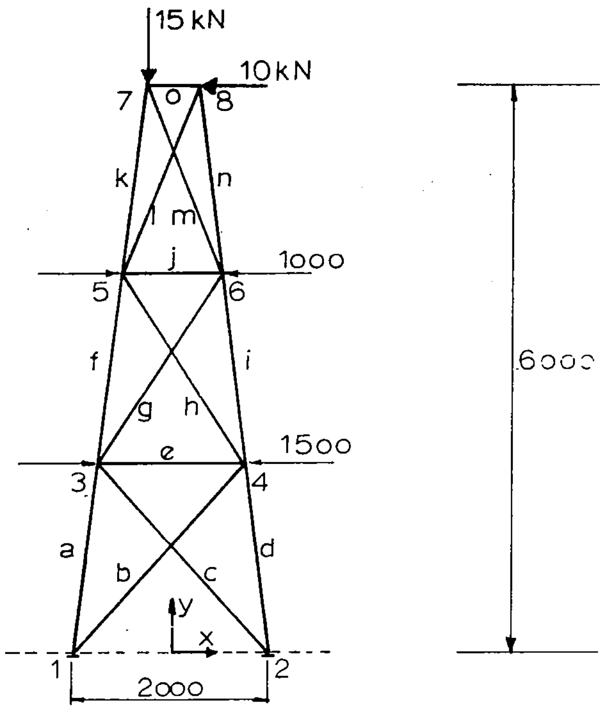


Figure 5.- Original tower shape.

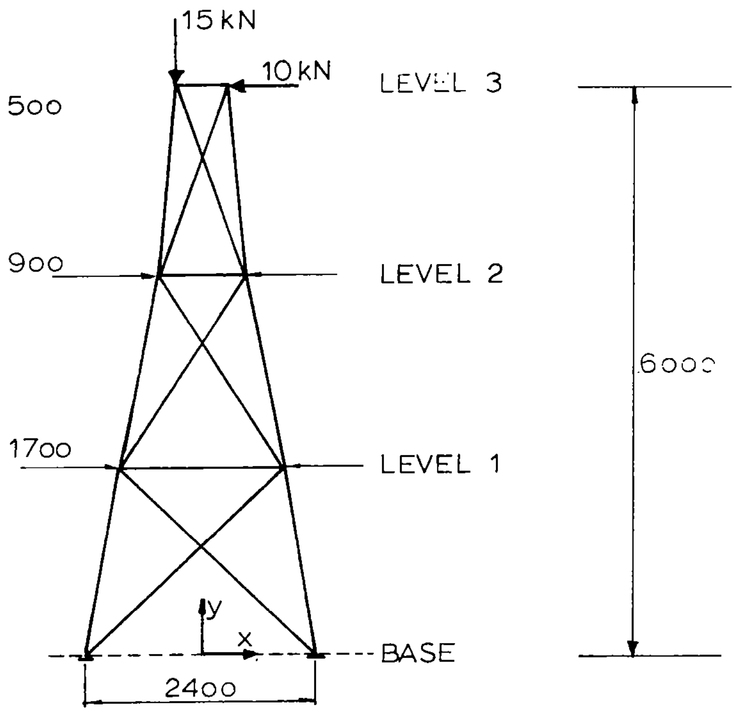


Figure 6.- Optimum tower shape.

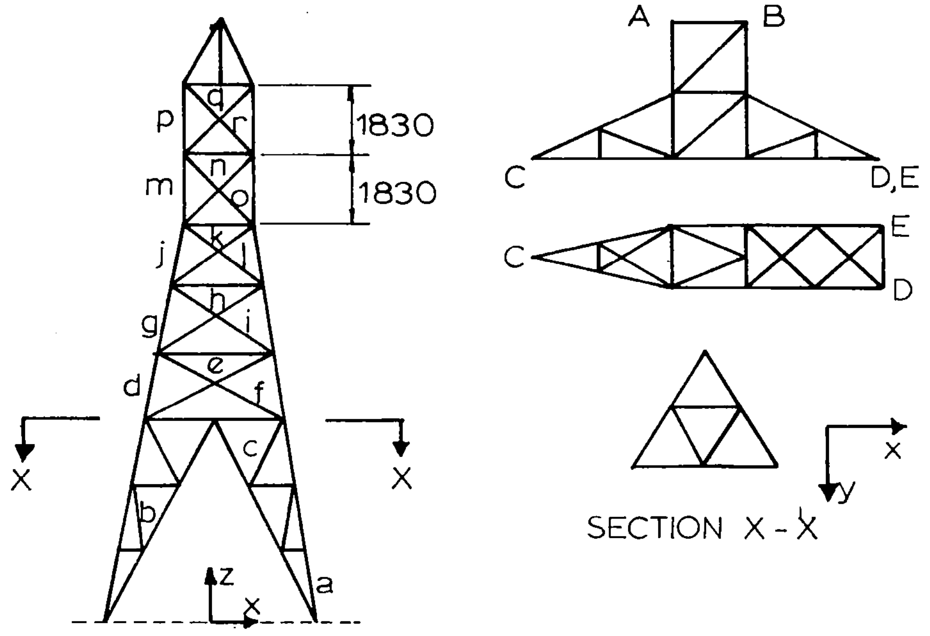


Figure 7.- Three-legged tower.

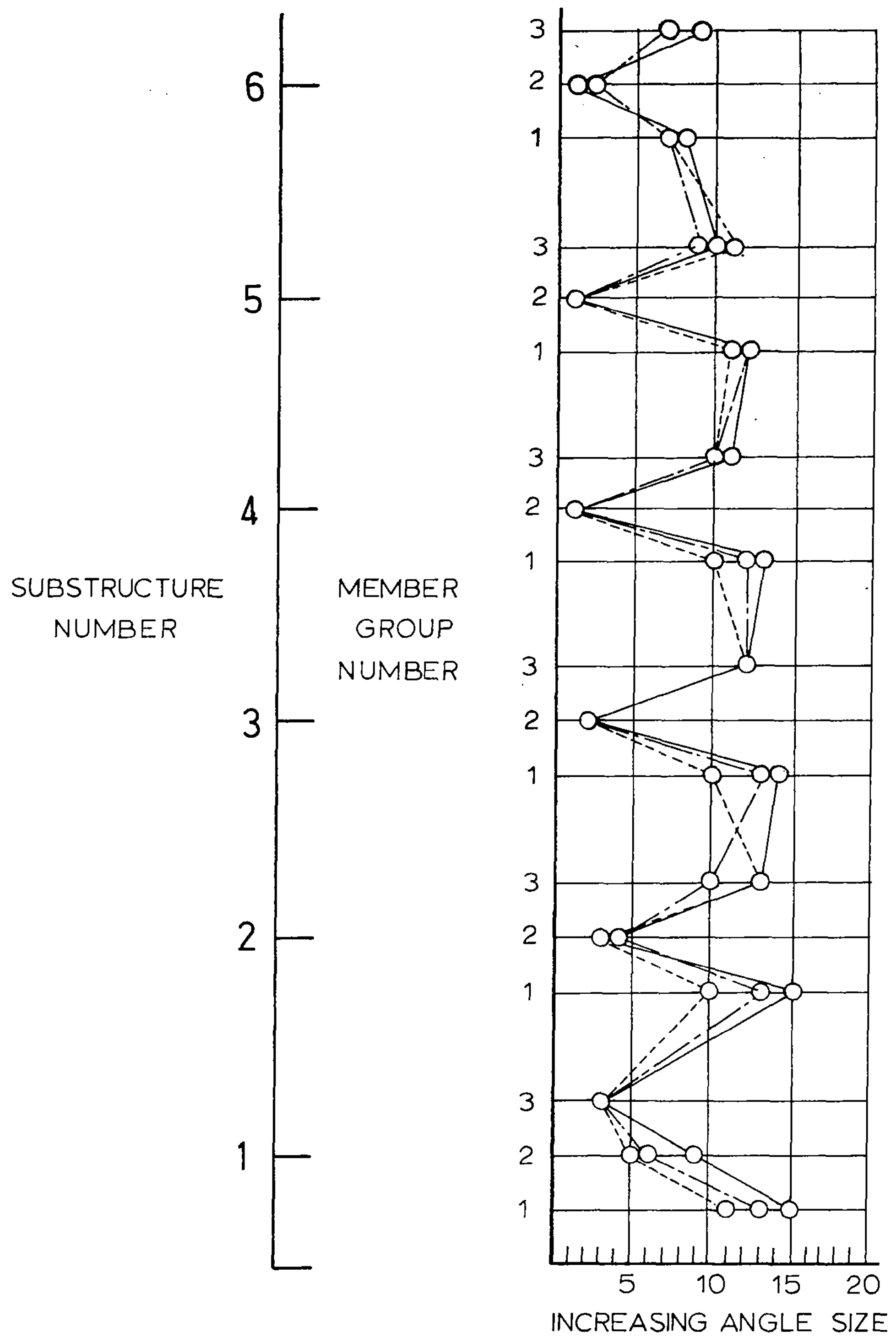


Figure 8.- Example 2: Structural design.

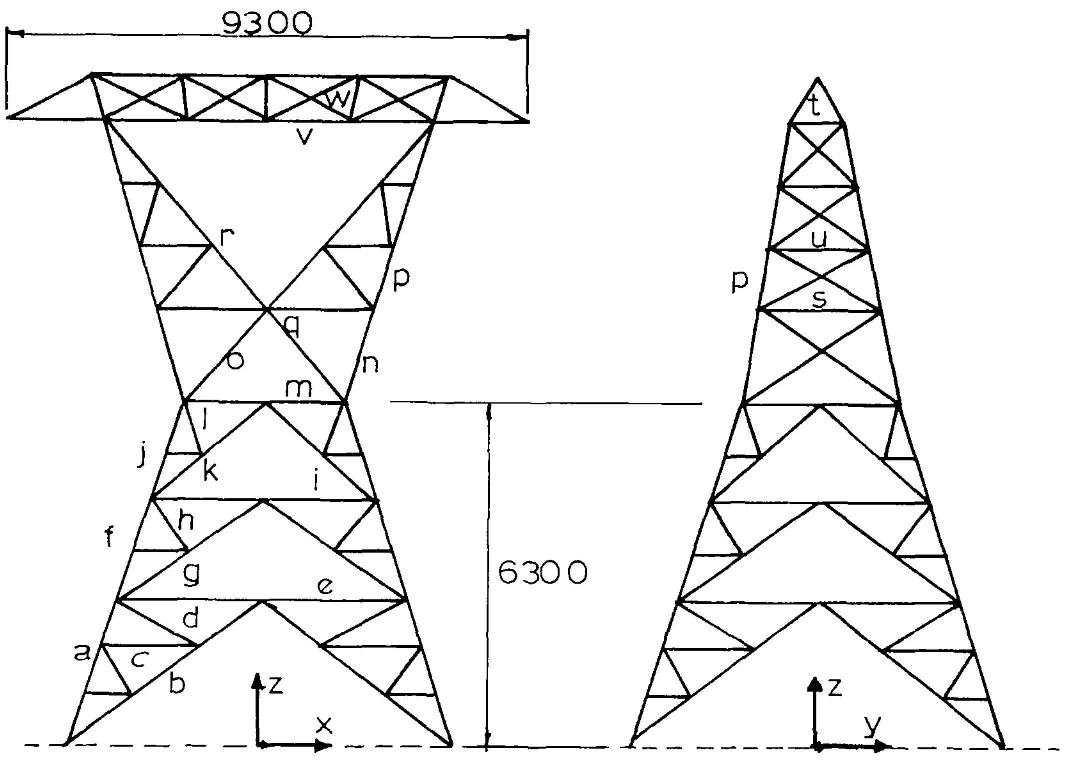


Figure 9.- Typical tower.

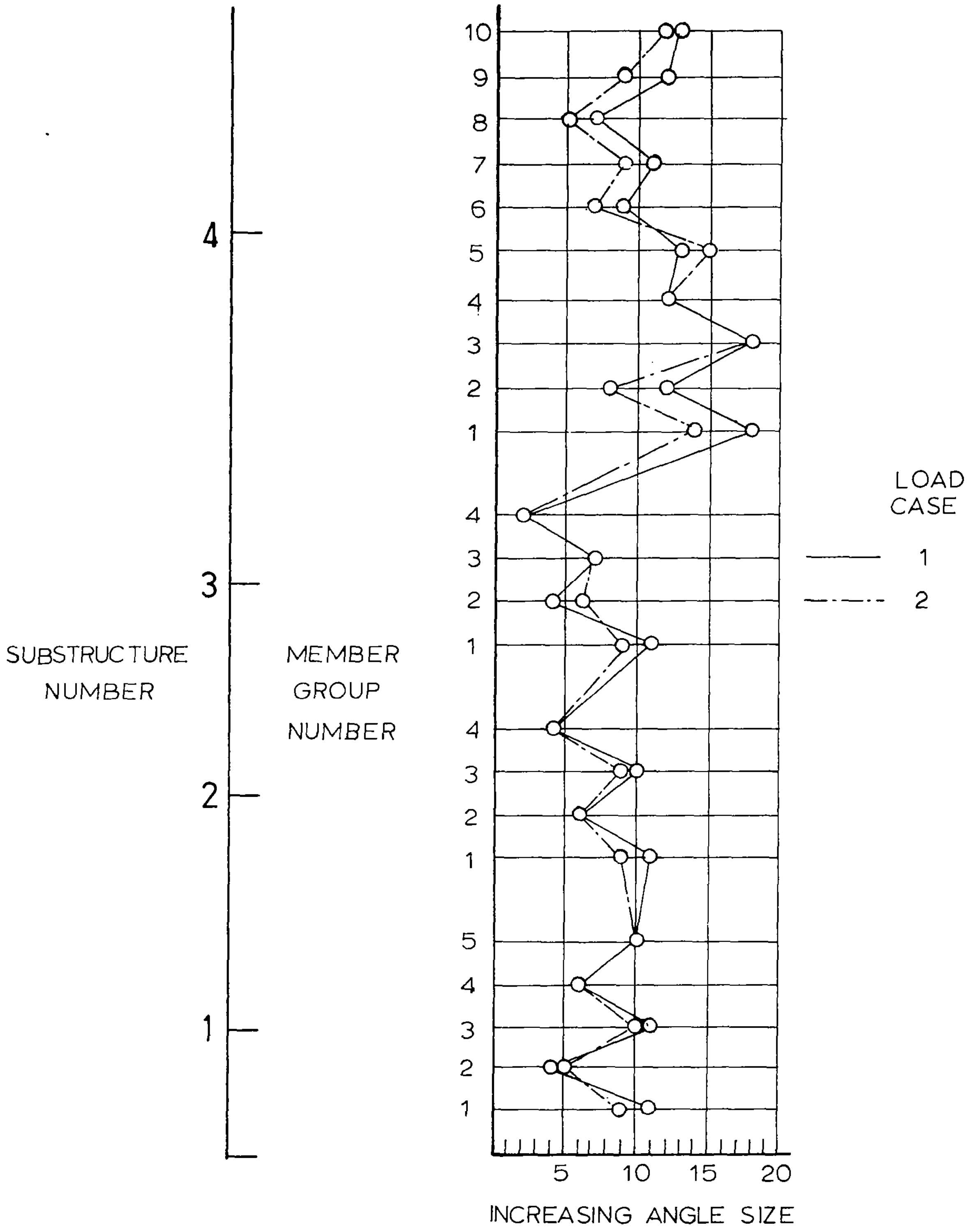


Figure 10.- Example 3: Structural design.

ADAPTATION OF A PROGRAM FOR NONLINEAR FINITE ELEMENT ANALYSIS TO THE CDC STAR 100 COMPUTER*

Allan B. Pifko
Grumman Aerospace Corporation

Patricia L. Ogilvie
Grumman Data Systems Corporation

SUMMARY

This paper discusses an effort to convert a nonlinear finite element program to the CDC STAR 100 pipeline computer. The program called DYCAST was developed under contract with NASA Langley Research Center for crash simulation of structures. Initial results with the STAR 100 computer indicate that significant gains in computation time are possible for operations on global arrays. However, for element level computations that do not lend themselves easily to long vector processing, the STAR 100 was slower than comparable scalar computers. On this basis it was concluded that in order for pipeline computers to impact the economic feasibility of large nonlinear analyses it is absolutely essential that algorithms be devised to improve the efficiency of element level computations.

INTRODUCTION

During the last decade, finite element methods, originally developed for linear structural analysis, have been extended to nonlinear problems. These developments have progressed to the point where the available methods are on a firm analytical basis and have been implemented in a number of general purpose computer codes. Since nonlinear analysis with the finite element method is essentially a successive linearization of a nonlinear problem, a given analysis completed in "n" steps can require a level of computation associated with "n" linear finite element solutions. Consequently, the problem facing the analyst today is not so much whether the solution to a given problem can be obtained, but whether the computer cost can be afforded. As analysts we can search for better algorithms based on the currently available computers in order to achieve near optimum computational efficiency while requesting that developers of computer systems constantly produce larger capacity faster machines. The purpose of this paper is to address this problem by describing an ongoing effort to convert a nonlinear finite element program to the CDC STAR 100 computer. This currently operational large fourth generation computer has a number of distinguishing features. From a user's vantage point, the two most important are the capability for vector arithmetic, e.g., pipeline processing, and the use of virtual memory. Vector arithmetic refers to the capability of performing computations on a string of numbers

*This work was supported by NASA Langley Research Center under Contract NAS-1-13148.

(vector) with a single vector instruction. More significantly, the computer central processing unit (CPU) is designed to exploit this type of calculation. For example, experience to date has indicated time savings as high as 35 to 1 over comparable CDC 6600 times for pure vector operations.

The virtual memory capability is a method of data management which gives the illusion that physical memory is larger than it really is. This is accomplished with both computer hardware and operating system software. In principle, the user can organize a program as if all the core necessary is available and the operating system "pages" data in and out of primary core. Thus, the process of overlaying code and using auxiliary storage devices to accommodate large quantities of data, common to programs implemented on third generation serial computers, can be assumed by the hardware and operating system software.

In order to exploit the features of the STAR 100, it is incumbent upon the user to design a solution strategy, if possible, that can utilize these features to the fullest. This can simply mean converting an existing algorithm to vector operations, or it can require devising an entirely new algorithm that can exploit pipeline processing.

An in-depth review of the features of the STAR 100, as well as its anticipated usefulness for finite element analysis, has been given by Noor and Fulton (ref. 1). A study of the feasibility of transferring NASTRAN® to STAR is found in ref. 2.

Work is currently underway by the authors to convert a program for non-linear transient dynamic analysis to the STAR 100 computer. This program, called DYCAST (DYnamic Crash Analysis of STructures), was developed under contract with NASA Langley Research Center, for crash simulation of structures. It is the purpose of this paper to discuss our user experiences, successes and pitfalls, during the course of this effort. In the remainder of this paper we first outline the theory on which DYCAST is based and identify potential areas where the STAR vector processing will have a significant effect. In order to gain experience in the usage of the STAR pipeline processing system and the associated extended subset of FORTRAN vector syntax, a pilot effort was initially begun to vectorize a program for the eigenvalue/eigenvector extraction of large matrices (ALARM - Automatic LArge Reduction of Matrices to tridiagonal form - ref. 3).

In doing this we not only had the advantage of working with a small program but one that was particularly amenable to vector processing. Results from this study as well as those from DYCAST are presented in the paper.

DYCAST FORMULATION

DYCAST is a finite element program for the nonlinear transient dynamic analysis of structures with particular emphasis on crashworthiness evaluation. It is based on, and represents a continuation of the development of a system of finite element programs for nonlinear static analysis called PLANS (PLastic and Large Deflection ANalysis of Structures - ref. 4). A review of the salient features of DYCAST can be found in ref. 5.

DYCAST implements an updated Lagrangian formulation (refs. 6,7,8) for geometric nonlinearity and an incremental plasticity theory (ref. 9) to represent material nonlinear behavior.

In this section we outline the governing matrix equations used here in order to identify the key computation bound operations on which the STAR 100 will have its greatest impact.

The governing matrix equation based on the updated Lagrangian formulation with the displacement increment ΔU_{n+1} and acceleration $\ddot{U}_{n+1}$ as unknowns is

$$[K_n^{(0)} + K_n^{(1)}] \; \Delta U_{n+1} = P_{n+1} - F_n - M\ddot{U}_{n+1} \tag{1}$$

This equation assumes that third order terms in the integral work relation have been neglected in going from the nth configuration to the n+1th. These neglected terms are discussed in refs. 6, 8. The matrix term $K_n^{(0)}$ is

$$K_n^{(0)} = \int_{V_n} W_n^t \, D_n^{-1} \, W_n \, d\,V$$

where V_n is the volume in the n^{th} configuration, W_n relates increments in total strain to increments in displacement, and $D_n = E^{-1}$ (I + EC) with E the matrix of elastic material properties and C a matrix obtained from the plastic constitutive relations. The matrix term $K_n^{(1)}$ is

$$K_n^{(1)} = \int_{V_n} \Omega_n^t \, \tau_n \, \Omega \, d\,V$$

where Ω_n relates the nonlinear components of the strain displacement relations to displacement increment and τ_n is a matrix of Cauchy stresses with respect to the nth configuration. The matrix term F_n is

$$F_n = \int_{V_n} W_n^t \, \hat{\tau}_n \, d\,V$$

where $\hat{\tau}_n$ is a vector of Cauchy stresses with respect to the n^{th} configuration. One can use either an explicit or implicit time integrator to obtain solutions to eq. (1). Both types are implemented in DYCAST.

Explicit Integrator

DYCAST currently implements a constant time step central difference and a variable step modified Adams predictor-corrector time integrator. In both procedures, eq. (1) is written as

$$M\ddot{U}_{n+1} = P_{n+1} - F_n - \Delta f_{n+1} \tag{2}$$

where

$$\Delta f_{n+1} = [K_n^{(0)} + K_n^{(1)}]\ \Delta U_{n+1}$$

is the incremental force vector.

This vector is then approximated in terms of previously calculated variables by making use of the discrete time integrator so that if the mass matrix M is unchanging the solution reduces to calculating a right hand side to eq. (2) and then solving for $\ddot{U}_{n+1}$.

For central difference use is made of the recurrence relations

$$\begin{aligned} \Delta U_{n+1} &= 2\Delta U_n - \Delta U_{n-1} + \Delta t^2 \Delta\ddot{U}_n \\ \Delta\dot{U}_n &= \frac{\Delta U_{n+1} - \Delta U_{n-1}}{2\Delta t} \end{aligned} \tag{3}$$

in order to obtain Δf_{n+1}.

The modified Adams procedure is based on substituting a predictor solution for ΔU_{n+1}, into eq. (2)

$$\Delta U^P_{n+1} = \Delta U^{dev}_n + \Delta t\ \dot{U}_n + \frac{\Delta t}{2}(\dot{U}_n - \dot{U}_{n-1}) \tag{4}$$

Equation (4) *is the Taylor series expansion for* U_{n+1} *with the backward* difference used for the acceleration and ΔU^{dev}_n is the difference between the nth predictor and corrector solutions, $U^c_n - U^p_n$. Once eq. (2) is solved for $\ddot{U}_{n+1}$ the corrector solution is generated based on a forward difference for the third term.

$$\begin{aligned} U^c_{n+1} &= U_n + \Delta t\ \dot{U}_n + \frac{\Delta t}{2}(\dot{U}_{n+1} - \dot{U}_n) \\ \dot{U}^c_{n+1} &= \dot{U}_n + \Delta t\ \ddot{U}_n + \frac{\Delta t}{2}(\ddot{U}_{n+1} - \ddot{U}_n) \end{aligned} \tag{5}$$

An error criterion is used to ensure that the difference between the predictor and corrector solutions satisfies some preset error criterion. In practice, the convergence criterion fails on the difference between the predictor and corrector velocities. This is

$$\begin{aligned} \dot{U}^c_{n+1} - \dot{U}^P_{n+1} &= \frac{\Delta t}{2}((\ddot{U}_{n+1} - \ddot{U}_n) - (\ddot{U}_n - \ddot{U}_{n-1})) \\ &= \frac{\Delta t}{2}(\Delta\ddot{U}_{n+1} - \Delta\ddot{U}_n) \end{aligned} \tag{6}$$

and the error criterion becomes

$$\frac{\Delta t}{2}\left|\frac{(\Delta\ddot{U}_{n+1} - \Delta\ddot{U}_{n})}{\dot{U}_{n+1}}\right| < \varepsilon \tag{7}$$

It can be seen from eq. (7) that the error criterion limits the rate of change of acceleration. Whenever the inequality of eq. (7) is violated the time step is halved. Conversely, the time step is doubled whenever the inequality is satisfied within a predetermined lower bound.

Implicit Integrator

A variable time step implicit solution algorithm with inner loop iterations, based on the Newmark-Beta family of integrators, is implemented within DYCAST.

The governing recurrence relations for the Newmark-Beta method are

$$\begin{aligned}\ddot{U}_{n+1} &= \frac{\Delta U_{n+1}}{\beta\Delta t^2} - \frac{\dot{U}_n}{\beta\Delta t} - \left(\frac{1}{2\beta} - 1\right)\ddot{U}_n \\ \Delta\dot{U}_{n+1} &= \Delta t\,\ddot{U}_n + \gamma\Delta t\,\Delta\ddot{U}_{n+1}\end{aligned} \tag{8}$$

Substituting eq. (8) into eq. (1) yields

$$\bar{K}_n \Delta U_{n+1} = P_{n+1} + Q^d_n - F_n \tag{9}$$

where

$$\bar{K}_n = K^{(0)}_n + K^{(1)}_n + \frac{M}{\beta\Delta t^2}$$

$$Q^d_n = M\left(\frac{\dot{U}_n}{\beta\Delta t} + \left(\frac{1}{2\beta} - 1\right)\ddot{U}_n\right)$$

Equation (9) can be solved iteratively within each time step by feeding back the effect of an equilibrium correction term.

In this form eq. (9) becomes

$$\bar{K}_n \Delta U^i_{n+1} = P_{n+1} + Q^d_n - F_n + \sum_{j=o}^{i} R^j_n \tag{10}$$

where

$$R^j_n = P_{n+1} - F^j_{n+1} - M\,\ddot{U}^j_{n+1}$$

When i = 0, eq. (10) reduces to eq. (9) since $R^0_n = 0$.

There are a number of ways to define convergence. Use is made in DYCAST of the following criterion

$$\left\| \frac{\Delta U^{i}_{n+1} - \Delta U^{i-1}_{n}}{U^{i}_{n+1}} \right\| < \varepsilon \tag{11}$$

In addition to eq. (11), an admissibility test is used in order to control the time step. Based on our experience with the modified Adams method, the admissibility test is developed by writing a predictor form of eq. (8) that leads directly to a criterion similar to eq. (7)

$$\gamma\Delta t \left| \frac{\Delta\ddot{U}_{n+1} - \Delta\ddot{U}_{n}}{\dot{U}_{n+1}} \right| < \varepsilon \tag{12}$$

ALARM IMPLEMENTATION

As an initial test of STAR vector processing, a program for the eigenvalue/eigenvector evaluation for large matrices was converted to STAR. This program called ALARM (Automatic LArge Reduction of Matrices to tridiagonal form) is based upon a method which reduces a large matrix to an "equivalent" tridiagonal one of much smaller size. It is based on the work of Ojalvo and Newman (ref. 10) and as such is similar to the eigensolver, FEER, which has been implemented in recent versions of NASTRAN. For full details of the method the reader is referred to ref. 3 for ALARM and ref. 11 for FEER. In the following we outline the computational flow from ref. 3 as a means of identifying the areas of vector processing. The algorithm almost exclusively involves operations on large matrices and as such is particularly suitable for vector processing. It is based on generating a "tall" rectangular transformation matrix, V, the columns of which are orthonormal vectors. This transformation matrix reduces the equation

$$K\,\phi_i = \omega_i^2\, M\,\phi_i \tag{13}$$

where K and M are the stiffness and mass matrices, respectively, to symmetric tridiagonal form of much smaller size. The eigenvalues/eigenvectors of this equation are then obtained by Sturm sequencing and bisection of the intervals containing ω_i^2.

The sequence of operations excluding the starting procedure and error checking is as follows (ref. 3):

1. Factor the stiffness matrix into upper and lower triangular form $K = LL^t$

2. Solve for successive columns of the reduction vector, V, as follows:

$$v^*_{i+1} = L^{-1} M L^{-t} v_i$$

This is accomplished by first solving for

$$L^t w_i = v_i$$

then performing the matrix multiplication

$$u_i = Mw_i$$

and then solving

$$L v^*_{i+1} = u_i$$

All of these operations involve vector processing on vectors of the order of the semibandwidth and bandwidth of K and M. The procedure is completed by

3. $\bar{v}_{i+1} = v^*_{i+1} - \alpha_i v_i - \beta_{i-1} v_{i-1}$

where

$$\alpha_i = v_i v^*_{i+1}$$

$$\beta_{i-1} = v_{i-1} v^*_{i+1}$$

The desired column of V is then

$$v_{i+1} = \frac{\bar{v}_{i+1}}{\beta_i}$$

where α_i, β_i are the diagonal and off-diagonal terms, respectively, of the final reduced tridiagonal matrix.

The method outlined above was vectorized using the STAR 100 subset of vector FORTRAN syntax. Results are shown in Table I for CPU time for a number of benchmark problems using an IBM 370/168, CDC STAR 100 with all scalar operations and STAR 100 using vector processing. The times shown for benchmark problems exclude the factoring of the stiffness matrix, and assume a diagonal mass matrix. The table reveals that as the matrix size increases, the relative running time on the STAR 100 in the scalar mode increases relative to the IBM 370/168. However, as the problem size increases, the payoff for vector calculations grows. The final result, that of a 10000 degree of freedom matrix with a semibandwidth of 100 and a running time of 9.91 CPU seconds, indicates almost a 160-fold decrease in time over the IBM 370/168.

This table clearly indicates two points: the efficiency of vector processing for large global arrays and the relative inefficiency of the current version of the STAR 100 for primarily scalar arithmetic.

Figure 1 shows a section of a longitudinally compressed skin stringer element typical of the NASA space shuttle wing cover construction. A linear bifurcation buckling analysis of this structure was performed using one of the PLANS system programs. This program uses a higher order plate element to model the structure and the ALARM program to obtain the eigenvalues and eigenvectors. The resulting model led to matrices with 3158 degrees of freedom and a semi-bandwidth of 301. Running time on an IBM 370/168 in double precision depended on the amount of available core. Table II shows data from ref. 12 and indicates the normalized running time versus increased core size. The increased core size was due to increasing the work area available to the matrix packages. In no case, however, were we able to use more than 150000 words. This corresponds to the available incore capacity of STAR equal to approximately 450000 words for code and data and demonstrates a source of efficiency of STAR due to large incore storage coupled with the virtual core facility. This same problem was run with the STAR version of ALARM. To do this, the necessary matrices were assembled and passed to STAR via tape files.

The resulting computer times are summarized in Table III. Evaluation of two eigenvalues and eigenvectors to the desired accuracy took 12.5 CPU seconds on the STAR 100 computer, compared to 269.6 CPU seconds on the IBM 370/168 in double precision using 921 KBYTES of core. This represents a ratio of 21.5 to 1 which is consistent with the data presented in Table I. The computer time to calculate one column of the transformation matrix, step 2 outlined above, was 0.9 CPU second for the STAR 100 and 19 CPU seconds for the IBM 370/168. Also shown in Table III is the time to factor the stiffness matrix. To perform this operation, use was made of a vectorized equation solver written by Dr. J. Lambiotte, Jr. of NASA Langley Research Center. This procedure factors the matrix as $\bar{L}D\bar{L}^t$, where $\bar{L}$ is a lower triangular matrix and D is diagonal. In order to obtain the Cholesky factorization required by ALARM, we performed the additional operation of taking the square root of the diagonal matrix and premultiplied $\bar{L}$ by the result.

The predicted buckling load obtained from this analysis agreed quite favorably with the average of three tests (see fig. 1). Plots of the mode shape predicted by this model are also shown in fig. 1. For a clearer representation, only the portions of the structure that buckled are shown, and the deformed sheet and stringer flanges were plotted as if detached.

DYCAST COMPUTER IMPLEMENTATION

Given that the appropriate theories from structural mechanics have been implemented, the features that distinguish a program for nonlinear analysis from a linear program are: 1) the solution algorithm is of the repetitive type so that calculations performed once for a linear analysis must be repeated for a nonlinear analysis, and 2) field quantities such as displacements, stresses, and strains must be stored for use in succeeding calculations. These considerations force the designer of a nonlinear code to exercise extreme care in coding key "number crunching" sections since any inefficiencies, while perhaps not being crucial for a linear analysis, are multiplied "n" times in a nonlinear analysis. It is in these areas that the STAR vector processing can potentially make a contribution.

In the following, the flow of the computational bound section of DYCAST is outlined in order to identify areas of potential vector processing. The program is separated into two functional units as shown in fig. 2. A small main program initially determines core allocations and then passes control to a subroutine that reads and processes all input and defines necessary data bases. Since computations are minimal in this routine the only impact of the STAR 100 here is the use of virtual core capability over utilization of temporary scratch files.

After the input phase, control is returned to the calling program which in turn calls a subprogram that controls the main computational loop. This loop, shown in block diagram form in fig. 3, has all the ingredients usually found in a linear finite element program, namely, (1) Matrix assembler, (2) Equation solver, (3) Time integrator, (4) Finite element matrix formation, and (5) Stress and strain calculations.

Items 1 through 3 are global functions that are easily recast into vector form. That is, the equation solver involves dot products on vectors whose length is of the order of the semibandwidth of the matrix, a process that can exploit the STAR vector processing. Implementing the integrator, although not taking a large percentage of the computation time, involves sums of vectors that can easily utilize STAR vector processing. However, items 4 and 5 are carried out on the element level and involve primarily operations with small matrices.

The method devised to implement element level calculations can be an important consideration. This is because it has been our experience that more than half the computer time per time step is taken by the element level calculations (for moderate sized problems) when using an implicit integrator. The relative time can be as much as 3/4 for an explicit integrator. Figure 4 shows the computational flow for a typical element level sequence of calculations for the simplest case of a three node constant strain triangle. For this element, the calculations involve primarily scalar arithmetic along with vector products on vectors of length 3 and 9. These lengths are too small for efficient vector processing because of instruction start-up time for vector operations. Vectors of length 100 begin to substantially exploit the STAR pipeline processing. It is also worth noting that since a finite element model consists of a multitude of elements, a computer system optimized for this type of operation might more naturally be based on parallel rather than pipe-line processing. That is, calculations can be carried out on many elements simultaneously. Reference 13, for example, conceptually describes the implementation of a finite element program on the Illiac IV parallel processing computer.

Practical experience was gained in implementing the items discussed above into a finite element program for nonlinear analysis by the Computer Sciences Corporation under Contract with NASA Langley Research Center. This program developed for plastic analysis alone, is part of PLANS (Plastic and Large deflection ANalysis of Structures - ref. 4). The program chosen treats the plastic analysis problem using the "initial strain" or pseudo load method. Consequently, the problem reduces to the solution of a sequence of linear analyses with a changing pseudo load vector that accounts for plasticity. The stiffness matrix is unchanged in each step so that it can be factored once with subsequent solutions requiring only a forward and backward substitution of triangular matrices. Within this framework the major effort in each step is to solve the set of finite element equations, calculate stresses and strains (impose plastic constitutive equations), and reformulate the vector of pseudo loads. Thus, it has all the steps that the explicit formulation of DYCAST has, eq. (2), with the exception that DYCAST must form a contribution to the pseudo load vector, Δf_{n+1}, that is the incremental internal force vector. Consequently, the effectiveness of the STAR computer on this program should carry over to DYCAST.

Results provided to us by R. Fudurich and D. Dunlop of Computer Sciences Corporation are shown in Table IV and summarize the results obtained to date. These results indicate a substantial speedup in the solution algorithm, up to 13:1 for initial solutions, and up to 190:1 for subsequent solutions that require only forward and back substitution. However, in the area of stress and strain recovery, where operations are presently primarily scalar, the STAR computer was appreciably slower than the CDC CYBER 175. As mentioned previously these element level calculations are performed on small matrices for which we anticipate that the vector capability would not lead to significant savings. Based on the results shown in Table IV it becomes apparent that it is essential to restructure the program in this area in order to exploit the STAR 100 pipeline capability.

Work is currently underway to convert the DYCAST program to the STAR 100 computer. To date, subroutines for matrix assembly and equation solution that use the STAR virtual core and vector processing capability have been implemented into DYCAST. The effect of vector processing was demonstrated on a finite element model containing 332 elements (179 triangular membrane, 33 axial force stringer, 120 beam), 413 degrees of freedom, and 136 semibandwidth. The computer times for one time step for this problem using both the implicit and explicit integrators on an IBM 370/168 and CDC STAR 100 are shown in Table V. As anticipated, any savings due to vectorizing the solution algorithm is offset due to the slower computation time on STAR for scalar operation. Vector processing, although on small vectors, is currently being introduced into the element level routines.

CONCLUSIONS AND RECOMMENDATIONS

This paper discusses our successes and pitfalls in converting a finite element program for nonlinear finite element analysis to the CDC STAR 100 computer. Our experience may be summarized by the following: On global functions, the vector processing, coupled with the virtual memory capability, led to decisively improved computation times over the available scalar computers. However, on element level computations that were not vectorized, and indeed do not lend themselves easily to long vector processing, the STAR 100 was decisively slower than comparable scalar computers. We do expect some slight improvement when these routines are converted to vector code. Consequently, as the number of successive linearization increments increased in a nonlinear analysis, the gains made on processing global functions were offset by the element level calculations. On this basis it can be concluded that in order for pipeline computers to impact the economic feasibility of large nonlinear analyses it is absolutely essential that algorithms be devised to improve the computational efficiency of element level computations. Recommendations to accomplish this are discussed below.

In problems involving plasticity alone it is usual that a contained region of plastic flow exists. For this situation the pseudo load method can be used and element level stress recovery can be simply limited to this contained region. This can be done using the pseudo load method since these effective plastic loads are nonzero only in the contained region and element stiffness matrices are not reformed in each increment. This notion can be expanded even further (ref. 14) by employing a substructuring technique to eliminate the unknowns in the assumed elastic region. Since these calculations are exclusively on global arrays, the substructuring operations can effectively make use of vector processing. The remaining incremental calculations can then be carried out using the reduced set of equations.

Because of the effectiveness of vector processing on global arrays, any method that operates on these primarily large matrices during an incremental nonlinear solution will be computationally efficient on the STAR 100. Methods were previously developed (ref. 9) that eliminate the displacements in the governing matrix equation so that the remaining incremental equations are global arrays of stress or strain increments. For example, the total strain increment can be written as shown in ref. 9 as follows:

$$\Delta e^T = A\ \Delta P + J\ \Delta\varepsilon \tag{14}$$

where $\Delta\varepsilon$ is the incremental plastic strain, ΔP is an incremental load factor, and A and J are matrices that depend on the number of strain recovery points in the finite element model. Once eq. (14) is formulated, it is solved incrementally as part of the nonlinear analysis thereby replacing the usual element level calculations. Consequently, it may be worthwhile to re-examine this method for implementation on the STAR 100.

The previous comments are also pertinent to the formulation of linear dynamic analysis because the coefficient matrices do not change when either the explicit or implicit methods are employed. Consequently, the method reduces to calculating the right hand side vector and then solving for either accelerations or displacements. The right hand side vector for both methods can be recast in terms of global matrices by assembling and storing the total stiffness and mass matrices and then performing the matrix multiplications indicated in eqs. (2) and (9). Stress and strain recovery may still be a limiting factor, but these need not be computed in every time step.

The methods described above cannot be used in DYCAST because the stiffness matrix changes in each step due to the problem nonlinearities. One is therefore constrained to perform these element level calculations in every time step. Consequently these calculations must be reformulated so that they can effectively use vector processing. It may be possible to do this by partitioning the structure so that calculations are carried out on assemblages of elements, i.e., super elements. This technique will be pursued in future developments of DYCAST.

REFERENCES

1. Noor, A.K.; and Fulton, R.E.: Impact of the CDC STAR 100 Computer on Finite Element Systems, J. Structural Div. ASCE, vol. 101, 1975, pp. 731-750.

2. Study of the Modifications Needed for Efficient Operation of NASTRAN on the Control Data Corporation STAR 100 Computer. Aerospace Division of Control Data Corp., NASA CR-132644.

3. Ojalvo, I.U.: ALARM -- A Highly Efficient Eigenvalue Extraction Routine for Very Large Matrixes. The Shock and Vibration Digest, vol. 7, no. 12, Dec. 1975.

4. Pifko, A.; Levine, H.S.; and Armen, H. Jr.: PLANS - A Finite Element Program for Nonlinear Analysis of Structures, vol. I - Theoretical Manual. NASA CR-2568, August 1974.

5. Winter, R.; Pifko, A.; and Armen, H. Jr.: Crash Simulation of Skin-Frame Structures Using a Finite Element Code. Presented at Soc. of Automotive Engineers, Business Aircraft Meeting, Wichita, Kansas, Mar. 29 - Apr. 1, 1977, Paper no. 770484.

6. Armen, H.; Levine, H.; Pifko, A.; and Levy, A.: Nonlinear Analysis of Structures. NASA CR-2351, March 1974.

7. Hofmeister, L.; Greenbaum, G.; and Evensen, D.: Large Strain Elasto-Plastic Finite Element Analysis. AIAA J., vol. 9, no. 7, 1971, pp. 1248.

8. Bathe, K.; Ramm, E.; and Wilson, E.L.: Finite Element Formulations for Large Deformation Dynamic Analysis. Inter. J. for Numerical Methods in Engineering, vol. 9, 1975, pp. 353-386.

9. Armen, H. Jr.; Pifko, A.; and Levine, H.: Finite Element Analysis of Structures in the Plastic Range. NASA CR-1649, February 1971.

10. Ojalvo, I.U.; and Newman, M.: Vibration Modes of Large Structures by Automatic Matrix Reduction Method, AIAA J., vol. 8, no. 7, July 1970, pp. 1234-1239.

11. Newman, M.; and Pipano, M.: Fast Model Extraction in NASTRAN via the FEER Computer Program. NASA TMX-2893, Sept. 1973, pp. 485-506.

12. Crouzet-Pascal, J.: PLANS - Current and Potential Capabilities for Finite Element Analysis of Intersecting Thin Shell Structures. Grumman Research Dept. Memorandum RM-635, June 1977.

13. Field, E.I.; Johnson, S.E.; and Stralberg,: Software Development Utilizing Parallel Processing. Structural Mechanics Computer Programs, Surveys, Assessments, and Availability, Univ. Press of Virginia, Charlottesville, VA, 1974.

14. Armen, H., Jr.; and Levy, A.: Substructuring, Restart, and Variable Constraints in a Three-Dimensional Finite Element Program for Fracture Analysis. Grumman Aerospace Corp. Report RE-553, April 1978.

TABLE I. - COMPUTER TIME (CPU SECONDS EXCLUSIVE OF FACTORING STIFFNESS MATRIX) FOR EIGENVALUE/EIGENVECTOR EXTRACTION FOR VARIOUS SIZE MATRICES

Matrix Size	Maximum Semi-bandwidth	Solution Time IBM 370/168 (single precision)	Solution Time STAR 100 Scalar	Solution Time STAR 100 Vector	$\frac{\text{STAR Scalar}}{\text{STAR Vector}}$	$\frac{\text{IBM 370}}{\text{STAR Vector}}$	$\frac{\text{IBM 370}}{\text{STAR Scalar}}$
10	1	3.44	1.30	0.61	2.13	5.64	2.65
100	10	4.47	5.64	1.05	6.42	4.26	0.79
1000	100	26.91	76.21	1.76	45.06	15.29	0.353
4000	100	175.08		4.86		36.02	
4000	400	225.52		7.18		31.4	
10000	100	1532.3+		9.91		154.6	

+Time to form tridiagonal matrix

TABLE II. - PERCENT DECREASE IN CPU TIME ON IBM 370/168 VERSUS AVAILABLE CORE SIZE FOR BUCKLING ANALYSIS OF SKIN-STRINGER PROBLEM

Problem Size = 3158 degrees of freedom

Semibandwidth = 301

Core Utilized KBYTES	Normalized CPU Time
964	1.00
1356	0.667
1744	0.564

TABLE III. - COMPARISON OF CPU TIMES IN SECONDS FOR BUCKLING ANALYSIS OF SKIN-STRINGER PROBLEM

	Step 1 Factor K (CPU Time)	Step 2 Eigenvalue Eval. (CPU Time)	Total (CPU Time)
IBM 370/168	342.	269.6	611.6
CDC STAR 100	57.7	12.5	70.2
Ratio 370/STAR	5.9	21.5	8.7

TABLE IV. - SUMMARY OF CDC STAR 100 VERSUS CDC CYBER 175 COMPUTING TIMES FOR PLANS PROGRAM

Problem	No. of Elements	DOF	Semi-Bandwidth	Initial Solution[+]	Initial Total[*]	Total Subsequent Solutions	Total S Recove Pseudo
1	25	138	42	0.2126/ 1.514	3.885/ 3.864	5.229/ 470.265	530.0 109.7
2	66	300	60	0.8142/ 5.390	8.8926/ 8.305	1.4579/ 277.679	191.9 34.71
3	560	1492	93	6.3/ 82.96	45.26/ 100.7	0.312/ N.A.	35.04 N.A.
4	710	4366	170	58.1/ 546.9	---	---	---

[+]Refers to initial factoring and forward and backward solution. All times given
STAR time/CYBER-175 time.

[*]All initial preprocessing, element formation, critical load stresses and strains
algorithm vectorized.

Problem 4 Compared versus IBM 370/168; Total times depend on number of increme
range.

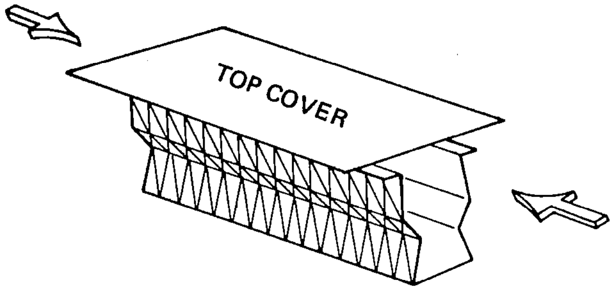
TABLE V. - COMPARISON OF CDC STAR 100 VERSUS IBM 370/168 COMPUTING TIMES FOR ONE TIME STEP USING DYCAST 413 DOF, 136 SEMIBANDWIDTH, 332 ELEMENTS

	Explicit	Implicit (one iteration)
Matrix Assembly	0.0/0.0	0.32/4.03
Solution	0.03/0.65	2.05/6.77
Time Integrator	0.02/0.02	0.04/0.10
Stress/Strain Recovery and Element Formation	8.86/4.08	16.90/6.87
Total	8.91/4.75	19.31/17.77

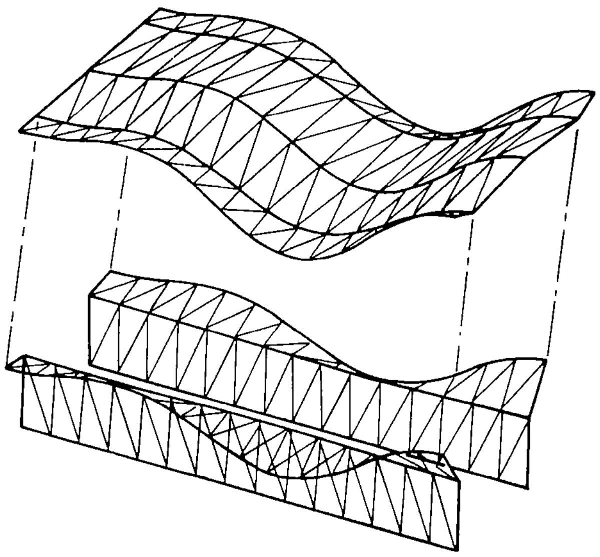
All times given in CPU seconds and STAR time/IBM 370/168.

BUCKLING LOAD, N (lb)
BENST 81500 (18320)
TESTS(3) 88850 (19970) AVERAGE

SKIN THICKNESS = 0.18 cm (0.070 in.)
LENGTH = 21.5 cm (8.5 in.)



a) FINITE ELEMENT MODEL OF STRINGER



b) COMPUTER PHOTO OF BUCKLING MODE SHAPE IN SKIN AND ATTACHED STRINGER FLANGES (ONLY PORTIONS OF STRINGER THAT BUCKLED)

Figure 1 Analysis of Longitudinally Compressed Skin-Stringer Element Typical of Shuttle Wing Cover Construction: Comparison of Computed Buckling Load with Test Results

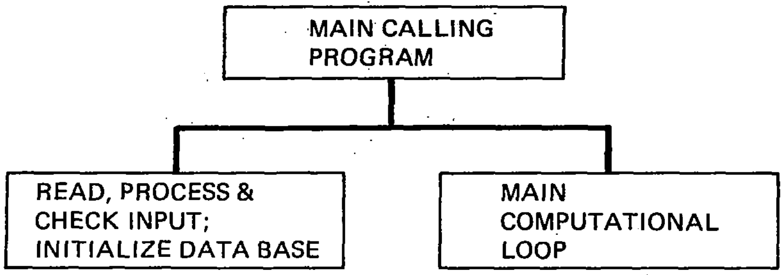


Figure 2 Global Structure of DYCAST

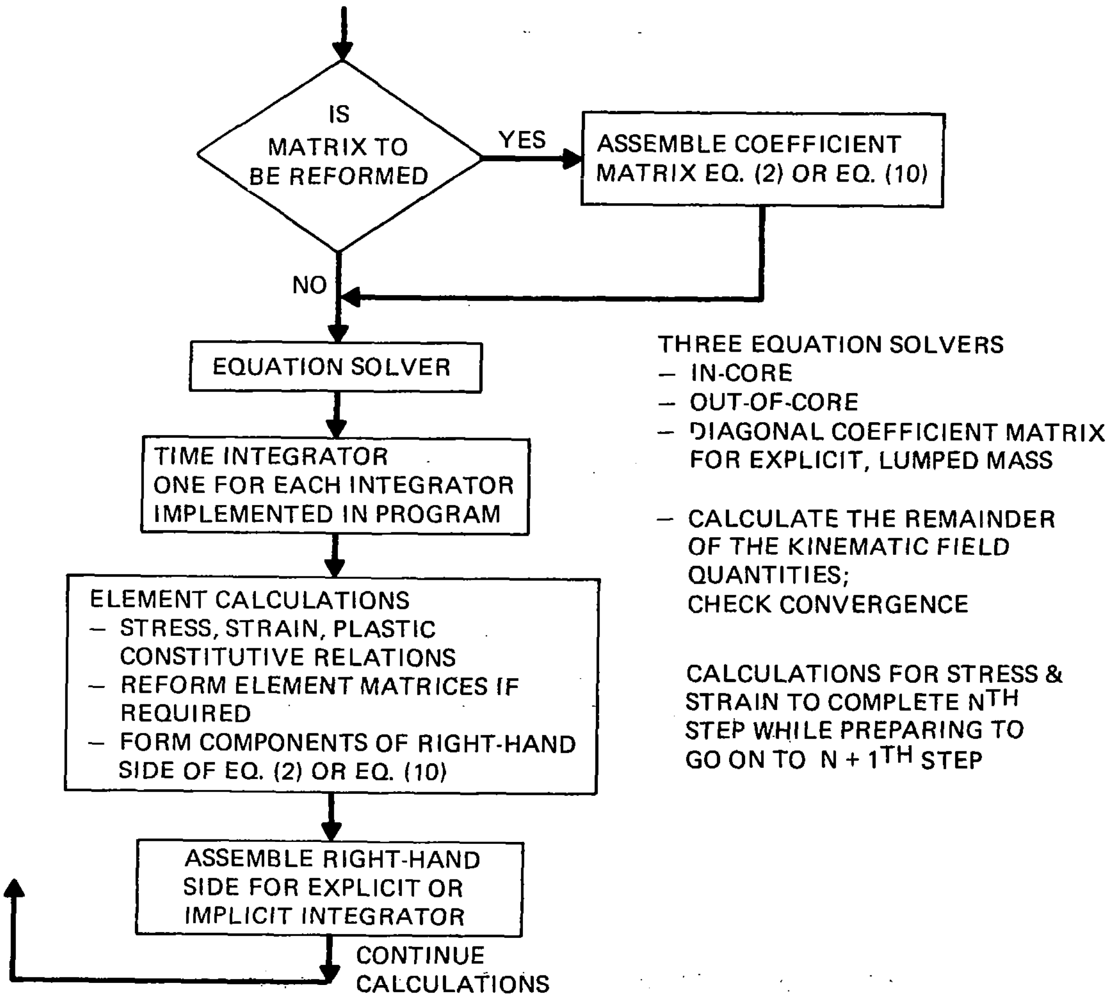

IS MATRIX TO BE REFORMED
YES
ASSEMBLE COEFFICIENT MATRIX EQ. (2) OR EQ. (10)
NO
EQUATION SOLVER
THREE EQUATION SOLVERS
– IN-CORE
– OUT-OF-CORE
– DIAGONAL COEFFICIENT MATRIX FOR EXPLICIT, LUMPED MASS
TIME INTEGRATOR ONE FOR EACH INTEGRATOR IMPLEMENTED IN PROGRAM
– CALCULATE THE REMAINDER OF THE KINEMATIC FIELD QUANTITIES; CHECK CONVERGENCE
ELEMENT CALCULATIONS
– STRESS, STRAIN, PLASTIC CONSTITUTIVE RELATIONS
– REFORM ELEMENT MATRICES IF REQUIRED
– FORM COMPONENTS OF RIGHT-HAND SIDE OF EQ. (2) OR EQ. (10)
CALCULATIONS FOR STRESS & STRAIN TO COMPLETE NTH STEP WHILE PREPARING TO GO ON TO N + 1TH STEP
ASSEMBLE RIGHT-HAND SIDE FOR EXPLICIT OR IMPLICIT INTEGRATOR
CONTINUE CALCULATIONS

Figure 3 Computional Flow of Major Loop of DYCAST

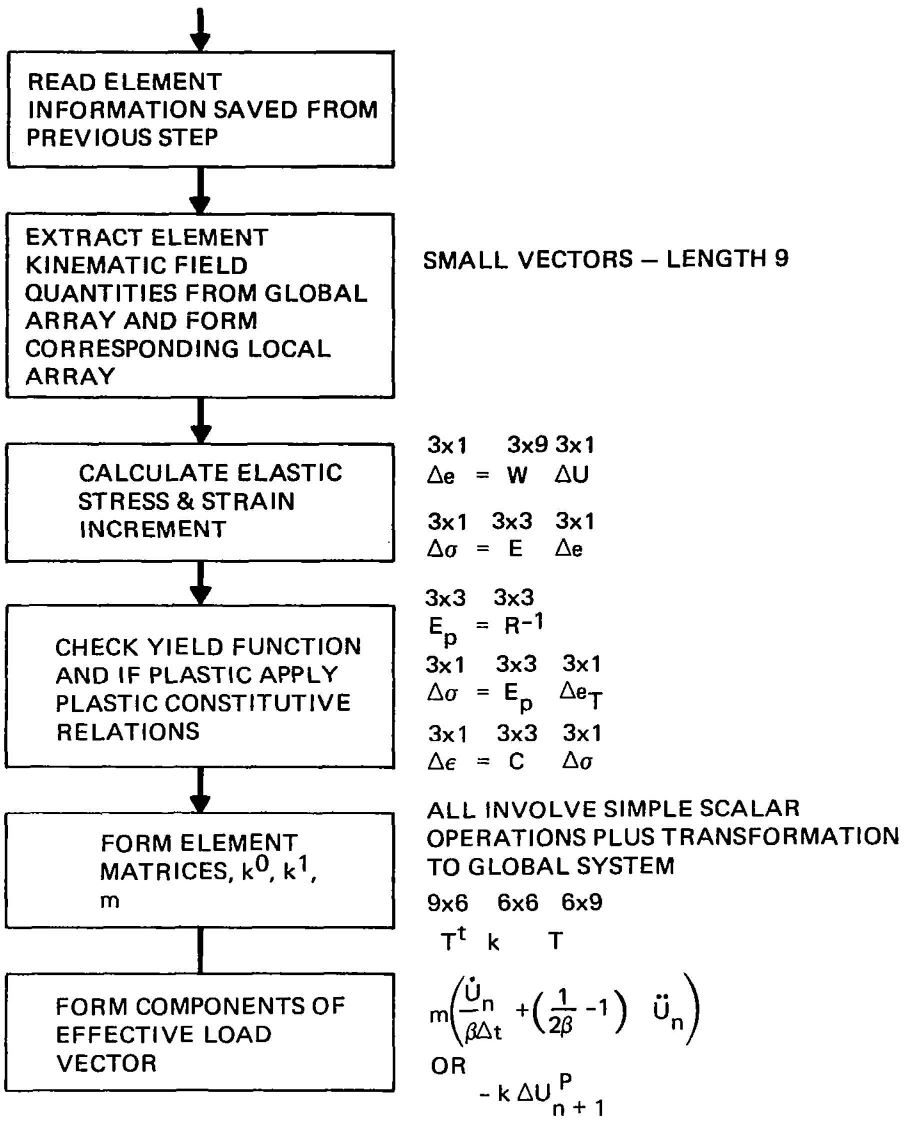

READ ELEMENT INFORMATION SAVED FROM PREVIOUS STEP
EXTRACT ELEMENT KINEMATIC FIELD QUANTITIES FROM GLOBAL ARRAY AND FORM CORRESPONDING LOCAL ARRAY
SMALL VECTORS – LENGTH 9
CALCULATE ELASTIC STRESS & STRAIN INCREMENT
$\underset{3x1}{\Delta e} = \underset{3x9}{W}\ \underset{3x1}{\Delta U}$
$\underset{3x1}{\Delta\sigma} = \underset{3x3}{E}\ \underset{3x1}{\Delta e}$
CHECK YIELD FUNCTION AND IF PLASTIC APPLY PLASTIC CONSTITUTIVE RELATIONS
$\underset{3x3}{E_p} = \underset{3x3}{R^{-1}}$
$\underset{3x1}{\Delta\sigma} = \underset{3x3}{E_p}\ \underset{3x1}{\Delta e_T}$
$\underset{3x1}{\Delta\epsilon} = \underset{3x3}{C}\ \underset{3x1}{\Delta\sigma}$
FORM ELEMENT MATRICES, k^0, k^1, m
ALL INVOLVE SIMPLE SCALAR OPERATIONS PLUS TRANSFORMATION TO GLOBAL SYSTEM
$\underset{9x6}{T^t}\ \underset{6x6}{k}\ \underset{6x9}{T}$
FORM COMPONENTS OF EFFECTIVE LOAD VECTOR
$m\left(\frac{\dot{U}_n}{\beta\Delta t} + \left(\frac{1}{2\beta} - 1\right)\ddot{U}_n\right)$
OR
$- k\,\Delta U^P_{n+1}$

Figure 4 Computational Flow of Element Level Subroutine

ON A NEW ALGORITHM FOR TIME STEP INTEGRATION OF NONLINEAR SYSTEMS

Edoardo Anderheggen and Gianni Bazzi
Swiss Federal Institute of Technology Zürich

SUMMARY

A new implicit algorithm for time step integration of finite element structural dynamic equations is presented. Convergence, stability and numerical damping properties are discussed. Due to the way nonlinear structural behavior is taken into account the algorithm is expected to compare favourably with existing ones. Some simple numerical results are presented. A related explicit algorithm is also derived and shortly discussed.

INTRODUCTION

Lately much attention is being paid to finite element structural dynamic problems where nonlinear behavior is taken into account. It is typical for this kind of problems that the internal nodal forces developed by the structure and resisting external loads and inertia forces are not linear functions of nodal displacements but have to be evaluated from the actual stress state of the deformed structure by virtual work integration or similar procedures. The displacement time history is then obtained integrating step-by-step the nonlinear dynamic equations, a very cumbersome procedure in most practical cases.

Time step algorithms can be classified as implicit, requiring the solution of a system of coupled and generally nonlinear equations at each time step or as explicit where the unknown problem parameters at the end of each step are obtained directly. A clear and concise discussion of requirements and applicability for both kinds of algorithms can be found in reference [1].

In the present paper an implicit algorithm is first derived and discussed using simple numerical tests to show some of its properties. A corresponding explicit algorithm (which has not been implemented yet) is then also derived. It should be clear, however, that the present paper has the limited scope of reporting some early results of a research project presently in progress. More extensive numerical tests and comparisons with different algorithms are needed for general conclusions.

Within a time step Δt, i.e. between t_0 and $t_1 = t_0 + \Delta t$ (with $\tau = t-t_0$), the following system of differential equations, a dot indicating derivation with respect to time, has to be solved:

$$\{R(\tau)\} = [M]\{\ddot{W}(\tau)\} + [C]\{\dot{W}(\tau)\} + \{F(\tau)\} - \{P(\tau)\} = 0$$

where: $\{R(\tau)\}$ = residual vector; $\{W(\tau)\}$ = vector of nodal displacement parameters; $[M]$ = mass matrix; $[C]$ = viscous damping matrix; $\{F(\tau)\}$ = vector of internal nodal forces; $\{P(\tau)\}$ = vector of prescribed external loads. The internal structural nodal forces $\{F(\tau)\}$ are evaluated as follows (see Fig. 1):

$$\{F(\tau)\} = \{F_0\} + [\tilde{K}](\{W(\tau)\} - \{W_0\}) + \{\overline{F}(\tau)\}$$

where: $\{F_0\}$ = internal forces at $t = t_0$ (subscripts 0 and 1 always refer to the beginning and the end of the time step); $[\tilde{K}]$ = an approximation of the actual secant stiffness matrix between t_0 and t_1 (generally the tangent stiffness matrix evaluated at $t = t_0$ or at the beginning of some earlier time step will be used for $[\tilde{K}]$); $\{\overline{F}(\tau)\}$ = vector of corrective internal forces due to the nonlinear behavior of the structure within the time step. The distribution of these corrective forces $\{\overline{F}(\tau)\}$ is assumed to be given by:

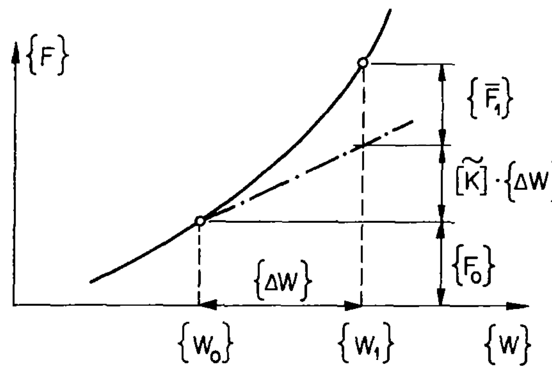


Fig. 1 : Internal Forces

$$\{\overline{F}(\tau)\} = [b_{ii}(\tau)]\{\overline{F}_1\}$$

where the $b_{ii}(\tau)$, coefficients of the diagonal interpolation matrix $[b_{ii}(\tau)]$, are time functions (with $b_{ii}(0) = 0$ and $b_{ii}(\Delta t) = 1$) to be chosen according to the expected distribution of each single corrective force $\overline{F}_i(\tau)$ as explained later. The internal corrective forces $\{\overline{F}_1\}$ at the end of the time step will have to be evaluated by stress-virtual-strain integration or by similar procedures (taking into account strain-stress history, large displacements, etc.) for all elements where nonlinear behavior is expected.

For the displacement parameters the following assumptions are used:

$$\{W(\tau)\} = N_1(\tau)\cdot\{W_0\} + N_2(\tau)\cdot\{\dot{W}_0\} + N_3(\tau)\cdot\{W_1\}$$

the N_i's being shape functions in the time dimension. In order to insure the necessary continuity of $\{W\}$ and $\{\dot{W}\}$ (but not of $\{\ddot{W}\}$) between time steps the following conditions have to be satisfied:

$$N_1(0) = 1 \qquad \dot{N}_1(0) = N_1(\Delta t) = 0$$
$$\dot{N}_2(0) = 1 \qquad N_2(0) = N_2(\Delta t) = 0$$
$$N_3(\Delta t) = 1 \qquad N_3(0) = \dot{N}_3(0) = 0$$

We also impose the following conditions:

$$\dot{N}_1(\Delta t) = -\frac{1+\rho}{\Delta t} \qquad \dot{N}_2(\Delta t) = -\rho \qquad \dot{N}_3(\Delta t) = \frac{1+\rho}{\Delta t}$$

$$\int_0^{\Delta t} N_1 \cdot d\tau = \frac{\rho}{1+\rho} \cdot \Delta t \qquad \int_0^{\Delta t} N_2 \cdot d\tau = 0 \qquad \int_0^{\Delta t} N_3 \cdot d\tau = \frac{1}{1+\rho} \cdot \Delta t$$

the parameter ρ $(0 < \rho \leq 1)$ being the spectral radius in the limit $\Delta t \to \infty$ of the operator matrix for the linear case as explained later. Polynomials of 4th degree in s (with $s = \tau/\Delta t$) can be chosen for the shape functions N_i:

$$N_1 = 1 - N_3$$
$$N_2 = (-5(1+\rho)s^4 + 4(3+2\rho)s^3 - 3(3+\rho)s^2 + 2s) \cdot \frac{\Delta t}{2}$$
$$N_3 = (5(\rho^2-4\rho+7)s^4 - 8(\rho^2-5\rho+9)s^3 + 3(\rho^2-6\rho+13)s^2) \cdot \frac{1}{2+2\rho}$$

For the important special case where $\rho = 1$ (no numerical damping) the shape of the N_i functions is shown in Fig. 2.

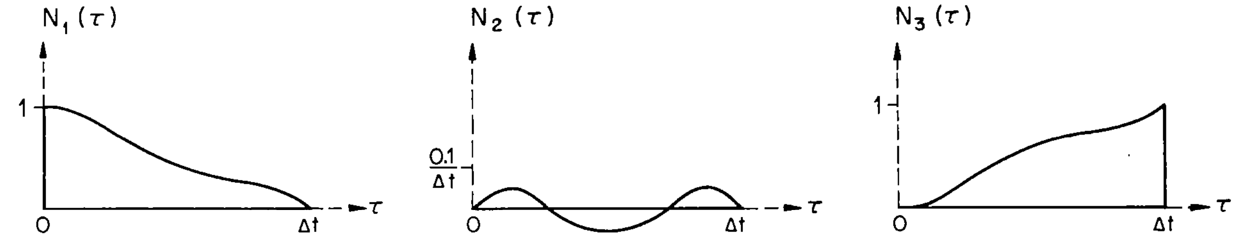


Fig. 2 : Shape Functions $N_1(\tau)$ $N_2(\tau)$ $N_3(\tau)$

In order to evaluate $\{\dot{W}_1\} = \{\dot{W}(t_1)\}$ the weighted integral of the residual vector $\{R(\tau)\}$ is set to zero at each time step:

$$\int_0^{\Delta t} G \cdot \{R(\tau)\} \cdot d\tau = 0 \qquad \text{with: } G = \frac{\Delta t}{1+\rho}$$

where the choice of a constant weighting function G insures that the integral of all external and internal forces vanishes at each time step. The following system of equations for $\{\Delta W\} = \{W_1\} - \{W_0\}$ is found:

$$\left([M] + \frac{\Delta t}{1+\rho}[C] + \frac{\Delta t^2}{(1+\rho)^2}[\tilde{K}]\right)\{\Delta W\} = \Delta t[M]\{\dot{W}_0\} - \frac{\Delta t^2}{1+\rho}(\{F_0\} + [B_{ii}]\{\overline{F}_1\} - \{P^*\})$$

where:
$$[B_{ii}] = \frac{1}{\Delta t}\int_0^{\Delta t}[b_{ii}(\tau)] \cdot d\tau \qquad \{P^*\} = \frac{1}{\Delta t}\int_0^{\Delta t}\{P(\tau)\} \cdot d\tau$$

As the matrix multiplying the unknown vector $\{\Delta W\}$ has the same structure as the stiffness matrix $[\tilde{K}]$, the algorithm is called implicit. The corrective forces $\{\overline{F}_1\}$ can only be evaluated when $\{\Delta W\}$ is known; the system of equations is therefore nonlinear. Unless the influence of $\{\overline{F}_1\}$ is neglected, equilibrium iterations within each time step are necessary, requiring reiterate evaluations of $\{\overline{F}_1\}$. A norm for the changes of $\{\overline{F}_1\}$ (rather than $\{\Delta W\}$) was used in all numerical examples as a convergence criterion.

The displacement velocity vector $\{\dot{W}_1\}$ is found from the assumed shape functions:

$$\{\dot{W}_1\} = \frac{1+\rho}{\Delta t}\{\Delta W\} - \rho\{\dot{W}_o\}$$

The convergence and stability properties of the algorithm for proportionally damped, linear-elastic free vibrations can be discussed applying a modal transformation (see reference [2]). For a mode Y(t) associated with a frequency ω and with a damping coefficient ξ the following operator relations are obtained ($\Omega = \omega \cdot \Delta t$):

$$\begin{Bmatrix} Y_1 \\ \Delta t \cdot \dot{Y}_1 \end{Bmatrix} = \frac{1}{1+\dfrac{2\xi\Omega}{1+\rho}+\dfrac{\Omega^2}{(1+\rho)^2}} \cdot \begin{bmatrix} 1+\dfrac{2\xi\Omega}{1+\rho}-\dfrac{\rho\Omega^2}{(1+\rho)^2} & 1 \\ -\Omega^2 & 1-\dfrac{2\rho\xi\Omega}{1+\rho}-\dfrac{\rho\Omega^2}{(1+\rho)^2} \end{bmatrix} \begin{Bmatrix} Y_o \\ \Delta t \cdot \dot{Y}_o \end{Bmatrix}$$

The operator matrix shows the following properties:

1. For small Δt's i.e. in the limit $\Omega \to 0$ convergence to the true solution is insured.

2. The factor ρ represents the spectral radius of the two-roots operator matrix in the limit $\Omega \to \infty$. The algorithm is therefore unconditionally stable for $\rho \leq 1$.

3. With $\rho = 1$ the operator matrix becomes identical to Newmark's with $\gamma = 1/2$ and $\beta = 1/4$ ("trapezoidal rule", see references [3] and [2]). No numerical damping, second order accuracy, minimal period elongation and no amplitude decay are obtained.

4. For $\rho < 1$ numerical damping used to "filter out" disturbing high frequencies is introduced. In fact, the choice of ρ represent a convenient and natural way of controlling numerical damping. This, however, decreases linearly with Ω (as in Newmark's method with $\gamma > 1/2$) and not quadratically as might be desirable (see references [2], [4], [5]). For $\rho < 1$ second order accuracy is therefore lost, which limits the choice of ρ to values close to unity.

5. The so-called "overshoot" effect both for displacements and velocities (see reference [6]) is avoided for all values of ρ.

All this is of course only valid in the linear case. However, the algorithm has two other properties worth mentioning as they appear to be valuable in the nonlinear case.

The first one concerns the way the $b_{ii}(\tau)$ functions describing the distribution of the internal corrective forces $\{\overline{F}(\tau)\}$ within the time step or the corresponding average values B_{ii} are chosen.

By setting $B_{ii} = 0$ the influence of the corrective force $\overline{F}_i$ is neglected, i.e. linear behavior as described by the stiffness matrix $[\tilde{K}]$ is assumed in that zone of the structure which affects $\overline{F}_i$. If all B_{ii} are set to zero, $\{\overline{F}_1\}$ is neglected and no equilibrium iterations are necessary, which may lead to rapidly diverging results unless sufficiently small time steps are used.

If nonlinear behavior is expected and the tangent stiffness matrix evaluated at the beginning of the time step is used for $[\tilde{K}]$ then $B_{ii} = 1/3$, corresponding to $b_{ii} = (\tau/\Delta t)^2$, should be used, because the time derivatives $\dot{\overline{F}}_i(0)$ of the corrective functions at $\tau = 0$ are known to vanish.

If for $[\tilde{K}]$ the tangent stiffness matrix of an earlier time step is used, then $B_{ii} = 1/2$ may be used corresponding to the assumption of a linearly distributed corrective force $\overline{F}_i$, i.e. to $b_{ii} = \tau/\Delta t$. In the commonly used undamped version of the Newmark algorithm (with $\gamma = 1/2$ and $\beta = 1/4$) implemented with the so-called "out-of-balance-load" procedure as well as in the original iterative formulation of the Newmark method (see reference [3]) the B_{ii}'s are implicitly always set to 1/2, a rather poor choice when the actual tangent stiffness matrix is used.

In some parts of the structure, where highly nonlinear behavior is expected, it might be usefull to evaluate the $\overline{F}_i$'s not only at $\tau = \Delta t$ but also at $\tau = \Delta t/2$ or even at several points between $\tau = 0$ and $\tau = \Delta t$. This allows evaluation of the B_{ii} coefficients more exactly by simple numerical integration procedures.

The second welcome property of the algorithm is due to the fact that for each time step average values $\{P^*\}$ of the external loads $\{P(t)\}$ are used instead of considering only instantaneous values of $\{P(t)\}$ as in most other well-known algorithms. If the time step Δt is large compared to the typical period of the loads and if the load values are known in intervals smaller than Δt, quite relevant improvements can be obtained by using very little-time-consuming numerical integration procedures to evaluate $\{P^*\}$.

NUMERICAL RESULTS

The single-degree-of-freedom, numerically and physically undamped ($\rho = 1$; $\xi = 0$) examples presented below are intended to show the beneficial influence of choosing the proper B_{ii} coefficients and of correctly evaluating the average external loads $\{P^*\}$.

In the first two examples the homogeneous equation of motion is given by:

$$M \cdot \ddot{W} + F(W) = 0$$

with a constant mass M, a prescribed starting velocity $\dot{W}(o)$ and a vanishing displacement $W(o)$ at $t = 0$. In the first example a quadratic relation between F(W) and W (nonlinear elasticity) is assumed as shown in Fig. 3. In the second example the internal force F(W) is produced by five linear-elastic brittle springs

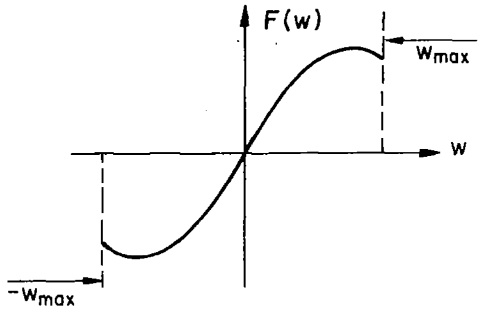


Fig. 3 : First Example
F(w)-w-Relation

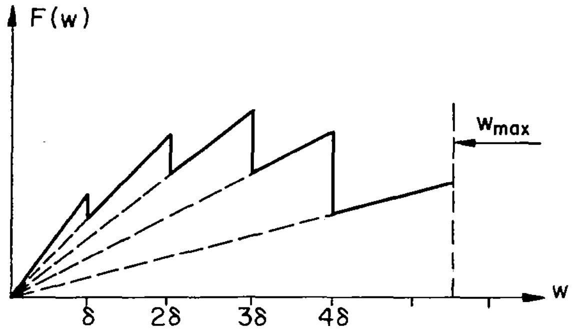
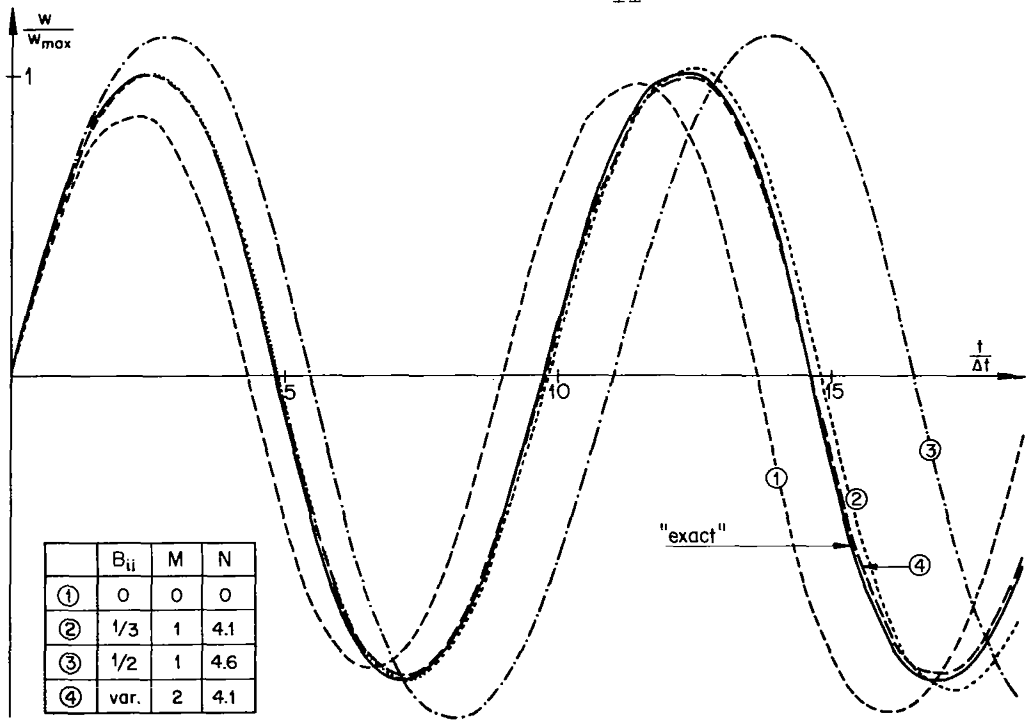


Fig. 4 : Second Example
F(w)-w-Relation

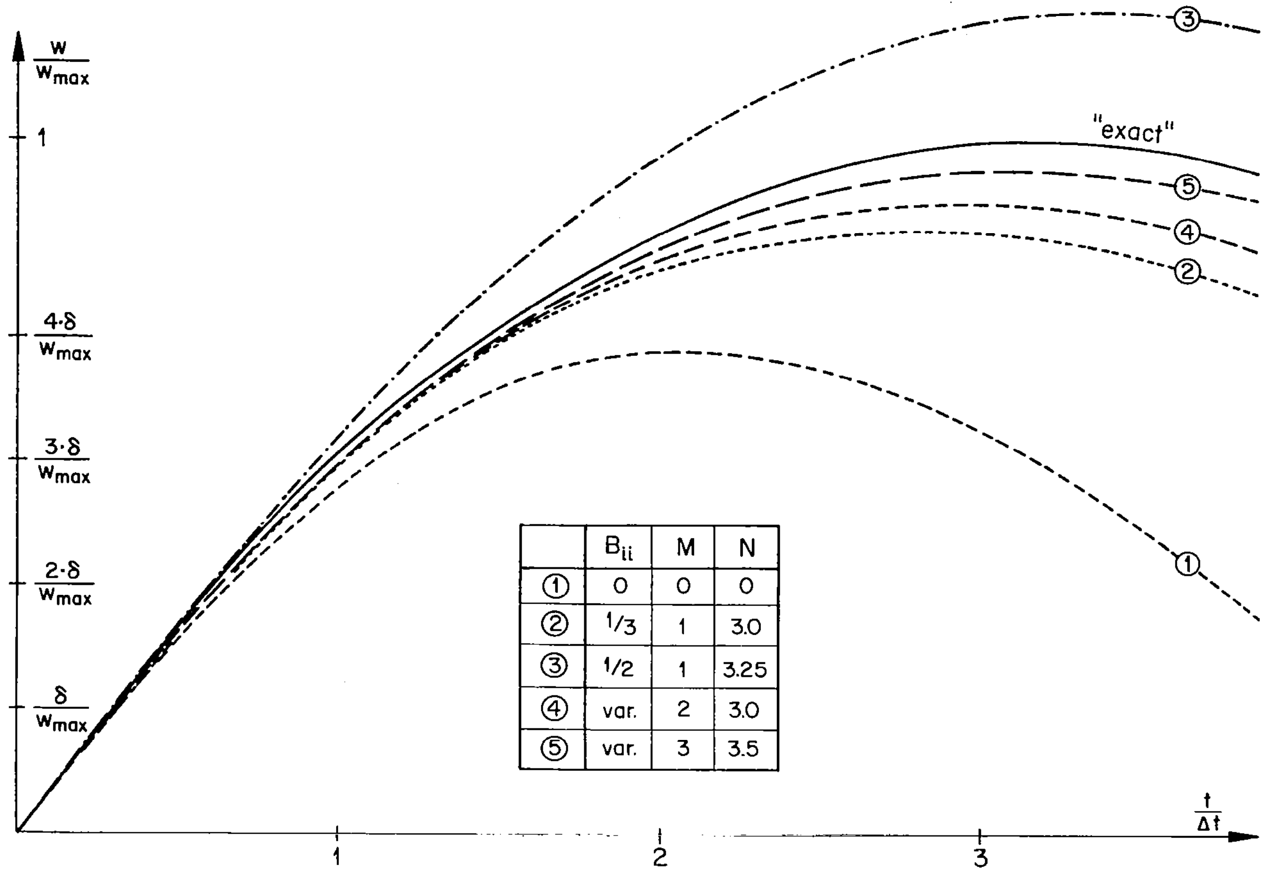
with identical stiffnesses. Four of them are assumed to break at $W = \delta, 2\delta, 3\delta$ and 4δ leading to the F-W-diagram shown in Fig. 4 remindful of crack propagation problems.

Different time histories for different B_{ii}'s are shown in Figs. 5 and 6.



	B_{ii}	M	N
①	0	0	0
②	1/3	1	4.1
③	1/2	1	4.6
④	var.	2	4.1

Fig. 5 : First Example. Time History for Several B_{ii} Coefficients



	B_{ii}	M	N
①	0	0	0
②	1/3	1	3.0
③	1/2	1	3.25
④	var.	2	3.0
⑤	var.	3	3.5

Fig. 6 : Second Example. Time History for Several B_{ii} Coefficients

The "exact" ones were obtained using very small time steps. For the others large time steps were used. The tangent stiffness (i.e. the derivative of F(W) with respect to W) was updated at each time step and equilibrium iterations (except for B_{ii} = 0) were repeated until the change of the corrective force $\overline{F}_1$ became negligible. "N" is the average number of such equilibrium iterations. "M" is the number of times the internal corrective forces $\{\overline{F}(\tau)\}$ have to be evaluated at each time step for the different B_{ii}'s: M = N = 0 for B_{ii} = 0; M = 1 for B_{ii} = 1/3 or B_{ii} = 1/2; for M > 1, B_{ii} is variable as M values of $\{\overline{F}(\tau)\}$ are used to evaluate the time integral leading to B_{ii}.

The advantage, when using the actual tangent stiffness matrix, of choosing B_{ii} = 1/3 instead of B_{ii} = 1/2 (which, as explained earlier, would correspond to the most used version of Newmark's algorithm) is evident. An even more marked improvement, as clearly demonstrated in Fig. 6, is obtained by evaluating B_{ii} more exactly (M > 1). This, however, requires time-consuming evaluations of $\{\overline{F}(\tau)\}$ for different τ's within each time step.

The third example (Fig. 7) shows the response of a linear undamped system with a natural period T_s = 4 to a sinusoidal load with a period T_p = 1. The time step used (Δt = 0.3) is very large compared with T_p but reasonable if compared with T_S. The parameter "K" given in Fig. 7 represents the number of discrete values of the load function P(t) used for the evaluation of the load average P* within each time step (K = 2 if only $P(t_0)$ and $P(t_1)$ are used as in most well-known algorithms). Of course convergence to the exact solution due to the large time step used is not obtained; nevertheless the advantage of correctly evaluating P* is evident, the computational effort needed being negligible.

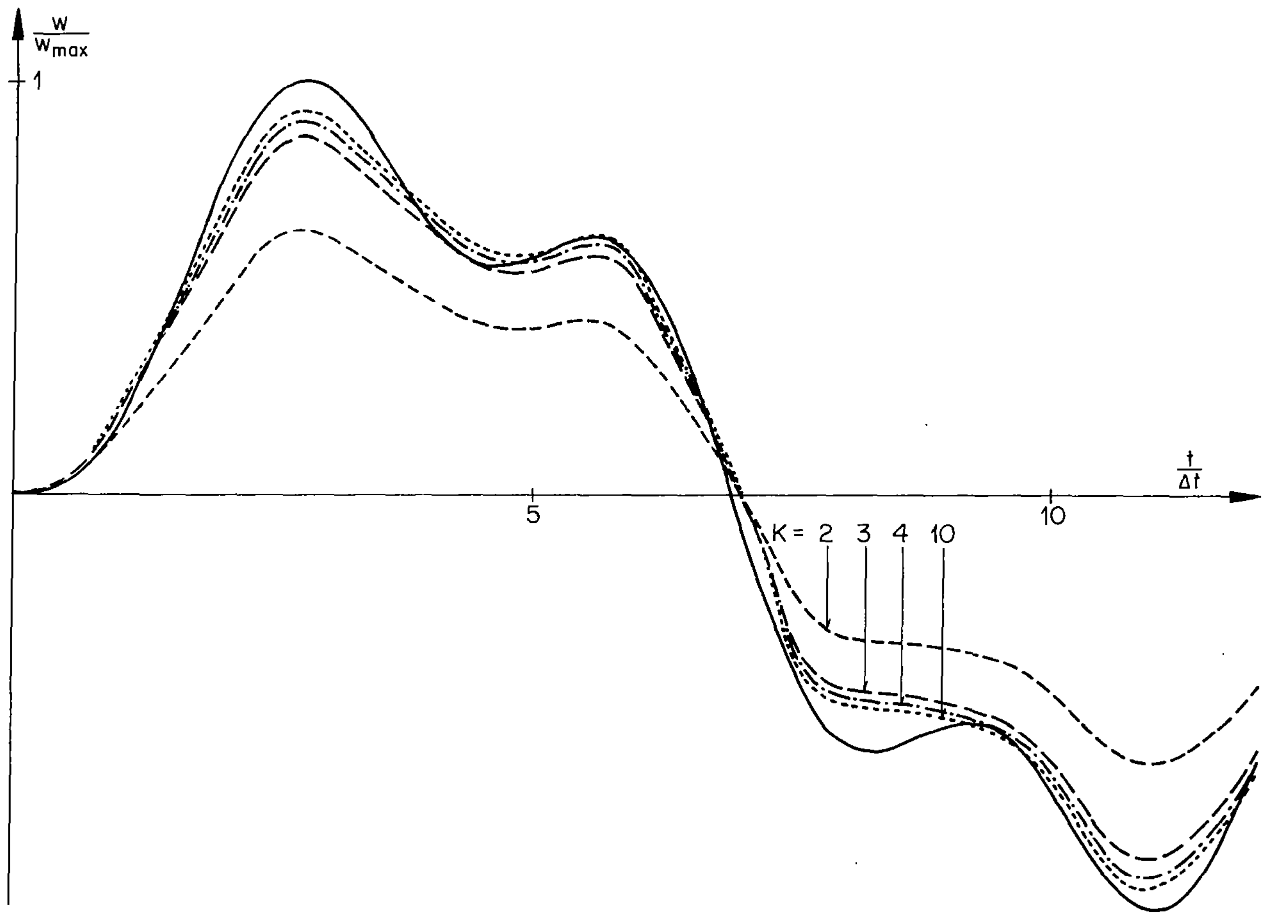


Fig. 7 : Third Example. Time History for Different Evaluations of $\int_0^{\Delta t} P(\tau)\,dt$

EXPLICIT METHOD

An explicit algorithm closely related to the implicit one discussed above has also been derived. The main difference lies in the way the integral of the internal forces $\{F(\tau)\}$ within the time step is evaluated:

$$\int_0^{\Delta t}\{F(\tau)\}\cdot dt \simeq \Delta t\cdot\{\tilde{F}\}$$

where $\{\tilde{F}\}$ is determined assuming a linear distribution of $\{F(\tau)\}$ as well as uniform motion within the time step:

$$\{\tilde{F}\} = \{F(\{\tilde{W}\})\} \qquad \text{with:} \quad \{\tilde{W}\} = \{W_o\} + \frac{\Delta t}{2}\{\dot{W}_o\}$$

Requiring the weighted integral of the residuals to vanish while using the same shape functions $N_i(\tau)$ and the same constant weighting function G as before, the following system of equations for $\{\Delta W\}$ is obtained:

$$([M] + \frac{\Delta t}{1+\rho}[C])\{\Delta W\} = \Delta t[M]\{\dot{W}_o\} - \frac{\Delta t^2}{1+\rho}(\{\tilde{F}\} - \{P^*\})$$

Assuming the matrices [M] and [C] to be diagonal, i.e. lumping masses and viscous dampers in the joints, these equations are trivial. The algorithm is then called explicit.

If, as for the implicit method, a modal transformation is performed, the following operator equations are found:

$$\begin{Bmatrix} Y_1 \\ \Delta t \cdot \dot{Y}_1 \end{Bmatrix} = \frac{1}{1 + \frac{2\xi\Omega}{1+\rho}} \begin{bmatrix} 1 + \frac{2\xi\Omega}{1+\rho} - \frac{\Omega^2}{1+\rho} & 1 - \frac{\Omega^2}{2+2\rho} \\ -\Omega^2 & 1 - \frac{2\xi\Omega\rho}{1+\rho} - \frac{\Omega^2}{2} \end{bmatrix} \begin{Bmatrix} Y_o \\ \Delta t \cdot \dot{Y}_o \end{Bmatrix}$$

As expected the operator matrix shows that the algorithm is only conditionally stable. For $\rho = 1$ the stability condition obtained by setting the spectral radius equal to unity is:

$$\Omega_{cr} = 2$$

or, for the more stringent "bifurcation" condition (see reference [7]):

$$\Omega_{bif} = 2 \cdot \sqrt{1-\xi^2}$$

For damped systems ($\xi > 0$) these conditions are less stringent than those given in references [7] or [8], due to the fact that here a diagonal damping maxtrix [C] is assumed. For $0.8 \leq \rho < 1.0$ the changes in Ω_{cr} and Ω_{bif} are found to be minimal. However, as in the implicit method, with $\rho < 1$ numerical damping is introduced and second order accuracy is lost. The parameter ρ has not the same meaning as before but still controls numerical damping.

In fact it is questionable if a choice different from $\rho = 1$ would make much sense when using the explicit method. Because, due to stability reasons, explicit methods require very small time steps, they are mainly used for problems where all or almost all frequencies of the finite element model have to be taken into account (e.g. for shock or wave propagation problems) so that a "filtering out" of frequencies by numerical damping is not necessary and mostly not desirable.

The main reason why ρ was left as a variable in the above derivations is due to the fact that it seems possible to combine the implicit and the explicit method following the procedure suggested by Hughes et. al. in reference [8], where their damping coefficient γ is also used in both cases.

FURTHER DEVELOPMENTS

As stated in the introduction this paper is only concerned with early results of a research project presently in progress. The following developments are planned:

1. Extensive tests of the implicit algorithm for reinforced concrete structures under earthquake loads. Different element formulations are to be compared.
2. Implementation and tests of iterative methods (e.g. overrelaxation) for the solution of the equations arising at each time step, possibly combining iterative solution steps with equilibrium iterations. As good guesses for starting iteration are available from earlier time steps, iterative methods might well prove to be quite efficient.
3. Implementation and tests of the explicit algorithm and of the coupled implicit-explicit-method as suggested by Belytschko and Mullen in reference [8] or by Hughes et. al. in reference [7]. Also in this case the use of iterative methods for the solution of the implicit equations looks promising.
4. Development of efficient, state-of-the art computer programs to be applied in practical cases. This is a critical point, the applicability of step-by-step procedures being severely limited by the great amount of computations involved.

REFERENCES

1. Belytschko, T.: A Survey of Numerical Methods and Computer Programs for Dynamic Structural Analysis. Nuclear Engineering and Design, Vol. 37, 1976, pp. 23-24.

2. Bathe, K. J.; and Wilson, E. L.: Stability and Accuracy Analysis of Direct Integration Methods. Earthquake Engineering and Structural Dynamics, Vol. 1, 1973, pp. 283-291.

3. Newmark, N. M.: A Method of Computation for Structural Dynamics. J. Eng. Mech. Div., ASCE, 1959, pp. 67-94.

4. Wilson, E. L.; Farhoomand, I.; and Bathe, K. J.: Nonlinear Dynamic Analysis of Complex Structures. Earthquake Engineering and Structural Dynamics, Vol. 1, 1973, pp. 241-252.

5. Hilber, H. M.; Hughes, T. J. R.; and Taylor, R. L.: Improved Numerical Dissipation for Time Integration Algorithms in Structural Dynamics. Earthquake Engineering and Structural Dynamics, Vol. 5, 1977, pp. 283-292.

6. Hilber, H. M.; and Hughes, T. J. R.: Collocation, Dissipation and 'Overshoot' for Time Integration Schemes in Structural Dynamics. Earthquake Engineering and Structural Dynamics, Vol. 6, 1978, pp. 99-117.

7. Hughes, T. J.; Pister, K. S.; and Taylor, R. L.: Implicit-Explicit Finite Elements in Nonlinear Transient Analysis. Proceedings of the International Conference on Finite Elements in Nonlinear Mechanics (Fenomech 78) (Stuttgart, Germany), 1978.

8. Belytschko, T.; and Mullen, R.: Explicit Integration of Structural Problems. Proceedings of the International Conference on Finite Elements in Nonlinear Solid and Structural Mechanics (Geilo, Norway), Vol. 2, 1977.

THREE-DIMENSIONAL FINITE STRIP ANALYSIS OF ELASTIC SOLIDS

M.S. Cheung and M.Y.T. Chan
Public Works Canada

SUMMARY

Three-dimensional (3-D) finite strips are formulated by combining finite element shape functions with beam eigenfunctions. Because of the orthogonality of the beam functions, three-dimensional problems are reduced to a series of two-dimensional problems, often with stiffness matrices of very narrow bandwidth. These require considerably less computer memory and computation time to solve. Isoparametric and high order finite element shape functions are used in the formulation of the 3-D finite strips. Numerical examples such as the static and free vibration analyses of simply supported thick plates are presented. Results are compared with existing solutions. Good agreements are obtained in all cases. Potential applications of the 3-D finite strips include the static and dynamic analyses of voided slabs, thick box girders and axisymmetric thick-walled shell structures.

INTRODUCTION

Although the finite element method is at present the most powerful and versatile numerical approach for structural analysis, the computing cost can often be very high. This is particularly true in the case of three-dimensional structural analyses. In an attempt to reduce the computational requirements of the finite element method, researchers have developed the finite strip technique (ref.1), a semi-analytical method that couples simple polynomial expressions for one or two directions with beam eigenfunction series for the other directions. This reduces a two-dimensional problem to one dimension, and a three-dimensional problem to two dimensions. Furthermore, because of the orthogonal properties of the eigenfunction series, the terms of the series may become uncoupled depending on the type of boundary conditions, and the stiffness matrices of each term can be formed, assembled and solved separately, resulting in a substantial reduction in computing costs. The method is suited for the analysis of structures having regular geometric plans and simple boundary conditions, and has been successfully applied to the static and the dynamic analyses of slabs, folded plate structures and box-girder bridges (ref. 1).

In this study, 3-D finite strips are formulated in two ways, one by coupling isoparametric quadrilateral and triangular plane-stress finite element shape functions with beam eigenfunctions, and the second by coupling high order quadrilateral plane-stress finite element shape functions with beam eigenfunctions. The stiffness, mass and load matrices are derived following standard finite element procedures. Three-dimensional elasticity constitutive equations are used in the derivation of the various stiffness matrices. Applications of the 3-D finite strips to some prismatic solids such as thick plates are described. Numerical integration using Legendre-Gauss or Radau-Gauss quadratures was employed in the derivations.

SYMBOLS

a_1, a_2, etc.	lengths of sides of a quadrilateral
a	span of 3-D finite strips
A	surface area of 3-D finite strips
$[B]$	matrix relating strains to displacement components
$[C]$	matrix containing finite strip displacement functions
$[D]$	material constant matrix
$\{F\},\{\bar{F}\}$	individual and assembled consistent load vectors
$\{g\}$	vector containing distributed body forces
h	thickness of plate
L_1, L_2, L_3	triangular area co-ordinates
$[K]$, $[\bar{K}]$	individual and assembled stiffness matrices
$[M]$, $[\bar{M}]$	individual and assembled consistent mass matrices
N	finite element shape functions
n	number of nodes in finite elements
$\{P\}$	vector containing concentrated nodal forces
P	point in $\xi - \eta$ space or a concentrated point load
$\{q\}$	vector containing distributed surface forces
V	volume of 3-D finite strip
u, v, w	displacement components in the x, y and z directions
x, y, z	Cartesian co-ordinates
x_ξ	distance along lines of equal η (equation 8)
$\{\delta\}$, $\{\bar{\delta}\}$	individual and assembled nodal displacement vectors
ξ, η	curvilinear co-ordinates
θ_{zi}	finite element rotational degree of freedom $=(\partial v/\partial x_\xi)_i$
θ_{yi}	3-D finite strip rotational degree of freedom $=(\partial w/\partial x_\xi)_i$
ρ	mass density
ω	circular natural frequencies

ω_{xy}	finite element skew symmetric rotational degree of freedom $=(\partial v/\partial x - \partial u/\partial y)/2$
ω_{xz}	3-D finite strip skew symmetric rotational degree of freedom $=(\partial w/\partial x - \partial u/\partial z)/2$

ISOPARAMETRIC 3-D FINITE STRIPS

A family of 3-D isoparametric quadrilateral and triangular finite strips can be developed by using the plane-stress isoparametric finite element shape functions reported by Ergatoudis (ref. 2). Considering only simply supported situations in which $u=w=\partial v/\partial y=0$ at the ends, a suitable set of displacement functions for a 3-D strip of span a (fig. 1) is

$$u = \sum_{m=1}^{\infty} \sum_{i=1}^{n} N_i u_{im} \sin\frac{m\pi y}{a}$$

$$v = \sum_{m=1}^{\infty} \sum_{i=1}^{n} N_i v_{im} \cos\frac{m\pi y}{a} \qquad (1)$$

$$w = \sum_{m=1}^{\infty} \sum_{i=1}^{n} N_i w_{im} \sin\frac{m\pi y}{a}$$

The x and z co-ordinates of the isoparametric section are defined as

$$x = \sum_{i=1}^{n} N_i x_i$$

$$z = \sum_{i=1}^{n} N_i z_i \qquad (2)$$

The most simple isoparametric quadrilateral is the four node IPLQ quardilaterial (ref. 2) which has linearly varying displacements (fig. 2a). The shape functions for this finite element are simply

$$N_i = \frac{1}{4} (1 + \xi\xi_i) (1 + \eta\eta_i) \qquad (3)$$

Other more sophisticated isoparametric quadrilateral elements of the same family include the eight node IPQQ quadrilateral (ref. 2) whose displacements vary quadratically (fig. 2b), and the twelve node IPCQ quadrilateral (ref. 2) with cubically varying displacements (fig. 2c).

For the Isoparametric triangular finite elements, the shape functions are most conveniently expressed in terms of the area co-ordinates L_1, L_2 and L_3. The first element of the series is the three node IPCST constant strain triangle (fig. 3a) whose shape functions are simply the area co-ordinates (ref. 2). Thus,

$$N_1 = L_1,\ N_2 = L_2,\ N_3 = L_3 \tag{4}$$

Using a recurrence formula (ref. 2), more refined triangular elements such as the six node IPLST linear strain triangle (fig. 3b) and the ten node IPQST quadratic strain triangle (fig. 3c) can be formulated.

Although in theory more refined elements can be derived by introducing additional nodes, such elements are often of limited practical use since they usually result in stiffness matrices having very large bandwidths.

HIGH ORDER QUADRILATERAL 3-D FINITE STRIPS

Because of its linearly varying displacements, the accuracy of the IPLQ 3-D finite strip is usually very limited. The strip could be refined by adding extra nodes as was done in the last section. However, such a procedure is not always desirable, since the bandwidths of the stiffness matrices are increased. The alternative, and probably most effective, approach is to introduce additional degrees of freedom at the nodes. Two high order plane stress quadrilateral finite elements were selected from the published literature for this purpose.

The first element is the QCC3 in-plane quadrilateral with displacements u, v and the skew symmetric rotations

$$\omega_{xy} = \frac{1}{2}\left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right) \tag{5}$$

as degrees of freedom (fig. 4a). This element was derived by Abu-Ghazaleh (ref. 3) and subsequently used by Scordelis (ref. 4) to analyze box-girder bridges.

Using the shape function of this finite element, a high order 3-D finite strip with u, v, w and the skew symmetric rotations

$$\omega_{xz} = \frac{1}{2}\left(\frac{\partial w}{\partial x} - \frac{\partial u}{\partial z}\right) \tag{6}$$

as degrees of freedom is formulated (fig. 4a). Considering only simply supported cases, the displacement functions of the finite strip can be written as

$$u = \sum_{m=1}^{\infty} \sum_{i=1}^{4} [N_{1i}\, u_{im} + N_{2i}\omega_{xzim}] \sin \frac{m\pi y}{a}$$

$$v = \sum_{m=1}^{\infty} \sum_{i=1}^{4} N_{1i}\, v_{im} \cos \frac{m\pi y}{a} \tag{7}$$

$$w = \sum_{m=1}^{\infty} \sum_{i=1}^{4} [N_{1i}\, w_{im} + N_{3i}\omega_{xzim}] \sin \frac{m\pi y}{a}$$

where the shape functions N_{1i}, N_{2i} and N_{3i} are the same as those used for the finite element in reference 3.

The other high order element selected is the plane stress QLC3 element developed by Sisodiya et al.(ref. 5). The nodal parameters of the element are $\{\delta\}_i = [u, v, \theta_z]_i^T$, with $\theta_{zi}=(\partial v/\partial x_\xi)_i$ where x_ξ is a distance along lines of equal η (fig. 4b); and at a general point P

$$x_\xi = \frac{1}{4}\ \{a_1\ (1-\eta) + a_3\ (1+\eta)\}\xi \tag{8}$$

where a_1 and a_3 are the lengths of opposite sides of a quadrilateral (fig. 4b).

Adopting the shape function of the element for the 3-D finite strip, the displacement functions of a simply supported 3-D strip can be expressed as

$$u = \sum_{m=1}^{\infty} \sum_{i=1}^{4} N_{1i}\, u_{im} \sin\frac{m\pi y}{a}$$

$$v = \sum_{m=1}^{\infty} \sum_{i=1}^{4} N_{1i}\, v_{im} \cos\frac{m\pi y}{a} \tag{9}$$

$$w = \sum_{m=1}^{\infty} \sum_{i=1}^{4} [N_{2i}\, w_{im} + N_{3i}\ \theta_{yim}] \sin \frac{m\pi y}{a}$$

where $\theta_{yi} = (\partial w/\partial x_\xi)_i$ and x_ξ is as defined before. The functions N_{1i}, N_{2i} and N_{3i} are identical to those presented in reference 5.

STIFFNESS, MASS AND LOAD MATRICES

The stiffness and load matrices can be derived through the minimization of the total potential energy, a standard finite element procedure that leads to the familar expression

$$[K]\ \{\delta\} - [F] = 0 \tag{10}$$

in which [K] is the stiffness matrix, {δ} the unknown nodal displacement vector and {F} the consistent load vector. A typical submatrix $[K_{ij}]$ of the stiffness matrix [K] is

$$[K_{ij}] = \int [B]_i^T [D] [B]_j dV \tag{11}$$

where [B] is the so-called strain matrix that relates the strains to the displacement components and [D] is the elasticity matrix for the material which can be isotropic or orthotropic. A typical submatrix $\{F_i\}$ of {F} is

$$\{F_i\} = \{P_i\} + \int [C_i]^T \{q\} dA + \int [C_i]^T \{g\} dV \tag{12}$$

where $[C_i]$ contains the nodal displacement functions and the force terms represent concentrated, surface and body forces.

Using the displacement functions defined in the previous sections, equation (11) would become

$$[K_{ij}]_{\ell m} = \int [B_i]_\ell^T [D] [B_j]_m dV \tag{13}$$

Because of the orthogonality of the series used, it can be shown that

$$[K_{ij}]_{\ell m} = \begin{cases} 0 & \text{for} \quad \ell \neq m \\ \frac{a}{2} \iint [B_i]_m^T [D] [B_j]_n dxdz & \text{for} \quad \ell = m \end{cases} \tag{14}$$

i.e., the series terms are uncoupled and off-diagonal submatrices in the stiffness matrix are null matrices.

To obtain the consistent load vector for the 3-D strips, the external applied loads are expressed in terms of series similar to those used for the displacement functions and substituted into the appropriate integrals in equation (12). Details of the derivation of the load terms can be found in reference 1.

The formula for deriving the consistent mass matrix is quite standard, and is

$$[M] = \int \rho [C]^T [C] dV \tag{15}$$

where [M] is the consistent mass matrix and ρ is the mass density of the material.

As the displacement functions are either defined in terms of the curvilinear co-ordinates ξ and η or area co-ordinates L_1, L_2 and L_3, it is necessary to rewrite the derivatives and integrals of the displacements with respect to the local co-ordinate system. This is a fairly straightforward matter involving the determination of the Jacobian matrix (ref. 2).

Once the individual finite strip stiffness, mass and load matrices are formulated, they can be assembled in the usual manner to form the static problem of

$$[\bar{K}]\ \{\bar{\delta}\} - \{\bar{F}\} = 0 \qquad (16)$$

or the free vibration problem of

$$([\bar{K}] - \omega^2\ [\bar{M}])\ \{\bar{\delta}\} = 0 \qquad (17)$$

where $[\bar{K}]$, $[\bar{M}]$, $\{\bar{F}\}$ and $\{\bar{\delta}\}$ are respectively the assembled stiffness, mass, load and displacement matrices, and ω is the circular frequency of free vibration.

ILLUSTRATIVE EXAMPLES

Static and free vibration analyses were carried out for the simply supported thick, square plate shown in fig. 5. The central deflections due to uniformly distributed load and a central point load are shown in tables 1 and 2. The values shown were obtained with ten series terms. Comparing the present results with existing finite element solutions (ref. 6) and closed form solutions (ref. 7) as well as the classical thin plate theory (ref. 8), it can be seen that the agreement is good. By doubling the number of series terms in a number of runs, it was found that the displacement values remained unchanged. The first three lowest flexural frequencies for a simply supported square plate with a thickness vs. span ratio of 0.2 are tabulated in table 3. Excellent agreement between the present frequencies and those obtained from finite element (ref. 9) and closed form (ref. 10) solutions can be seen, while the thin plate theory tends to overestimate the frequencies. The results in tables 1-3 indicate that the effects of thickness-shear deformation and rotary inertia can be accurately predicted by the present 3-D finite strip formulation.

CONCLUDING REMARKS

Three-dimensional (3-D) simply supported finite strips with quadrilateral and triangular cross sections have been formulated using isoparametric and high order finite element shape functions and beam eigenfunctions. In general, the accuracies of both the isoparametric and high order 3-D strips can be considered good since reasonably good results can be achieved even with a relatively coarse mesh.

Three-dimensional finite strips with other than simply supported boundaries can be derived by employing the appropriate beam eigenfunctions to match the boundary conditions. Curved, and circular 3-D strips can be developed by using a cylindrical co-ordinate formulation. Continuous structures can be analyzed either by coupling the finite element shape functions with eigenfunctions of continuous beams (ref. 11); or by using a finite strip flexibility approach (ref. 1). Potential applications of the 3-D finite strips include the static and dynamic analyses of voided slabs, thick box girders and axisymmetric thick-walled shell structures. These potential applications along with the above mentioned extension of the 3-D finite strip method are the topics of current investigations by the authors; and the results will be reported when they become available.

REFERENCES

1. Cheung, M.S.: Finite Strip Analysis of Structures. Ph.D. Thesis, University of Calgary, Alberta, May 1971.

2. Ergatoudis, J.G.: Isoparametric Finite Elements in Two and Three Dimensional Stress Analysis. Ph.D. Thesis, University of Wales, Swansea, Oct. 1968.

3. Abu-Gazaleh, B.N.: Analysis of Plate-Type Prismatic Structures. Ph.D. Thesis, University of Calfornia, Berkeley, 1965.

4. Scordelis, A.C.: Analysis of Continuous Box-Girder Bridges. Research Report No. SESM 66-17, University of California, Berkeley, Oct. 1967.

5. Sisodiya, R.G.; Cheung, Y.K.; and Ghali, A.: New Finite Elements with Application to Box-Girder Bridges. Proc. Instit. Civil Engrs., Supplement Volume, 1972, pp. 207-224.

6. Pryor, C.W.; Barker, R.M.; and Frederick, D.: Finite Element Analysis of Reissner Plates. J. Engg. Mech. Div., ASCE, vol. 96, EM6, Dec. 1970, pp. 963-983.

7. Salerno, V.L.; and Goldberg, N.A.: Effect of Shear Deformation on the Bending of Rectangular Plates. J. Appl. Mech., vol. 27, Mar. 1960, pp. 54-58.

8. Timoshenko, S.P.; and Woinowsky-Krieger, S.: Theory of Plates and Shells. 2nd edn., McGraw-Hill, London, 1959, pp. 120 and 143.

9. Rock, T.; and Hinton, E.: Free Vibration and Transient Response of Thick and Thin Plates Using the Finite Element Method. Int. J. Earthq. Engg. Struct. Dyn., vol. 3, No. 1, Jul.-Sept. 1974, pp. 51-63.

10. Srinivas, S.; Joga Rao, C.V.; and Rao, A.K.: An Exact Analysis for Vibration of Simply-Supported Homogeneous and Laminated Thick Rectangular Plates. J. Sound Vib., vol. 12, Feb. 1970, pp. 187-199.

11. Cheung, Y.K.; and Delcourt, C.R.: Buckling and Vibration of Thin Flat-Walled Structures Continuous over Several Spans. Proc. Instit. Civil Engrs. Part 2, vol. 63, Mar. 1977. pp. 93-103.

TABLE 1 COMPARISON OF CENTRAL DEFLECTIONS FOR SIMPLY SUPPORTED PLATES UNDER UDL

	h/a=0.05	h/a=0.1	h/a=0.2	h/a=0.25
IPLQ	3692.1	463.4	64.1	35.4
IPQQ	3694.0	482.1	65.5	36.3
IPCQ	3704.2	487.2	67.1	36.4
IPCST	3681.1	462.1	63.2	34.2
IPLST	3688.5	465.2	64.8	36.3
IPQST	3701.1	471.3	66.1	36.5
QCC3	3683.1	465.4	64.3	35.5
QLC3	3686.5	465.1	64.5	35.8
REF. 6	3575.2	461.2	64.8	35.9
REF. 7	3588.8	463.2	65.2	36.2
REF. 8	3549.6	443.7	55.5	28.4

TABLE 2 COMPARISON OF CENTRAL DEFLECTIONS FOR SIMPLY SUPPORTED PLATES UNDER POINT LOAD

	h/a=0.05	h/a=0.1	h/a=0.2	h/a=0.25
IPLQ	105.98	13.97	2.24	1.39
IPQQ	106.52	14.47	2.31	1.43
IPCQ	106.71	14.87	2.36	1.49
IPCST	104.92	13.88	2.27	1.39
IPLST	106.16	14.32	2.29	1.42
IPQST	106.89	14.77	2.32	1.48
QCC3	106.11	14.12	2.28	1.41
QLC3	106.31	14.16	2.30	1.41
REF. 6	106.49	14.77	2.46	1.47
REF. 8	101.34	12.67	1.58	0.81

TABLE 3 COMPARISON OF CIRCULAR FREQUENCIES OF A SIMPLY SUPPORTED THICK PLATE (h/a=0.2)

	MODES OF VIBRATION (NUMBER OF HALF-WAVES IN x AND y DIRECTIONS)		
	1,1	1,2	2,2
IPLQ	1.0621	2.3324	3.4000
IPQQ	1.0611	2.3251	3.2781
IPLST	1.0625	2.3351	3.3386
IPQST	1.0608	2.3284	3.2762
QCC3	1.0631	2.3342	3.4017
QLC3	1.0635	2.3289	3.3947
THIN PLATE THEORY	1.1947	2.9867	4.7788
REF. 10	1.0607	2.3291	3.3765
REF. 9	1.0565	2.3235	3.2758

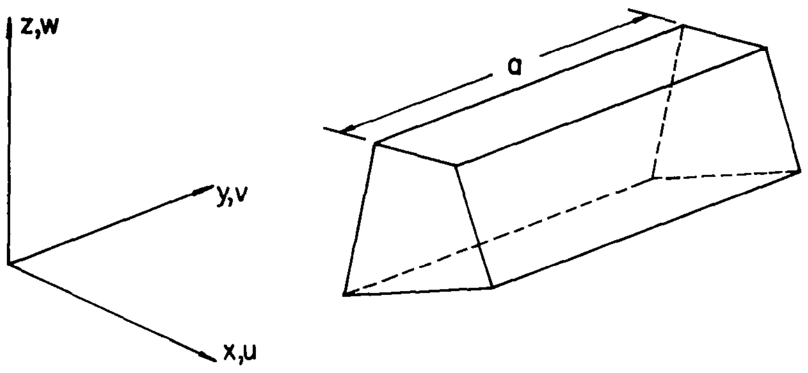


Figure 1.- A typical 3-D finite strip.

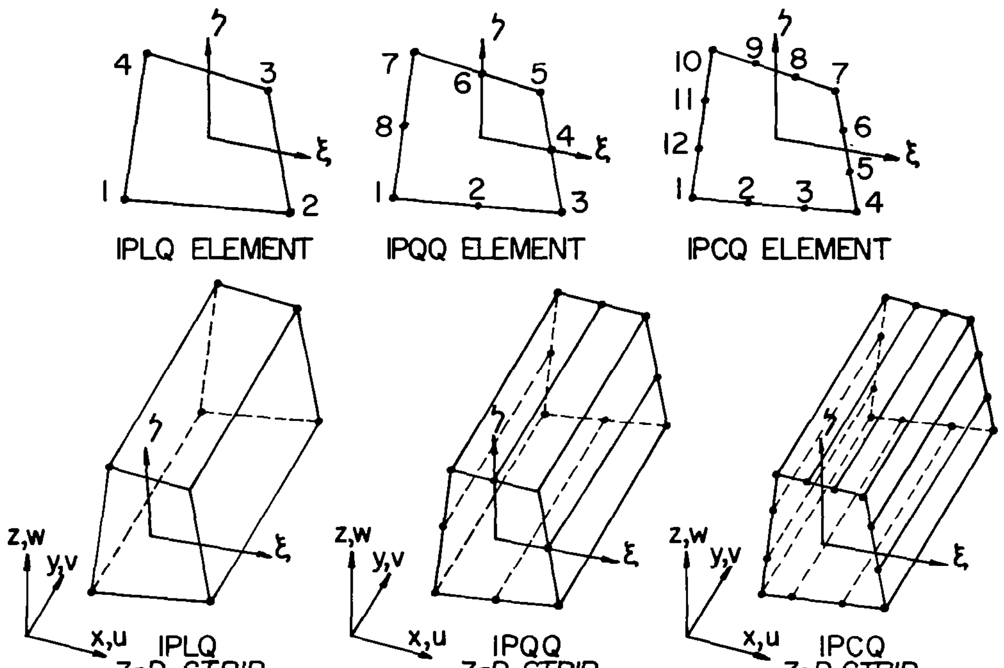


Figure 2.- Isoparametric quadrilateral 3-D finite strips.

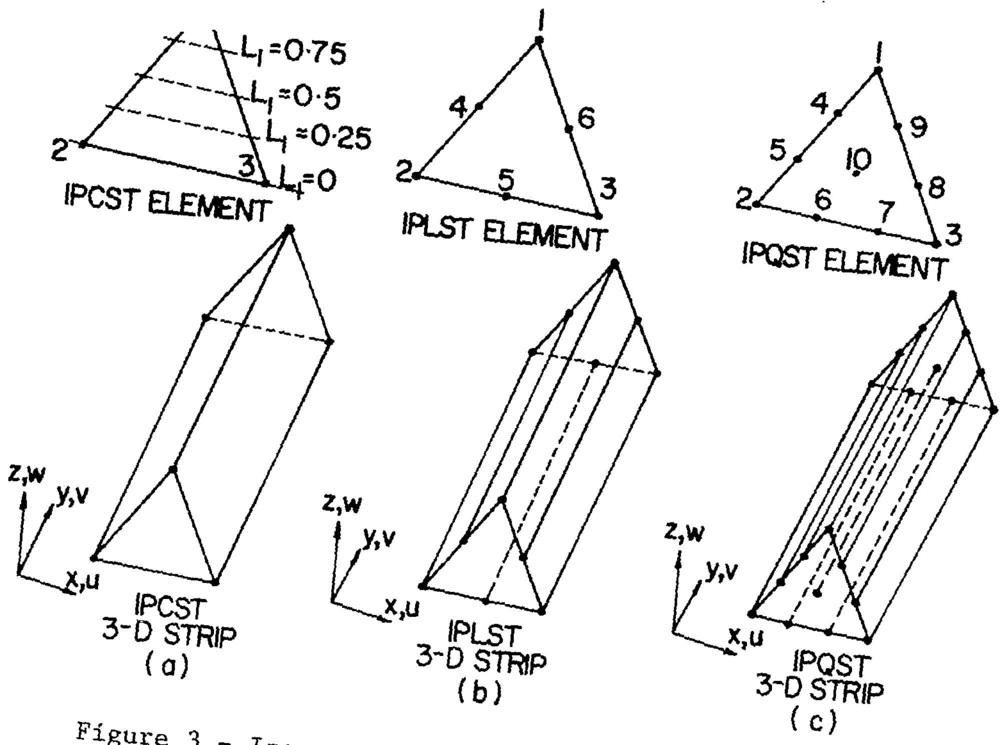


Figure 3.- Isoparametric triangular 3-D finite strips.

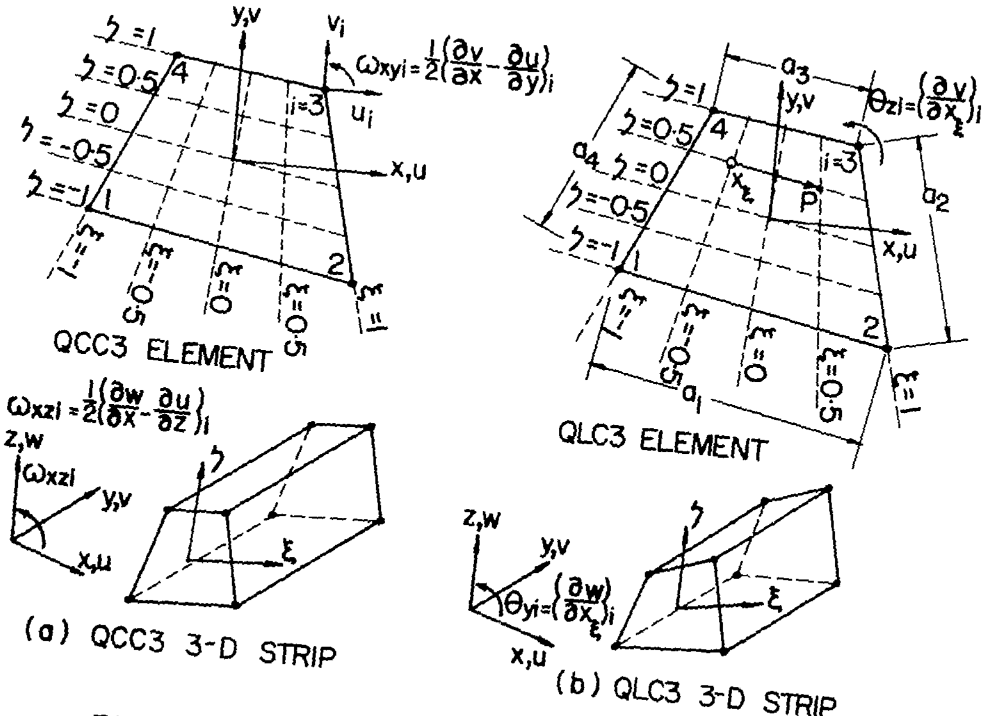
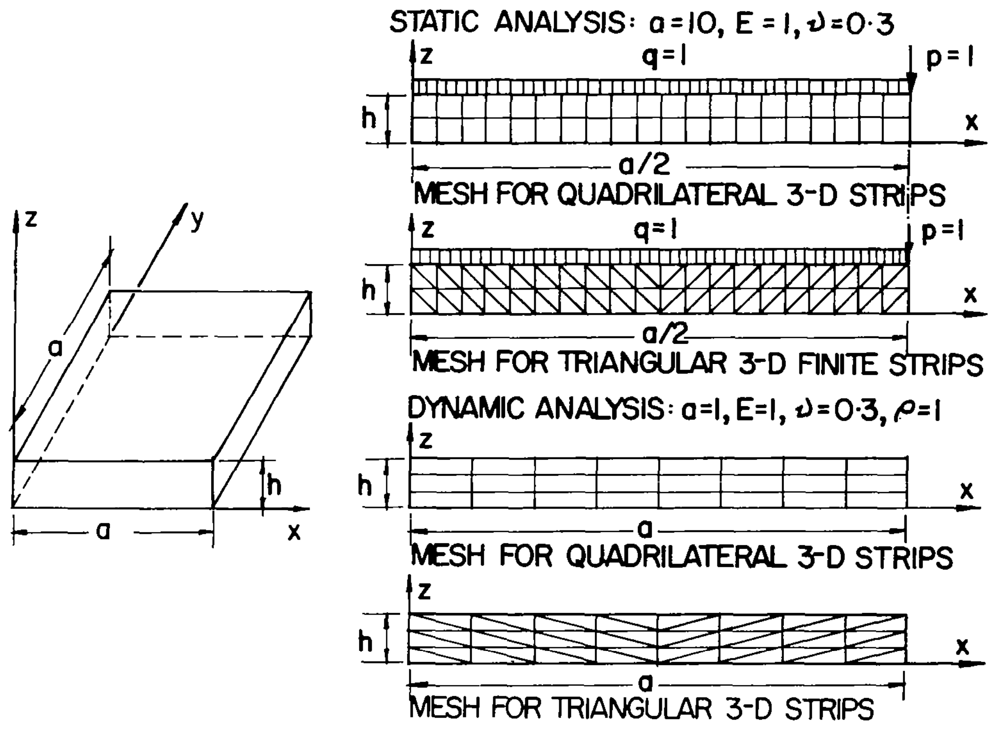

y,v
v_i
$\omega_{xyi}=\frac{1}{2}\left(\frac{\partial v}{\partial x}-\frac{\partial u}{\partial y}\right)_i$
u_i
x,u
QCC3 ELEMENT
$\omega_{xzi}=\frac{1}{2}\left(\frac{\partial w}{\partial x}-\frac{\partial u}{\partial z}\right)_i$
(a) QCC3 3-D STRIP
$\theta_{zi}=\left(\frac{\partial v}{\partial x_\xi}\right)_i$
$\theta_{yi}=\left(\frac{\partial w}{\partial x_\xi}\right)_i$
QLC3 ELEMENT
(b) QLC3 3-D STRIP

Figure 4.- High order quadrilateral 3-D finite strips.



STATIC ANALYSIS: a=10, E=1, ν=0·3
z
q=1
p=1
h
x
a/2
MESH FOR QUADRILATERAL 3-D STRIPS
z
q=1
p=1
h
x
a/2
MESH FOR TRIANGULAR 3-D FINITE STRIPS
DYNAMIC ANALYSIS: a=1, E=1, ν=0·3, ρ=1
z
h
x
a
MESH FOR QUADRILATERAL 3-D STRIPS
z
h
x
a
MESH FOR TRIANGULAR 3-D STRIPS
z
y
a
h
a
x

Figure 5.- Mesh sizes used for simply supported thick plate analyses.

THE INCORPORATION OF TRUNCATED FOURIER SERIES INTO FINITE DIFFERENCE APPROXIMATIONS OF STRUCTURAL STABILITY EQUATIONS

Steven R. Hannah
Former Graduate Student
Air Force Institute of Technology
Presently at Space and Missile Systems Organization (SAMSO)

Anthony N. Palazotto
Department of Aeronautics and Astronautics
AFIT School of Engineering

SUMMARY

A new trigonometric approach to the finite difference calculus was applied to the problem of beam buckling as represented by virtual work and equilibrium equations. The trigonometric functions were varied by adjusting a wavelength parameter in the approximating Fourier series. Values of the critical force obtained from the modified approach for beams with a variety of boundary conditions were compared to results using the conventional finite difference method. The trigonometric approach produced significantly more accurate approximations for the critical force than the conventional approach for a relatively wide range in values of the wavelength parameter; and the optimizing value of the wavelength parameter corresponded to the half-wavelength of the buckled mode shape. Thus, selection of the wavelength parameter is a simple process if the half-wavelength is known. Methods for selecting this parameter in the more general case are also presented. It was found from a modal analysis that the most accurate solutions are obtained when the approximating function closely represents the actual displacement function and matches the actual boundary conditions. It is more difficult to select a satisfactory value of the wavelength parameter for the equilibrium equation which makes the virtual work equation more attractive for practical applications. A comparison of the virtual work and the Galerkin approaches identified marked similarities between the two methods.

INTRODUCTION

Numerical analysis has played an important part in furthering the understanding of structures over the past decades. Finite element and difference are just two techniques which may be considered part of the overall numerical approach. The similarities between the two have come to light recently as they are applied to energy equations, and therefore further research into finite difference methods, with respect to structures, has been pursued. Stein and Housner (ref. 1) initiated work into a trigonometric approach to finite difference as applied to plate instability which appeared very appealing because of its convergence characteristics (ref. 2). Thus, the authors investigated this relatively new trigonometric approach and applied it to problems of beam buckling, incorporating both the virtual work and equilibrium equations by extending the method's fundamental mathematical concepts. In order to accomplish this, a trigonometric function was varied by adjusting a wavelength parameter in an approximating Fourier series. Values of the critical force obtained from the modified approach for beams with a variety of boundary conditions were, for the first time, at least to the authors' knowledge, compared to results using the conventional finite difference method. The trigonometric approach produced significantly more accurate approximations for the critical force than the conventional approach for a relatively wide range in values of the wavelength parameter; and the optimizing value of the wavelength parameter corresponded to the half-wavelength of the buckled mode shape. It was found from a modal analysis that the most accurate solutions are obtained when the approximating function closely represents the actual displacement function. It is more difficult to select a satisfactory value of the wavelength parameter for the equilibrium equations which makes the virtual work equation more attractive for practical applications. The buckled mode shape (or eigenfunction) is predicted with high accuracy regardless of the value of the wavelength parameter. A comparison of the virtual work and the Galerkin approaches identified marked similarities between the two methods.

MATHEMATICAL EQUATIONS AND NUMERICAL APPROXIMATIONS

<u>Virtual Work Equation</u>. The virtual work principle can be stated mathematically as

$$\delta W_e = \delta U \tag{1}$$

(W_e = external work; U = strain energy)

For a one-dimensional beam with an axial force P, this relationship reduces to the following equation (ref. 3):

$$EI \int_0^L \frac{d^2v}{dx^2}\frac{d^2\delta v}{dx^2}\,dx = P\int_0^L \frac{dv}{dx}\frac{d\delta v}{dx}\,dx = 0 \tag{2}$$

where v represents the vertical displacements during buckling. The derivatives in equation (2) can be replaced by trigonometric finite difference approximations:

$$v'_{i+\frac{1}{2}} = \frac{\pi}{2\lambda\,\sin(\frac{\pi h}{2\lambda})}\,(-v_i + v_{i+1}) \tag{3}$$

$$v_i'' = \frac{\pi^2}{4\lambda^2 \sin^2(\frac{\pi h}{2\lambda})} (v_{i-1} - 2v_i + v_{i+1}) \tag{4}$$

where h = mesh spacing and λ = buckle wavelength parameter. (Refer to the Appendix for complete derivation of equations (3) and (4)). Note that the virtual work equation has been evaluated using half-station approximations for the first derivative and full-station approximations for the second derivative. The trigonometric finite difference expressions represented by equations (3) and (4) reduce to the conventional polynomial expressions as λ approaches infinity since

$$\lim_{\lambda\to\infty} \frac{\pi}{2\lambda \sin(\frac{\pi h}{2\lambda})} = \frac{1}{h} \tag{5}$$

Integration in equation (2) is performed using the trapezoid rule. The virtual displacements in the resulting equation can be ordered to produce

$$\sum_{i=1}^{N} f_i(v,P)\delta v_i = 0 \tag{6}$$

For this equality to hold, the coefficients of the individual δv_i must be equal to zero which leads to an eigenvalue problem of the form

$$\sum_{i=1}^{N} g_{ij}(P)v_i = 0 \qquad j = 1,2,\cdots N \tag{7}$$

from which the critical force can be calculated.

Equilibrium Differential Equation. The elastic form of the differential equation describing the slightly bent equilibrium configuration of an initially flat beam can be stated as (see ref. 4)

$$\frac{d^4v}{dx^4} + \frac{P}{EI}\frac{d^2v}{dx^2} = 0 \tag{8}$$

The trigonometric finite difference approximation for the fourth derivative was derived incorporating a five term Fourier series with the following result:

$$v_i^{iv} = (\frac{\pi}{\lambda})^4 [\frac{1}{A_1^2}(v_{i-2} - 4v_{i-1} + 6v_i - 4v_{i+1} + v_{i+2}) + 16T_5(1 - \frac{\sin^4\theta}{A_1^2})] \tag{9}$$

where $$T_5 = \frac{A_1v_{i+2} + A_2v_{i+1} + A_3v_i + A_2v_{i-1} + A_1v_{i-2}}{A_4} \tag{10}$$

$$\theta = \frac{\pi h}{\lambda} \tag{11}$$

$$A_1 = -2 \cos \theta + 2 \tag{12}$$

$$A_2 = 2 \cos(2\theta) - 2 \tag{13}$$

$$A_3 = -4 \cos(2\theta) + 4 \cos \theta \tag{14}$$

$$A_4 = 16 \sin^2\theta - 48 \cos^2\theta + 32 \cos^3\theta + 48 \cos^4\theta - 32 \cos^5(\theta) \tag{15}$$

Equation (8) can be evaluated at each of the N grid points placed along the length of the beam, resulting in a set of equations with the following form:

$$\begin{aligned} &\frac{A}{h^2}(v_{i-2} - 4v_{i-1} + 6v_i - 4v_{i+1} + v_{i+2}) \\ &+ \frac{P}{EI}(v_{i-1} - 2v_i + v_{i+1}) = 0 \; ; \; i = 1,2,\cdot\cdot N \end{aligned} \tag{16}$$

where A represents the combined coefficient resulting from the evaluation of equation (9). The displacements in this set of equations can be ordered to produce an eigenvalue problem in the form of equation (7).

Virtual Work and Galerkin Approaches. To complete the discussion of finite differences, it is instructive to compare the virtual work approach, as applied in this paper, to the Galerkin approach. Both methods are based on the principle of minimum potential energy. The Galerkin approach approximates the displacement function by the following series expansion:

$$v(x) = \sum_{i=1}^{M} a_i \phi_i(x) \tag{17}$$

where a_i are undetermined coefficients and ϕ_i represent continuous functions. By substituting this approximating function in the potential energy expression and performing a sequence of variational operations, the following system of equations is obtained:

$$\sum_{i=1}^{M} f_{ij}(P)a_j = 0 \qquad i = 1,2\cdot\cdot\cdot\cdot M \tag{18}$$

where f_{ij} denotes a functional relationship in terms of the external force. The concept underlying the Galerkin approach is based on the fact that the error in the approximation of equation (17) is minimized for any value of M if the a_i are chosen such that equations (18) are simultaneously satisfied (ref. 5). The virtual work approach incorporating finite difference approximations can be described in a similar manner. The Fourier coefficients, T_i, are implicitly selected (through the computational properties of the virtual work algorithm) such that the error due to the approximation of v(x) is minimized. This hypothesis was substantiated numerically by an analysis of the Fourier series approximating function.

CONCLUSIONS AND RESULTS

This paper compares the results of using the trigonometric and conventional approaches to the finite difference calculus in order to solve the equilibrium and virtual work equations. A wide range of boundary conditions were investigated. Various values of λ were used in the trigonometric approach to determine the optimum value as well as to determine the range over which the trigonometric approach gives more accurate approximations than the conventional approach. In addition, an in-depth search was conducted to provide plausible explanations for the superiority of one method over the other and one value of λ over other values. Finally, the effect of decreasing the number of grid points and the use of full-station approximations in the virtual work equation were investigated.

The virtual work method was found to be an efficient and simple approach and provided excellent results for both the trigonometric and conventional techniques with as few as five grid points. As predicted from theory, computational data revealed that the magnitude of error in computing P_{cr} varies directly with the square of the grid size. The variable input parameter, λ, has the effect of adjusting the wavelength of the Fourier series approximating function, and the optimum value of λ corresponds to the half-wavelength of the buckled mode shape for each boundary condition. There is a range in the values of λ for which the trigonometric approach is superior. This range extends from approximately 25% below the optimum value to infinity. Thus, a large value of λ is guaranteed to provide more accurate results than the conventional approach. Of course, if λ is chosen to be too large, the error from the conventional and trigonometric techniques approaches the same value; and the benefit of using the trigonometric technique is lost. It can be shown that $\lambda=1.5L$ produces satisfactory results for all boundary conditions. An illustration of this can be observed in figures 1 and 2 which depict the error in the calculation of P_{cr} for pinned-pinned and free-guided beams. A potential explanation for the superiority of one method over the other was found from an investigation of the Fourier and Taylor series approximating functions. The λ value which yields the closest series approximation to the theoretical displacement function corresponds to the optimum value of λ. That is, the key to calculating an accurate estimate of the critical force is to supply an approximating function which very closely reproduces the buckled mode shape as well as to satisfy the geometric and force boundary condition.

The similarities between the virtual work technique as employed in this paper and the Galerkin approach have been explored. In both cases, equilibrium expressions are used to derive potential energy relationships; and the displacement functions are approximated by series expansions. In addition, there is a strong relationship between the resulting sets of equations developed by the two methods. Both methods attempt to minimize the error in approximating the displacement function. When the approximating function is altered such that this minimized error is larger, the error in the computed critical force will increase proportionally. This concept was demonstrated by the use of the full-station finite difference approximation for the first derivative. The decreased accuracy in the approximation of v(x) caused a significant increase in the error of the calculated value of the critical force.

The equilibrium approach was also found to be efficient, and excellent results were obtained using the trigonometric technique. Figures 3 and 4 depict the results of calculating the critical force for pinned-pinned and free-guided beams. However, it is more difficult to select an effective value of λ using this approach than was found to be true for the virtual work approach. There are two optimum values of λ due to the fact that five terms were used in the Fourièr series for the derivation of the fourth derivative. The first value corresponds to the half-wavelength of the buckled mode shape. A precise estimate of the buckled wavelength is required in this case, however, since there is little margin for error. The range around this optimum value for which the trigonometric approach is superior to the conventional approach is very small, and the error builds rapidly as estimates of the optimum value worsen. There is a large range around the second optimum value for which the trigonometric approach is superior. This range extends from approximately 27% below the optimum value to infinity. This provides a comfortable margin of error for selecting λ. The problem is that there is no known physical parameter from which this second optimal value can be estimated other than the realization that each term of the series is superimposed upon each other, yielding a wavelength equal to the theoretical value at this particular λ. It appears from the available data that a value which is 2.75 times the half-wavelength provides a reasonably close estimate in most cases, but specific boundary conditions vary considerably from this figure. Despite the uncertainty, it is much safer to attempt an estimate of the second optimal value of λ due to the larger error margin. An attempt to use the first optimal value is probably unwise unless the buckled mode shape is known a priori with reasonable accuracy. It can be shown that λ=3.75L provides more accurate results than the conventional approach for all boundary conditions.

In comparing the results of the virtual work and equilibrium approaches, many similarities were noticed despite the major conceptual differences in the derivation of these methods. The interpretation of the wavelength parameter, λ, is the same in both cases as already discussed. In addition, the virtual work and equilibrium methods give the same value for P_{cr} when conventional finite difference expressions are used. The two methods do not give the same result when trigonometric expressions are used due to the presence of the two additional Fourier series terms in the equilibrium equation. Several major differences were also noted in the two methods. For example, it is more difficult to predict the optimum value of λ for the equilibrium approach. Additionally, it was found that an error in the estimate of λ produces a larger error in the computed value of P_{cr} for the equilibrium method than for the virtual work method. For these reasons, the virtual work method is recommended for general use over the equilibrium method. The trigonometric approach to the finite difference calculus is recommended over the conventional approach, particularly in those cases when the shape of the displacement function is known within rather broad tolerances.

REFERENCES

1. Stein, M and Housner J.: Numerical Analysis and Parametric Studies of the Buckling of Composite Orthotropic Compression and Shear Panels. NASA TN D-7996, Oct. 1975.

2. Deschler, W. and Palazotto, A.N.: Comparing Trigonometric and Conventional Finite Difference Approximations with the Finite Element Method for Plate Buckling. Proceedings of the Symposium on Applications of Computer Methods in Engineering, Aug. 23-26, 1977, pp. 137-149.

3. Laursen, H.J.: Structural Analysis. McGraw-Hill Book Company, New York, N.Y., 1969.

4. Timoshenko, S.P. and Gere, J.M.: Theory of Elastic Stability. Second Edition, McGraw-Hill Book Company, New York, N.Y., 1961.

5. Almroth, B.O. and Brush, D.O.: Buckling of Bars, Plates, and Shells. McGraw-Hill Book Company, New York, N.Y., 1975.

APPENDIX

Trigonometric finite difference approximations are derived in a manner similar to the conventional expressions with the exception that the following form of the Fourier series is used rather than the Taylor series:

$$v(x) = T_1 + T_2 \sin \frac{\pi(x - x_o)}{\lambda} + T_3 \cos \frac{\pi(x - x_o)}{\lambda} \qquad (19)$$

The derivative of equation (19) evaluated at the reference point, x_o, is given by

$$v'(x_o) = T_2 \frac{\pi}{\lambda} \qquad (20)$$

and

$$T_2 = v'(x_o) \frac{\lambda}{\pi} \qquad (21)$$

Evaluation of equation (19) at $x_o + h/2$ and $x_o - h/2$ results in

$$v_{+\frac{1}{2}} = T_1 + T_2 \sin \frac{\pi h}{2\lambda} + T_3 \cos \frac{\pi h}{2\lambda} \qquad (22)$$

$$v_{-\frac{1}{2}} = T_1 - T_2 \sin \frac{\pi h}{2\lambda} + T_3 \cos \frac{\pi h}{2\lambda} \qquad (23)$$

Subtract equation (23) from equation (22) to obtain

$$v_{+\frac{1}{2}} - v_{-\frac{1}{2}} = 2T_2 \sin \frac{\pi h}{2\lambda} \qquad (24)$$

If equation (21) is substituted in equation (24) and the terms rearranged, the following expression is obtained

$$v'(x_o) = \frac{1}{\hat{h}} (v_{+\frac{1}{2}} - v_{-\frac{1}{2}}) \tag{25}$$

where

$$\hat{h} = \frac{2\lambda \sin(\frac{\pi h}{2\lambda})}{\pi} \tag{26}$$

The trigonometric finite difference approximation for the second derivative can be obtained in a similar manner. Equation (19) can be evaluated at $x = x_o + h$ and $x = x_o - h$ to provide

$$v_{+1} = T_1 + T_2 \sin \frac{\pi h}{\lambda} + T_3 \cos \frac{\pi h}{\lambda} \tag{27}$$

$$v_{-1} = T_1 - T_2 \sin \frac{\pi h}{\lambda} + T_3 \cos \frac{\pi h}{\lambda} \tag{28}$$

By adding equations (27) and (28) and subtracting two times equation (19) evaluated at $x = x_o$, the following expression is obtained:

$$v_{+1} - 2v_o + v_{-1} = 2T_3 (\cos \frac{\pi h}{\lambda} - 1) \tag{29}$$

The second derivative of equation (19) with respect to x evaluated at $x = x_o$ is

$$v_o'' = -T_3 \frac{\pi^2}{\lambda^2} \cos \frac{\pi (x_o - x_o)}{\lambda} \tag{30}$$

Solving for T_3 yields

$$T_3 = - (\frac{\lambda}{\pi})^2 v_o'' \tag{31}$$

If equation (31) is substituted in equation (29) and the resulting expression solved for v_o'', the following expression is obtained:

$$v_o'' = \frac{\pi^2}{4\lambda^2 \sin^2(\frac{\pi h}{2\lambda})} (v_{+1} - 2v_o + v_{-1}) \tag{32}$$

or

$$v_o'' = \frac{1}{\hat{h}^2} (v_{+1} - 2v_o + v_{-1}) \tag{33}$$

Note that the trigonometric and conventional finite difference expressions are similar with the mesh spacing, h, simply replaced by $\hat{h}$.

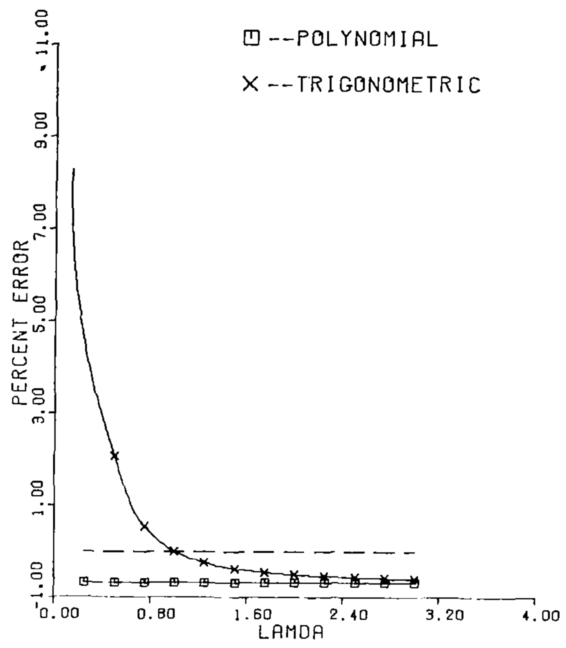


FIG. 1 ACCURACY COMPARISONS FOR P_{CR} USING THE VIRTUAL WORK EQUATION, PINNED-PINNED BOUNDARIES.

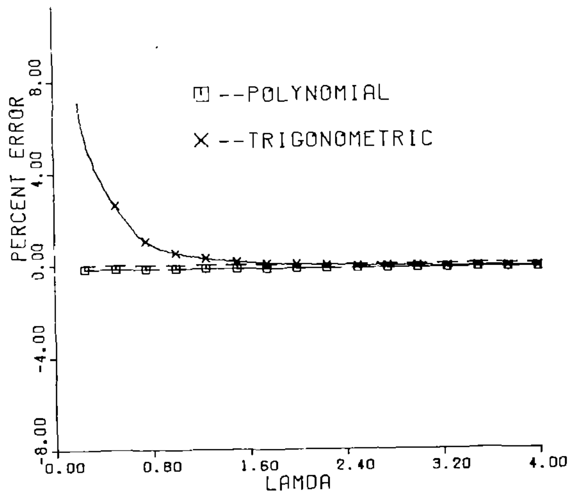
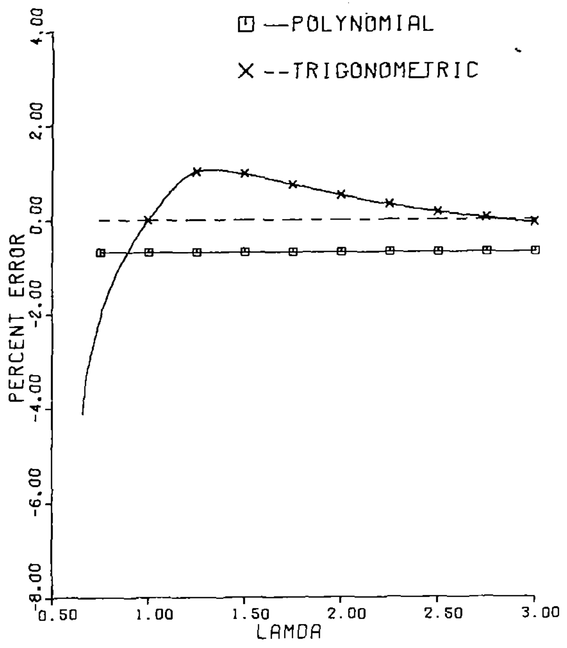
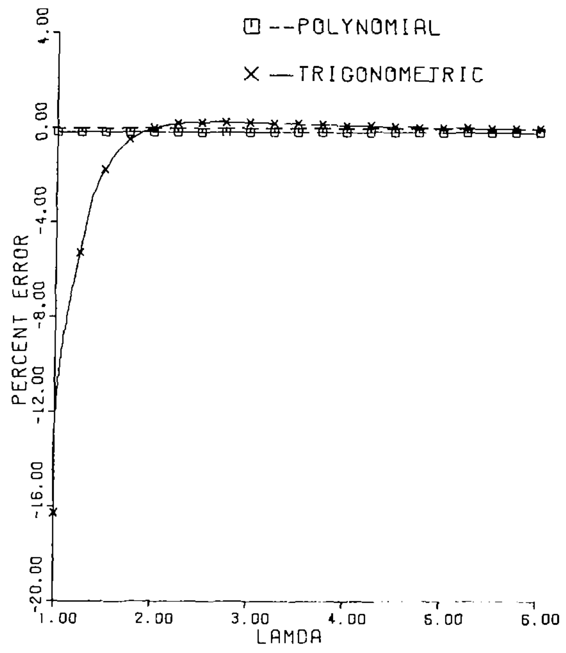


FIG. 2 ACCURACY COMPARISONS FOR P_{CR} USING THE VIRTUAL WORK EQUATION, FREE-GUIDED BOUNDARIES.



FIG, 3 ACCURACY COMPARISONS FOR P_{CR} USING THE EQUILIBRIUM EQUATION. PINNED-PINNED BOUNDARIES.



FIG, 4 ACCURACY COMPARISONS FOR P_{CR} USING THE EQUILIBRIUM EQUATION. FREE-GUIDED BOUNDARIES.

STRUCTURAL ANALYSIS CONSULTATION USING ARTIFICIAL INTELLIGENCE*

R. J. Melosh and P. V. Marcal
MARC Analysis Research Corporation

Les Berke
Flight Dynamics Laboratory, Wright Patterson Air Force Base

ABSTRACT

This presentation reports on implementation of engineering consulting for structural analysis using the concepts of artificial intelligence. It describes a knowledge base for the consultation and illustrates the use in sample engineering problems.

The primary goal of consultation is definition of the best strategy to deal with a structural engineering analysis objective. The knowledge base to meet the need is designed to identify the type of numerical analysis, the needed modeling detail, and specific analysis data required. Decisions are constructed on the basis of the data in the knowledge base - material behavior, relations between geometry and structural behavior, measures of the importance of time and temperature changes - and user supplied specifics - characteristics of the spectrum of analysis types, the relation between accuracy and model detail on the structure, its mechanical loadings, and its temperature states.

Existing software demonstrates the feasibility of the approach, encompassing the 36 analysis classes spanning nonlinear, temperature affected, incremental analyses which track the behavior of structural systems. It provides consultation, in an interactive environment, which can identify an effective analysis strategy in consultation times ranging from two to twenty minutes.

INTRODUCTION

The choice of the analysis strategy to use in computer simulation of a structural-mechanical system is important and difficult. The choice is important because it can affect required human and computer resources for engineering by an order of magnitude. The choice is difficult because there are a wide variety of analysis strategies and tactics.

*This research was sponsored by the Advanced Research Project Agency in conjunction with the Flight Dynamics Laboratory of Wright-Patterson AFB, Dayton, Ohio. Work was performed in cooperation with Bob Englemore, Lew Creary, and Jim Bennett of the Stanford Heuristic Programming Laboratory.

The concepts of artificial intelligence which encompass knowledge-based consultation (heuristic programming) offer a basis for automating the decision process. The central idea is to imbed the knowledge needed for the decision in a data base which can be particularized and manipulated during a user-computer interchange. In this "consultation", the computer system plays the role of consultant.

Heuristic programming can be based on a production rule system. Here the data base is made up of a collection of rules. Each of the rules describes some aspect of the knowledge of the "world" being considered. The rules are stated as a number of premises followed by conclusions. This results in an open system where the connectivity between the rules is implied and sought out by the program logic.

The consultation proceeds from rule to rule. At each rule every premise is resolved by means of previously accumulated consultation data or by questions to the user.

The approach has been used successfully in a number of disciplines. Feigenbaum, Buchanan, and Lederberg [1] describe a chemical spectrometry consultant. In this case the software leads to evaluating the chemical nature of a substance working from mass and nuclear spectrometry input. Shortliffe [2] describes a bacterial infectious consultant, MYCIN. This software addresses diagnosing infections based on patient data and prescribing treatments with the best prospects of success for the patient.

This paper describes an implementation of the concepts for selecting analysis strategy and tactics using a general purpose computer program like MARC [3]. It characterizes the knowledge base used for consultations and illustrates use of the base in a consultation.

STRUCTURAL MECHANICS CONSULTATION

The Mathematical Model

In order to provide the flexibility to take into account different behavior patterns in different portions of the structure, the model data is thought of as a number of substructures or regions. The behavior pattern may be estimated by using either a simple model or by direct questions to the user. For this reason the substructure is not restricted to a particular area of a structure but may overlay with other substructures.

In this pilot project we have provided simple models which depend on formulas and we require that the burden of describing the substructure data be placed on the user. It is, however, easy to visualize a scenario in which an increasing number of rules are added until a stage is reached where the consultant takes over most of the job of model idealization.

The consultant asks for data defining the material, general geometry, and boundary conditions for each substructure. It uses these data and its mathematical models to estimate stresses and deflections for the substructure. The behavior of the complete structure is determined as the peak relative stress and deflection behavior of all the substructures. Based on these peak responses, knowledge of the available analysis strategies, and user defined analysis requirements, the consultant recommends an analysis approach.

The substructure is a geometrically contiguous region of the structure composed of a unique material and with a unique set of kinematical boundary conditions. With this definition, the user can reduce his structure to substructures in a number of ways with the objective of insuring that he represents the most aggravated stress and displacement conditions.

Figure 1 illustrates some of these possibilities. Figure 1(a) depicts the conventional substructure concept of finite element analysis. Here the structure is divided into nonoverlapping regions. Every part of the structure falls into a substructure or onto a boundary shared by substructures. Figure 1(b) shows substructuring using overlapping substructures and exclusion of parts of the structure from a substructure. Figure 1(c) illustrates a decomposition into two particular parts of the structure to permit selecting the peak responses from two different models of the substructure's kinematic boundary conditions.

The engineer indicates the overall geometry and kinematic boundary conditions of an envelope model of the substructure. He describes geometry by defining the length, width, and (indirectly) depth of a rectangular prism which can just enclose the substructure. He indicates edges of the prism, which are supported either by adjacent substructures or external restraints. The engineer synthesizes the total loading for a substructure from one or more loadings. He constructs a loading from a number of point and/or distributed loading components.

Using this data, the consultant models the substructure as either a network or a continuum. Network models imply beamlike behavior. Continuum models imply platelike behavior. The cross section of a substructure may be treated as solid or thin-walled. In a solid section, all the material in the section resists loading. In a thin-walled section, that part of the material resisting loading is centered near the boundaries. A solid bar and a hollow tube illustrates the solid and thin-walled section, respectively.

Table 1 defines some of the formulas used for the plate model. These formulas estimate peak stresses and relative deflection considering the number of edges supported, the geometry of the panel, the material stiffness, the form of the cross section and the location and magnitude of loadings.

Figure 2 shows the relationship of the parts of the structural model. The stresses and deflections due to each loading component are added to determine stresses and deflection bounds for a particular loading. Behavior of loadings is combined assuming that each loading is statistically independent to arrive at limiting response estimates for each substructure. The analysis strategy is then determined by considering the most severe stress state and deflection change for any of the substructures of the structure.

Consultation Knowledge Base

The existing knowledge base provides for selecting one of 36 analysis strategies. These encompass nonlinear analysis of structures whose equilibrium equations are time independent and imply that the structure is fabricated and loaded at room temperature (21 C). If nonlinear analysis is not a constructive conclusion, the consultation recommends linear analysis.

Table 2 names the specific analysis strategies distinguished in the knowledge base. Distinction allows the user to consider substructures to be formed of any one of eight materials (three grades of aluminum, three of steel and two of concrete). Each substructure may be approximated by one of three construction models, have one of four support conditions, and be loaded by any number of distributed and/or point loads using any number of loading components to represent a loading.

The knowledge base consists of about 170 rules. These lead to valuing up to 140 consultation parameters. Using this data base, a typical consultaton (2 substructures, 3 loadings, 2 load components) requires about 25 minutes at an interactive terminal.

Table 3 defines the principal parameters of the consultation and their relation to the context tree. Valuing these parameters leads to values for the primary solution strategy variables:

Type of nonlinearity:	Geometric, material, both, boundary nonlinearity, geometric and boundary, material and boundary, all.
Itegrity goal:	Behavior, stability, both.
Integrity concern:	Local (stress exceedance, cracking...), global (deflection exceedance, stiffness degradation...).
Loading Type:	Cyclic, noncyclic.

Other parameters of the knowledge base define consultation interpretations for mathematical operations associated with the anatomical model.

A typical rule used in the consultation is given below. A rule consists of one or more if statements followed by one or more conclusions. The rule shown illustrates how considerations of accuracy, stress, and number of loading cycles interact with known data in determining the types of behavior which will justify analysis.

Typical Consultation Rule

- If the material is high strength steel
- If substructure non-dimensional stress is greater than .7
- If required analysis accuracy is less than 30 percent
- If number of cycles of loading is less than 10,000
- Conclude fatigue is a problem for the substructure

COMPUTER PROGRAM LOGIC

A computer program directs the manipulation of the rules and the engineer-consultant dialogue. The rule manipulation is sequenced to fulfill the consultation goal rule which requires that all data needed by the rules be accumulated before an opinion is offered. Through the dialogue, the consultant obtains data from the engineer whenever previously supplied user data or rule conclusions are unavailable.

The dialogue also permits the engineer to interrogate the consultant. The engineer can prompt an explanation of why a particular piece of data is required, how a particular conclusion was reached, and what rules and conclusions are available. Thus he can determine whether the knowledge base is appropriate for the particular problem in mind.

The principal tasks implemented by the computer code are as follows:

1. Determine areas of interest by keywords.

2. Find a rule with a conclusion pertaining to the keyword of interest.

3. Initiate processing of this rule:

   A. Process the next premise. If the premise is negative, terminate processing of rule and return to 2. If all premises have been processed, store the conclusion and proceed to 4.

   B. Check if this premise can be processed by using data and conclusions acquired from previous rules. If yes, conclude the processing of premise and return to A. If no, proceed to C.

   C. Ask for the required data from the user and return to A. If no, return to process another keyword. If all rules pertaining to the keyword have been processed, proceed to 5.

4. Print out conclusions of the consultation.

The following additional optional tasks may be requested by the user:

5. Develop an explanation of the reason for any question being asked by the consultant. (This requires particular printing of information on the current argument context and the particular rule being exercised.)

6. Explain how a particular conclusion was reached. (This requires reviewing the history of the consultation with respect to rules used in arriving at the conclusion.)

7. Detail what keywords are handled by the program. (This requires scanning and printing of conclusions in the knowledge base.)

8. Detail each rule. (This requires printing each rule when it has been fully exercised so the engineer can appraise the logic with respect to his problem.)

9. Allow an expert in the technical area, who knows little about the coding of the program, to enter and/or change rules in the course of building experience with the program.

ILLUSTRATIVE CONSULTATION

The following provides the computer printout for a consultation.

The program asks preliminary questions about the total structure. These cause the engineer to particularize his analysis goals and insure that his problem falls within the consultant's ken.

It is then followed by the interchange involved while the analyst described the loadings and three load components to be considered for each of the two substructures of the total structure.

The last paragraph shows the interchange as the consultation is completed. The exchange is terminated when the analyst indicates all substructures have been treated. The answering remarks define the analysis classification and special analysis tactics that should be used.

```
  <SCP>TYPESCRIPT.511   Fri 10-Mar-78 7:34AM

March 10, 1978 7:31AM in <SCP>MARC.SAV:9
Loading <SCP>STRUCTURE.511...

---------------------------------------------------------------------

10-Mar-78 07:31:36
Enter date and time of consultation (CR for today's date):
**

     --------STRUCTURE-5--------

     1)  What is the name of STRUCTURE-5?

     **  AIRPLANE WING

     2)  Assuming that your characterization of airplane wing in terms of  its
         geometry,  material  characteristics,  and  boundary  conditions  are
         accurate, how much error (in percent) is tolerable for the  analysis?

     **  10

     3)  Do you want to examine the responses of airplane  wing,  to  evaluate
         its unstability, or both?

     **EXAMINE-RESPONSES

     4)  Does airplane wing have any time dependent terms in its equations  of
         equilibrium?

     **NO

     5)  Is heating or cooling of interest in determining structure responses?

     **NO

     6)  Have you  decomposed  airplane  wing  into  a  number  of  potential
         substructures which are relevant to this analysis?

     **YES

                    --------SUBSTRUCTURE-1--------

     7)  What is the name of SUBSTRUCTURE-1?

     **  WING TIP
```

8) What is the material composing most of wing tip

(SUBSTRUCTURE-1)?

** HIGH-STRENGTH-ALUMINIUM

9) Enter the average longest dimension of wing tip (SUBSTRUCTURE-1) in inches (or <number> CMS):

** 40

10) Enter the average thickness of wing tip (SUBSTRUCTURE-1) in inches (or <number> CMS):

** 5

11) Enter the weight of wing tip (SUBSTRUCTURE-1) in pounds (or <numbers> KGS):

** 83

12) Would you describe the construction of wing tip (SUBSTRUCTURE-1) to be truss and/or frame elements (NETWORK), sheets and/or blocks of material (CONTINUUM), or a combination of both?

** CONTINUUM NETWORK

13) What is the configuration of support for wing tip (SUBSTRUCTURE-1)?

** ONE-SIDE

14) Can wing tip (SUBSTRUCTURE-1) be modelled as a one, two, or three dimensional system?

** 2

15) Is the overall geometry of wing tip (SUBSTRUCTURE-1) curved or planar?

** CURVED

The first loading applied to wing tip (SUBSTRUCTURE-1) will be called:

--------LOADING-1--------

16) What is the name of LOADING-1?

```
** AIR PRESSURE

17) Enter the number of loading cycles to be applied:

** 1

The first load component associated with LOADING-1 will be called:

  --------LOAD-COMPONENT-1--------

18) Would you describe LOAD-COMPONENT-1 as being DISTRIBUTED over most of
    the substructure or as acting at a POINT of the substructure?

** DISTRIBUTED

19) Which surface of the substructure does  LOAD-COMPONENT-1  act  NORMAL
    to?   (If  more  than one surface, you should consider the loading as
    two more component loadings)

** WIDTH-LENGTH

20) Enter the magnitude of the distributed load (in pounds):

** 1

21) Are there any other load components associated with LOADING-1?

** YES

  --------LOAD-COMPONENT-2--------

22) Would you describe LOAD-COMPONENT-2 as being DISTRIBUTED over most of
    the substructure or as acting at a POINT of the substructure?

** DISTRIBUTED

23) Which surface of the SUBSTRUCTURE does  LOAD-COMPONENT-2  act  NORMAL
    to?   (If  more  than one surface, you should consider the loading as
    two or more component loadings)

** WIDTH-LENGTH

24) Enter the magnitude of the distributed load (in pounds):

** .5

25) Are there any other load components associated with LOADING-1?
```

** NO

26) Are there any other loading conditions associated with wing tip (SUBSTRUCTURE-1)?

** YES

--------LOADING-2--------

27) What is the name of LOADING-2?

** LANDING

28) Enter the number of loading cycles to be applied:

** 300

The first load component associated with LOADING-2 will be called:

--------LOAD-COMPONENT-3--------

29) Would you describe LOAD-COMPONENT-3 as being DISTRIBUTED over most of the substructure or as acting at a POINT of the substructure?

** DISTRIBUTED

30) Which surface of the substructure does LOAD-COMPONENT-3 act normal to? (If more than one surface, you should consider the loading as two or more component loadings)

** WIDTH-LENGTH

31) Enter the magnitude of the distributed load (in pounds):

** 5

32) Are there any other load components associated with LOADING-2?

** NO

33) Are there any other loading conditions associated with wing tip (SUBSTRUCTURE-1)?

** NO

34) Are there any other substructures of airplane wing relevant to this analysis?

```
** YES

      --------SUBSTRUCTURE-2--------

35) What is the name of SUBSTRUCTURE-2?

** ROOT

36) What is the material composing most of root (SUBSTRUCTURE-2)?

** HIGH-STRENGTH-ALUMINIUM

37) Enter the average  longest  dimension  of  root  (SUBSTRUCTURE-2)  in
    inches (or <number> CMS):

** 54

38) Enter the  average  thickness  of  root  (SUBSTRUCTURE-2)  in  inches
    (<number> CMS):

** 8

39) Enter the weight of root (SUBSTRUCTURE-2)  in  pounds  (or  <numbers>
    KGS):

** 180

40) Would you describe the construction of root  (SUBSTRUCTURE-2)  to  be
    truss  and/or  frame  elements  (NETWORK),  sheets  and/or  blocks of
    material (CONTINUUM), or a combination of both?

** CONTINUUM NETWORK

41) What is the configuration of support for root (SUBSTRUCTURE-2)?

** ONE-SIDE

42) Can root (SUBSTRUCTURE-2)  be  modelled  as  a  one,  two,  or  three
    dimensional system?

** 3

43) Is the overall geometry of root (SUBSTRUCTURE-2) curved or planar?

** CURVED

The first loading applied to root (SUBSTRUCTURE-2) will be called:
   --------LOADING-3--------

44) What is the name of LOADING-3?

** LANDING

45) Enter the number of loading cycles to be applied:

** 300

The first load component associated with LOADING-3 will be called:

** 250000

    --------LOAD-COMPONENT-4--------

46) Would you describe LOAD-COMPONENT-4 as being DISTRIBUTED over most of
    the substructure or as acting at a POINT of the substructure?

** POINT

47) Which surface of the substructure does  LOAD-COMPONENT-4  act  NORMAL
    to?  (If  more  than one surface, you should consider the loading as
    two or more component loadings)

** WIDTH-LENGTH

48) Describe where on the substructure LOAD-COMPONENT-4 is applied:

** NEAR-FREE-EDGE

49) Enter the magnitude of the point load (in psi):

** 250000

50) Are there any other load components associated with LOADING-3?

** YES

    --------LOAD-COMPONENT-5--------

51) Would you describe LOAD-COMPONENT-5 as being DISTRIBUTED over most of
    the substructure or as acting at a POINT of the substructure?

** POINT

52) Which surface of the substructure does  LOAD-COMPONENT-5  act  NORMAL
    to?  (If  more  than one surface, you should consider the loading as
    two or more component loadings)
```

```
** WIDTH-LENGTH

53) Describe where on the substructure LOAD-COMPONENT-5 is applied:

** NEAR-FREE-EDGE

54) Enter the magnitude of the point load (in psi):

** 1000

55) Are there any other load components associated with LOADING-3?

** NO

56) Are there any other loading conditions associated with root
    (SUBSTRUCTURE-2)?

** NO

57) Are there any other substructures of airplane wing relevant to this
    analysis?

** NO

58) Do the supports of airplane wing involve Coulumb friction, nonlinear
    springs, and/or gapping?

** NO
```

```
GC:  lists
4021, 20405 FREE CELLS
```

The following analysis classes are relevant to the analysis of your structure:

1) General-inelastic

   Logic to scan deflections, calculate relative values, and compare with code limits should be called upon.

   Logic to scan stresses, smooth, and compare with allowable stresses (with appropriate safety factors) should be used.

   Activate incremental stress - incremental strain analysis.

   Model nonlinear stress-strain relation of the material.

Solution will be based on a mix of gradient and Newton methods.

End of Consultation.

DISCUSSION AND CONCLUSIONS

This examination of the use of heuristic programming to assist an engineer in selecting an appropriate analysis strategy leads to the following conclusions:

1. The heuristic approach includes all capabilities necessary for rationally selecting solution strategy. In particular, a preliminary analysis model can be imbedded in the rules, the analyst and consultant can address decision making substructure by substructure, and data on material and analysis characteristics can be accessed and manipulated as necessary.

2. The approach can offer valuable assistance to the structural analyst. With the implementation used, the consultant supplies expertise in a readily accessible and usable form. It interfaces with the analyst only on matters pertinent to his structure and analysis. By appropriate addition of logic it can be made to interface with the analyst over a spectrum of details pertinent to the structure, analysis procedure and model. The initial user of the program is prepared to put up with detailed questions in order to ensure that the model is correct. A proficient user will know the parts of the consultation that should be used. This method, therefore, resolves the problem in interactive computing of writing a program that can react to the knowledge level of the user.

3. The ability to query the data base and to obtain documentation on the use of the rules for consultation makes the system an efficient tool for programmed learning.

REFERENCES

[1] Feigenbaum, E. A., Buchanan, B. G. and Lederberg, J.: On Generality and Problem Solving: A Case Study Using the DENDRAL Program, Machine Intelligence, Edinburgh Univ. Press, 1971.

[2] Shortliffe, E.: Computer-based Medical Consultations: MYCIN; New York, Elsevier, 1976.

[3] MARC-CDC User Information Manual, Volumes I-IV; MARC Analysis Research Corporation, Palo Alto, California, 1978.

TABLE 1

STRESS FORMULA

	Point Load Site		Dist.Load	
Configuration	Support	Centroid	Free	Uniform
L 1 side W	$\frac{PDL}{8IeW}$	$\frac{PDL}{4IeW}$	$\frac{3PDL}{8\ W}$	$\frac{DpL}{4Ie}$
L 2 sides W opp.	$\frac{3PDL}{32IeW}$	$\frac{PDL}{8IeW}$	$\frac{DPL}{8IeW}$	$\frac{DPL}{16Ie}$

RELATIVE DEFLECTION

	Point Load Site		Dist.Load	
Configuration	Support	Centroid	Free	Uniform
L 1 side W	$\frac{PL}{48EIeW}$	$\frac{PL}{12EIeW}$	$\frac{3PL}{16EIeW}$	$\frac{pL}{24EIe}$
L 2 sides W opp.	$\frac{7PL}{192EIeW}$	$\frac{PL}{24EIeW}$	$\frac{7PL}{192EIeW}$	$\frac{5pL}{192EIe}$

*For plate, shell, and semi-monocoque structures; for multiple member networks.

"Plate" = Continuum, 2D, planar, width-length loading
"Shell" = Continuum, 2D or 3D, curved, any loading
"Semi" = Network and continuum, 2D or 3D, any loading
D = Section depth (in.)
E = Young's modulus (#/in^2)

Ie = Effective inertia $- \frac{D}{12}$ for solid section, $\frac{TD}{2}$ for thin walled section (T=wall thickness)

L = Longest distance between or from support lines
P = Point load (#)
p = Distributed load magnitude, (#/in^2)

W = $\frac{Wt}{LDe}$ for solid; = $\frac{Wt}{2LTp}$ for thin walled; and WT = weight (#)

TABLE 2 ANALYSIS STRATEGIES CONSIDERED

Nonlinear geometry crack growth
Nonlinear geometry stress margin
Nonlinear geometry fatigue
Buckling (extrapolation vs path)
Bifurcation
Nonlinear geometry strength
Nonlinear geometry deflections
Inelastic crack growth
Inelastic stress failure
Material Instability
Inelastic collapse
Inelastic fatigue
Inelastic strain accumulation failure
Elasto-plastic collapse (radial vs incremental)
Inelastic excessive deflection
Inelastic stiffness degradation
Inelastic strength
Inelastic deflection
Nonlinear crack growth
Nonlinear stress margin
Nonlinear material instability
Nonlinear yielding collapse
Nonlinear fatigue
Nonlinear strain accumulation
Nonlinear buckling
Nonlinear bifurcation
Nonlinear excessive deflection
Nonlinear stiffness degradation
Nonlinear strength
Nonlinear deflection
Nonlinear boundary condition
General large displacement analysis
General inelastic analysis
General nonlinear analysis

TABLE 3 PRINCIPAL CONSULTATION PARAMETERS

Structure

- o Name
- o Type of Nonlinearity
- o Integrity goal
- o Boundary nonlinearity
- o Analysis
- o Maximum deflection
- o Maximum stress

Substructure

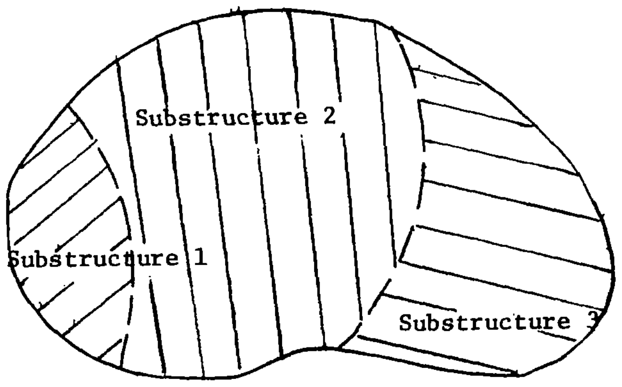
- o Name
- o Material
- o Geometry
- o Construction
- o Support conditions
- o Length
- o Skin thickness
- o Shape
- o Weight
- o Peak stress
- o Peak deflection
- o Stress criterion

Loading

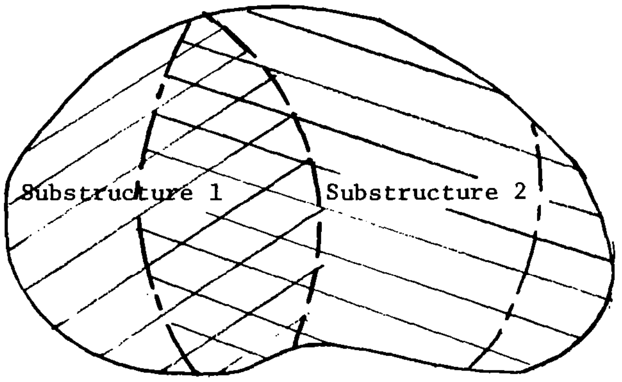
- o Name
- o Number of cycles
- o Stress bound
- o Deflection bound

Load Component

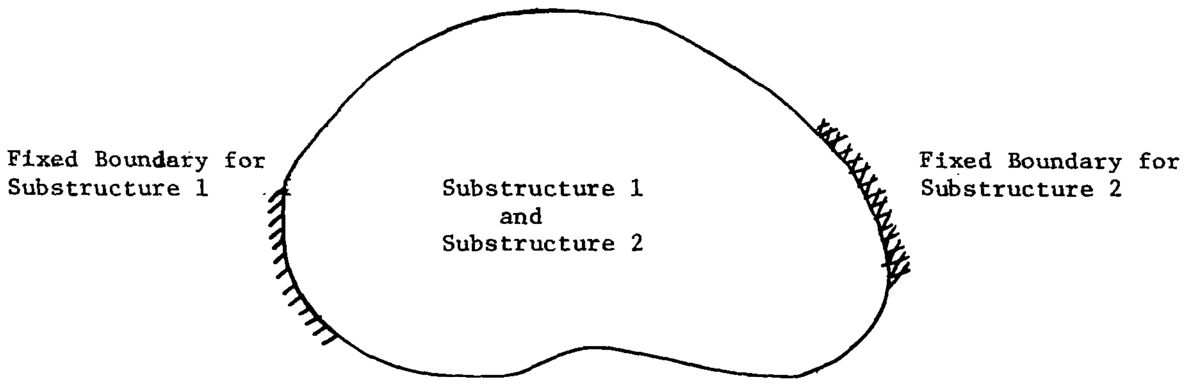
- o Name
- o Type
- o Direction
- o Magnitude
- o Stress
- o Deflection



(a) Conventional finite element substructures.



(b) Overlapping substructures.


Fixed Boundary for Substructure 1
Substructure 1 and Substructure 2
Fixed Boundary for Substructure 2

(c) Dual substructuring.

Figure 1.- Illustrations of substructuring.

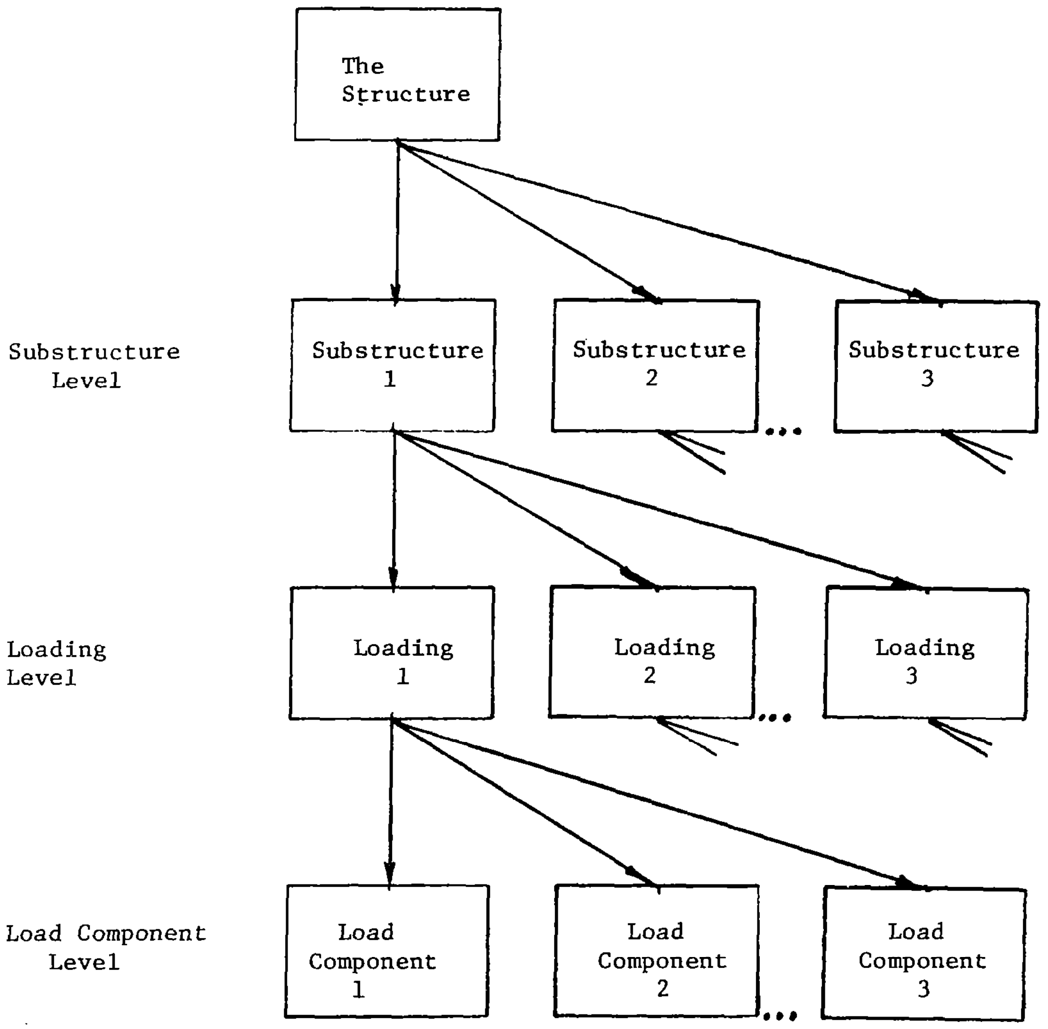


Figure 2.- Context tree.

A DIRECT ELEMENT RESEQUENCING PROCEDURE

J. E. Akin
University of Tennessee
Brunel University

R. E. Fulford
University of Tennessee

SUMMARY

Element by element frontal solution algorithms are utilized in many of the existing finite element codes. The overall computational efficiency of this type of procedure is directly related to the element data input sequence. Thus, it is important to have a pre-processor which will resequence these data so as to reduce the element wavefronts to be encountered in the solution algorithm. This paper reports on a direct element resequencing algorithm for reducing the element wavefronts. It also generates computational byproducts that can be utilized in the pre-front calculations and in various post-processors. Sample problems are presented and compared with other algorithms.

INTRODUCTION

Frontal solution procedures for finite element codes were presented independently at about the same time by Hellen [1], Irons [2], and Melosh [3]. These codes utilize an element by element assembly and factorization of the system equations. This procedure has been illustrated for simple models by Hellen [1], and Irons [4]. When utilizing this elimination process one is concerned with the maximum number of active columns or the maximum front associated with any element in the system. This quantity depends solely on the input order of the element incidences cards. By way of comparison, if one were using a frontal solution of the completely assembled system equations, one would be concerned with a wavefront defined by the nodal numbering system. This study describes an algorithm for resequencing the element order so as to reduce the element wavefronts.

THE ELEMENT RESEQUENCING STRATEGIES

Several bandwidth resequencing routines were published before the frontal solution methods became popular. Thus, in investigating frontal reduction methods one should also consider algorithms that were originally written for reducing the bandwidth of a system of equations. Many of these routines are effective in reducing both the nodal and element wavefronts. The Cuthill-McKee [5] algorithm has been shown to be effective in reducing both the system front and bandwidth [6]. The most efficient nodal resequencing algorithms written specifically for the system frontal method are probably those developed by Levy [7] and King [8]. The Cuthill-McKee algorithm is probably the most commonly used method for reducing system fronts.

The above references are concerned with frontal solutions that are dependent on the nodal numbering system. The present study is directed toward reducing wavefronts encountered in the element by element reduction procedure. Define the front associated with a particular element to be equal to the front of the previous element minus those degrees of freedom that made their last appearance in the previous element plus those degrees of freedom which make their first appearance in the particular element under consideration. This is a quantity that needs to be reduced to save storage and increase the computational efficiency of the equation solving algorithm. The value of the maximum element front is dependent on the input order of the element incidences list. The optimum input sequence is the one that results in the smallest front. Instead it is necessary to utilize a resequencing algorithm to obtain a reduced front in hopes that the reduced value is near the optimum value.

Akin and Pardue [9] have presented two procedures for reducing wavefronts by element resequencing strategies. Both of their procedures were based on generalizations of the Cuthill-McKee method. The disadvantages of these methods are that they require relatively large amounts of storage and initial calculations. The second method has an additional disadvantage in that it does not consider the number of nodes per element (element nodal degree) in its tie breaking options. The present paper introduces a new direct element resequencing strategy that has several advantages over the above methods. First, it requires a much smaller number of initial calculations and storage locations. In addition it has five levels of automatic tie breaking strategies in the resequencing algorithm.

Most resequencing programs are based on the concepts of the level and degree. The present study generalizes these concepts and utilizes the following definitions: Nodes (or elements) adjacent to a given node (or element) are said to be at the same level. The degree of a node (or element) is the number of nodes or elements to which it is connected. The term current degree will denote the degree based on the current number of unresequenced neighbors. For example, if a node has eight element neighbors, three of which have been renumbered, then its element degree is eight and its current element degree would be five.

Consider an effective front defined as, for element i,

$$W_i = W_{i-1} + F_i - L_{i-1} , \quad i = 1, \ldots, NE$$

where W_i is the front width, F_i the number of degrees of freedom first appearing, L_i the number of degrees of freedom making last appearance, NE the number of elements and where $W_0 \equiv L_0 \equiv 0$. As a check on this calculation we know $W_{NE+1} \equiv 0$. The present strategy is based on the concept of minimum element front growth. That is, select new element i such that the quantity $(F_i - L_{i-1})$ is minimum. Given a starting element as new element one the elements neighboring this element (i.e. at the same 'level') are numbered according to this strategy. Then all elements at the level of new element two are numbered. The numbering continues in this fashion, level by level, until all elements have been numbered. If a starting element is not given, the program could select the one with the smallest or largest F_i.

In such a procedure one often encounters a number of ties of candidates for the next few element numbers. The present code first selects the element (or elements) with the minimum number of new nodes (i.e. the minimum F_i). If this results in a tie then the element with the maximum number of existing nodes is selected (i.e. the maximum L_i). A second tie would be broken by choosing the element with the smallest number of un-numbered element neighbors. If a tie still exists, the element with the largest number of active nodes is selected. Finally, should a tie still exist, the last element in the list is utilized. Clearly, the rank of the tie breaking order could be changed.

EXAMPLE

Clearly one problem is to define F_i and L_i. This is accomplished by defining two scratch arrays, say LFIRST and NOADJL, equal in length to the number of nodes. When LFIRST(J) ≠ 0, it equals the new element number in which node J became active. It is initially zero. From this definition one notes that for new element i the value of F_i is equal to the number of nodes on that element for which LFIRST = 0. Once F_i is established the above group of nodes have their zero value of LFIRST changed to i, the new element number. Array NOADJL represents the current number of un-numbered elements adjacent to each node. Initially it equals the total number of elements adjacent to each point. Clearly when NOADJL(K) = 1 then node K is making its last appearance. The value of L_i for element i equals the number of nodes on the element for which NOADJL = 1. Once an element is renumbered the current value of NOADJL for each of its nodes is reduced by one. To illustrate the concepts consider the three-element model shown in Figure 1. The original element wavefronts are 3,4 and 3. The histories of arrays NOADJL and LFIRST are given Table 1. These are established in the following manner:

1. Set new $W_0 = L_0 = 0$, initialize NOADJL, zero LFIRST. Select the new first element L1. Say L1 = 2.

2. The nodes of L1 are 2,3 and 5. We observe:

NODE	ARRAY		COMMENT
2,	LFIRST = 0 ,	new node ;	set LFIRST = 1
	NOADJL = 3 ,	node remains ;	set NOADJL = 2
3,	LFIRST = 0 ,	new node ;	set LFIRST = 1
	NOADJL = 1 ,	node leaves ;	set NOADJL = 0
5,	LFIRST = 0 ,	new node ;	set LFIRST = 1
	NOADJL = 2 ,	node remains ;	set NOADJL = 1

Summary: There are three new nodes, the element front is 3, and one node is leaving, i.e.:

$$F_1 = 3, L_1 = 1 \ , \quad W_1 = W_0 + F_1 - L_0 = 0 + 3 - 0 = 3 \ .$$

3. The element neighbors of L1 = 2 are elements 1 and 3. Select L2 from that list.

A. Consider candidate element 1: Its nodes are 1, 2, 6.

NODE		COMMENT
1,	LFIRST = 0 ,	new node
	NOADJL = 1 ,	node leaves
2,	LFIRST = 1 ,	active since loop 1
	NOADJL = 2 ,	node remains
6,	LFIRST = 0 ,	new node
	NOADJL = 2 ,	node remains

Summary: 2 new nodes, 1 node leaving, element active since loop 1, 1 old node (3 - 2 = 1)

B. Consider candidate element 3: Its nodes are 2, 5, 6.

NODE	COMMENT
2,	active since loop 1, remains
5,	active since loop 1, leaves
6,	new node, remains

Summary: 1 new node, 1 node leaving, element active since loop 1, 2 old nodes (3 - 1 = 2)

Select L2 = 3. Then $F_2 = 1$, $L_2 = 1$, $W_2 = W_1 + F_2 - L_1 = 3 + 1 - 1 = 3$

For nodes 2, 5, 6, set NOADJL = NOADJL - 1, and if LFIRST = 0, set LFIRST = 2.

4. Are there any un-numbered elements adjacent to L1? Yes, old element 1. Consider element 1: Its nodes are 1, 2, 6.

NODE	COMMENT
1,	is new and leaves
2,	is active since loop 1 and leaves
6,	is active since loop 2 and leaves

Summary; 1 new node, 3 nodes leaving, element active since loop 1, 2 old nodes

Set L3 = 1, $F_3 = 1$, $L_3 = 3$, $W_3 = W_2 + F_3 - L_2 = 3 + 1 - 1 = 3$.

For nodes 1, 2, 5, set LFIRST = 3 if LFIRST = 0, and set NOADJL = NOADJL - 1.

5. No elements remain.

Check calculations: $W_4 = W_3 + F_4 - L_3 = 3 + 0 - 3 = 0$, check!

New element wavefronts are 3, 3, and 3. The new data are shown in Figure 2.

APPLICATIONS

Cuthill [2] has applied various resequencing algorithms to the labelled tree structure shown in Fig. 3. The present algorithm was also applied to this structure and the results are compared with those of Cuthill in Table 2.

The second example test was a simple structure considered by Akhras and Dhatt [10]. It consists of three concentric circles divided, from the center, into eight equal angular segments. Thus it contains eight triangular and sixteen quadrilateral elements. These quadratic elements and their original order are shown in [10]. The third example was a quarter symmetry mesh of a rectangle with a center circular hole. The original element data were generated in a random order so as to cause a large initial wavefront. The fourth problem was a half symmetry shell model. It involved a cylinder with hemispherical caps supported horizontally on two vertical plate saddles. The fifth and sixth problems involved complicated three-dimensional surfaces with branches.

The wavefront reduction data for these problems are given in Table 3. These results show the algorithm to be efficient in reducing the maximum element wavefront. It requires significantly less storage than the algorithm used by Akin and Pardue [9]. Other applications to practical engineering problems have shown reductions of at least thirty percent.

Table 4 shows some relative computation costs. The first item is a measure of the cost of building the neighbors lists that the resequencing subroutine uses as a data base. These calculations are clearly the most expensive. They can be done in various ways and it appears that more efficient procedures can be developed. Much of these data could be utilized in the pre-front stage of the solution algorithm. The second item shows the actual resequencing costs. These are quite small and indicate that one should try several different starting elements since most of the cost goes into the first choice.

Complete program listings and instructions are included in Reference [10].

REFERENCES

1. Hellen, T. K.: A Front Solution for Finite Element Techniques, Central Electricity Generating Board, R & D Dept. RD/B/N 1459, 1969.

2. Irons, B. M.: A Frontal Solution Program for Finite Element Analysis. Intern. J. Num. Meth. Engr. 2, 5-32, 1970.

3. Melosh, R. J. and Bamford, R. M.: Efficient Solution of Load Deflection Equations. J. Structural Div. ASCE, 95, ST4, 661-676, 1969.

4. Irons, B. M. and Kan, K. Y.: Equations Solving Algorithms for the Finite Element Mehtod, Numerical and Computer Methods in Structural Mechanics, S. J. Fenves, et al. (eds.). Academic Press, 497-512, 1973.

5. Cuthill, E. and McKee, J.: Reducing the Bandwidth of Sparse Symmetric Matrices. Proc. ACM Nat. Conf., 157-172, 1969.

6. Cuthill, E.: Several Strategies for Reducing the Bandwidth of Matrices, Sparse Matrices and Their Applications, D. J. Rose and R. D. Willoughby (eds.). Plenum Publishing Co., New York, 157-166, 1972.

7. Levy, R.: Resequencing of the Structural Stiffness Matrix to Improve Computational Efficiency. J. P. L. Quarterly Tech. Review, 1, 2, 61-70, 1971.

8. King, I. P.: An Automatic Reordering Scheme for Simultaneous Equations Derived from Network Analysis. Intern. J. Num. Meth. Engr. 2, 523-533, 1970.

9. Akin, J. E. and Pardue, R. M.: Element Resequencing Algorithm for Frontal Solutions, The Mathematics of Finite Elements and Applications, vol. 2, J. R. Whiteman (ed.). Academic Press, London, 1976.

10. Akhras, G. and Dhatt, G.: An Automatic Node Relabelling Scheme for Minimizing a Matrix Bandwidth. Intern. J. Num. Meth. Engr. 10, 787-797, 1976.

11. Fulford, R. E.: A Wavefront and Bandwidth Reduction Algorithm. M. S. Thesis Dept. Eng. Sci. & Mech., Univ. of Tenn., March, 1977.

Table 1

Logic Arrays for Example

J \ LOOP	NOADJL				LFIRST			
	0	1	2	3	0	1	2	3
1	1	1	1	0	0	0	0	3
2	3	2	1	0	0	1	1	1
3	1	0	0	0	0	1	1	1
4	0	0	0	0	0	0	0	3
5	2	1	0	0	0	1	1	1
6	2	2	1	0	0	0	2	2

Table 2

Results for Labelled Tree

Algorithm	Wavefront	Profile
Original	7	107
Cuthill-McKee	9	101
Reverse CMK	4	54
King	5	59
Reverse K	3	38
Levy	2	37
Present	3	42

Table 3

Element Wavefront Reduction Achieved
by the Present Algorithm

Example Number	Number of Nodes	Number of Elements	Types*	Wavefront		
				Original**	Resequenced	Reduction,%
1	24	23	L-2	7	4	43
2	73	24	Q-8,T-6	23	22	4
3	272	121	T-6	123	34	80
4	630	639	Q-8,T-3	224	35	84
5	623	760	Q-4,L-2	230	82	65
6	253	281	Q-4,T-3	52	36	31

* L = Line Element, T = Triangle, Q = Quadrilateral

** Assuming one degree of freedom per node

Table 4

Algorithm Steps as Percent of Total Run Time

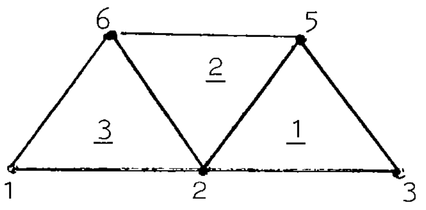
Example	Total CPU Time *	Generate Neighbors List	Resequence Elements **
2	4.35	8.3%	10 %
3	17.47	67.5%	12.5%
4	42.70	80 %	7.5%
5	38.54	75 %	7 %
6	17.55	52 %	15 %

* Seconds on IBM 360/65

** From four different starting elements

Element (L)	Incidences (NODES)			Adjoining Elements (LADJL)	
1	1	2	6	2	3
2	2	3	5	1	3
3	2	5	6	1	2

Figure 1. Iron's Element Front Example



Element Number			
old	2	3	1
new	1	2	3

Figure 2. New Element Model

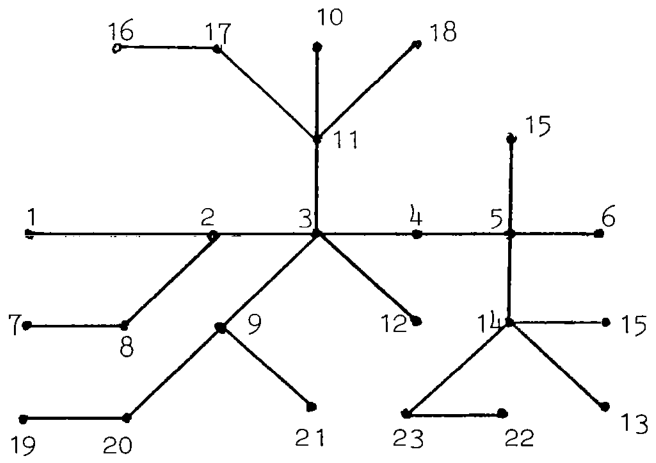


Figure 3. A Labelled Tree [2]

PROGRESSIVE FAILURE OF STRUCTURES*

Khalilollah Khozeimeh, Theodore G. Toridis
and Noor Hussain
The George Washington University

Shahram E. Zanganeh
Howard University

SUMMARY

A procedure is presented for determining the nonlinear behavior of structures subjected to extreme loading and the possibility of development of potential for progressive failure. The methodology takes into account the effect of both material and geometric nonlinearities. At a given stage of analysis, the individual components of the structure are checked against predetermined failure criteria. Subsequently, the failing components are removed and the modified structure is analyzed for overall failure. Examples, obtained from a computer program based on the proposed procedure, showing the applicability of the method are presented.

SYMBOLS

Values are given in both SI and U.S. Customary Units. The calculations were made in U.S. Customary Units.

$\{F\},\{F^\circ\}$	nodal applied and equivalent force vectors, respectively
$[K],[K_G],[K_T]$	elastic, geometric, and total system stiffness matrices, respectively
k_2,k_1	slope of inelastic and elastic branch of stress-strain curve, respectively
$[m]$	consistent mass matrix
P_i	_i_th component of element nodal forces
$\bar{P}_1,\bar{P}_2,\ldots$	normalized stress resultants used in the yield criterion
$\{q\},\{\dot{q}\},\{\ddot{q}\}$	generalized displacement, velocity, and acceleration vectors, respectively

*This research was sponsored by the National Science Foundation through grant NSF-ENG76-00332.

Δt	time increment
Φ	plastic potential function
Subscripts:	
est	estimate
pre	previous solution step

INTRODUCTION

The determination of the response of structural systems under externally applied loading, whether of static or dynamic nature, has been always of great concern to the structural engineers, especially when such response has extended into the nonlinear range. However, up to recent years the solution to only few simple problems had been obtained. This is due to the complicated nature of the problem which renders the classical methods of solution inapplicable.

With the advent of high speed computers in the past few decades, a more realistic solution of complex engineering problems has become an attainable goal. Consequently, numerous investigators have turned their attention to the solution methods for nonlinear structural problems. Some of the work done in this respect with regards to beam and frame type structures as well as plate structures can be found in references 1-8. In addition, numerous studies have been reported on the application of the finite element to nonlinear problems. Some of these studies deal primarily with the material nonlinearity while others outline methods for treating general problems. Notable among the first group are the works of Akyuz and Merwin (ref. 9), Argyris (ref. 10), Marcal (ref. 11) and Armen et al.(ref. 12). Among the works concerned with the latter category are the findings of Oden (ref. 13), Stricklin et al.(ref. 14), Marcal and collaborators (ref. 15), Bathe et al.(ref. 16), and Zienkiewicz et al. (ref. 17).

The nonlinear behavior of particular types of elements has also received the attention of the investigators in the field. Some of the works dealing with beam and frame type elements have been cited above. Other studies dealing with beams as well as plate type elements are the works of Toridis and Khozeimeh (ref. 18,19) Akkoush et al.(ref. 20), Marcal et al.(ref. 21), and McNeice (ref. 22).

In recent years some investigators have turned their attention to the question of structural damage and failure as the result of excessive loads and/or ensuing deformations which are well beyond the linear range or the acceptable design levels. In particular the effect of loss of certain supporting elements on the overall behavior of the structure has received due attention. These investigations have been motivated by the observations on the performance of actual structures in which such loss of elements has caused "progressive failure," a chain reaction type behavior, resulting in the collapse of the entire structural system. Recent examples of this type of behavior are the Ronan Point Modular Building collapse in England and the Skyline Towers High-Rise Building collapse in northern Virginia, U.S.A.

The study of existense of potentials for this type of failure is becoming more and more important as the concept of modular and panelized buildings gains in popularity. In this type of structures extensive use is made of premanufactured shear wall and floor panels that are interconnected to act as the basic load carrying systems, providing the three-dimensional rigidity of the building. The successful performance of such buildings depends on the behavior of the basic panels (elements) and the connecting system between the panels. It is, therefore, highly desirable to determine the performance of such structures under extreme loading and environmental conditions, in order to eliminate unsafe design practices. Since also the failure of one or more of the structural components or subassemblies gives rise to potential for progressive collapse and the ensuing disproportionate deformations, in studies dealing with such systems it is desirable to consider the ultimate strength properties of the structure prior, as well as after, the failure of some of its components.

In the present study, a procedure for determining the behavior and the potential for progressive collapse of the structural system subjected to extreme loading is formulated. The structure is modelled as an assemblage of beam and plate type elements and the response is found based on an incremental approach which allows for consideration of both material and geometric nonlinearity. At each stage of the structural deformation, failure criteria pertaining to excessive deformations, strength and stability of the structure are checked and parts of the structure that meet the appropriate failure criteria are removed and the remainder of the structure is checked against overall failure. The entire procedure is incorporated in a computer program.

GENERAL APPROACH

The response of the structural systems when subjected to high intensity loading generally extends into the nonlinear range. Consequently, in the analysis of such systems the effect of both material and geometric nonlinearities must be considered. Of special interest in the analysis is the incidence of abnormal loadings, i.e., loadings against which adequate measures have not been incorporated in the design. Such loadings, although infrequent, may lead to localized structural damage, which in turn may cause a "progressive" chain reaction type failure culminating in structural damages entirely disproportionate to the significance of the initiating cause. Thus to determine the complete nonlinear response of a structural system, its ability to form an alternative path to bridge any local damage must be studied.

The general approach adopted in this study to achieve the above objective is an extension of the work reported in references 18 and 19 and is based on the finite element method coupled with an incremental approach. To this end the structural system is modelled as a collection of beam and rectangular plate type elements. The three-dimensional beam element is used to model the skeletal frames while the plate element which is capable of simulating bending and/or in-plane action can be used effectively in representation of shear walls and floor panels, elements of construction which seem to become more important as trend towards modular, "panelized" construction continues. The detailed properties of the above beam and plate have been reported in references 4 and 18 and will not be repeated here. As shown in reference 23 the basic dynamical equations governing the behavior of a structural system can be obtained through the application of the Hamilton's Principle as applicable to discrete systems. In matrix form, these equations are expressed as

$$[m]\{\ddot{q}\} + ([K] + [K_G])\{q\} = \{F\} + \{F^O\} \tag{1}$$

where

$[m]$	=	generalized consistent mass matrix of the structural system
$\{q\}$ and $\{\ddot{q}\}$	=	generalized displacement and acceleration vectors, respectively
$[K]$	=	generalized elastic stiffness matrix of the structural system
$[K_G]$	=	generalized geometric stiffness matrix of the structure
$\{F\}$	=	generalized nodal force vector corresponding to the externally applied loads
$\{F^O\}$	=	equivalent generalized nodal force vector due to plastic strains, computed in accordance with the "initial stress or strain" method

In case of static loading, the above equation can be reduced to

$$([K] + [K_G])\{q\} = \{F\} + \{F^O\} \tag{2}$$

Based on equation (1) or equation (2), whichever governs the problem, the following incremental procedure to determine the behavior of the structure in the linear and nonlinear range is formulated.

Referring to equation (1) and based on the current configuration and the state of stress of the structural system, the components of the vector of the generalized accelerations, $\{\ddot{q}\}$, are found using the currently applied loads. Subsequently, the generalized displacement vector, $\{q\}$, is determined through a numerical integration procedure. The integration procedure utilized in this study is the Newmark's constant acceleration scheme (i.e., β=0) and can be expressed as

$$\{\dot{q}\}_2 = \{\dot{q}\}_1 + 0.5(\Delta t)[\{\ddot{q}\}_2 + \{\ddot{q}\}_1] \tag{3}$$

$$\{q\}_2 = \{q\}_1 + \Delta t\{\dot{q}\}_1 + 0.5(\Delta t)^2\{\ddot{q}\}_1 \tag{4}$$

In the above equations, Δt is the time step, the subscripts refer to the time stations, and $\{\dot{q}\}$ is the generalized velocity vector.

Having determined the vector $\{q\}$, the totals and the increments of the element nodal displacements and forces are obtained through the appropriate transformations and the current element stiffness matrices. The element nodal forces thus calculated are then used as an estimate to check for inelastic behavior in the element. To this end, the Mises yield criterion expressed in terms of the stress resultants is utilized. The normalized form of this criterion for a beam element is expressible as (ref. 19)

$$\Phi = [\bar{P}_1^2 + \bar{P}_2^2 + \bar{P}_3^2 + \bar{P}_4^2] = Y \tag{5}$$

where Y denotes the "yield value" which may change through straining, and $\bar{P}_1$, $\bar{P}_2$, $\bar{P}_3$, and $\bar{P}_4$ are the normalized form of the axial force, torsional moment, and bending moments about member y and z axes, respectively. A similar expression can be written in the case of a plate element.

If any element is undergoing inelastic deformation, its corrected force components and the corresponding contributions to the vector $\{F^o\}$ must be determined. This is done through a simplified approximate procedure known as "Proportioning Method" (ref. 19). In this approach, if Φ_{est} denotes the plastic potential function assuming elastic behavior, then an estimate of the increment of the plastic potential function due to current load/time increment, $d\Phi_{est}$, is found as

$$d\Phi_{est} = \Phi_{est} - \Phi_{pre} \tag{6}$$

where Φ_{pre} refers to the plastic potential function at the end of previous step. Then assuming a bilinear stress-strain relation and utilizing the "universal" stress-strain curve, the corrected plastic potential function, Φ, is determined as

$$\Phi = \Phi_{pre} + \frac{k_2}{k_1} d\Phi_{est} \tag{7}$$

in which k_1 and k_2 represent the slopes of the elastic and inelastic branches of the stress-strain curve, respectively. The basis of this simple procedure is explained in detail in reference 19.

Having found the final value of the plastic potential function, the _i_th component of the nodal forces, P_i, is determined by a proportioning process, i.e.,

$$P_i = \frac{P_{i_{est}}}{\Phi_{est}} \Phi \tag{8}$$

where $P_{i_{est}}$ is the estimate of the force component assuming elastic behavior. Furthermore, in the analysis the "Average Force Model" (ref. 19) is utilized. In the development of this model it is assumed that the entire element undergoes plastic deformations if the plastic potential determined from average value of stress resultants acting at the nodes exceeds the current yield value.

As is well known, the geometric nonlinearities can be attributed to two causes, namely the effect of large rotations and the contributions due to nonlinear strain-displacements (P-Δ) effects. The latter effects are accounted for in the present analysis through the inclusion of the geometric stiffness matrix, $[K_G]$. The entries to this matrix are directly affected by changes in the axial or in-plane force components acting on the elements of the structure. Consequently, this matrix is continuously updated in the solution process to reflect changes in the internal forces of the structure. To account for the effect of large rotations with its inherent change in geometry the total stiffness matrix, $[K_T]$, defined as the sum of elastic and geometric stiffness matrices, i.e.,

$$[K_T] = [K] + [K_G] \tag{9}$$

as well as the mass matrix and the equivalent force vector are regenerated through the use of the current transformation relations based on the deformed configuration of the structure.

In addition, before any new solution step is attempted, the modified configuration of the structure is determined. This is accomplished by checking the individual elements for excessive inelastic deformation and attainment of its ultimate strength which necessitates the removal of such elements from further consideration in the analysis. Also, the entire structure is checked for stability and functionality and all portions of the structural system that fail to meet the above requirements are also removed from consideration. Furthermore, if any modification is made to the structure, the appropriate system matrices are reformulated based on the latest configuration of the structure.

The entire procedure outlined above is incorporated in a general purpose computer program. The macro flow chart depicting the sequence of the operations is presented in Figure 1. As seen in the figure and based on the foregoing discussion, the procedure allows for removal of elements and nodes from further consideration. In case of an element, this is done by placing its number on the list of inactive elements and thus neglecting its contribution to system matrices in the subsequent solution steps. The removal of a given node entails two operations. First, all the elements incident to the node must be removed through the aforementioned procedure. Second, all the degrees of freedoms associated with the node must be eliminated to prevent the structural stiffness matrix from becoming singular. This is done in this study by introduction of artificial constraints at the node so that no degrees of freedom are assigned to the node in the subsequent analysis cycles. It should also be noted that in the case of static loading the analysis cycle refers to an increment of load rather than time.

FAILURE CRITERIA

A structural system or portions of it are said to have failed if certain prescribed conditions are violated. These may be based on strength requirements of individual parts of the structure or due to excessive displacements. Obviously, one of these possible modes of failure is instability. In general, instability is induced in the structural system composed of various members if the state of stress and deformation is such as to cause the system to lose its stiffness. This can come about if the axial or in-plane forces reach a critical value (buckling mode). Alternatively, the failure of a segment of the structure, perhaps through excessive deformation and formation of plastic hinges, may cause other portions or subassemblies of the structural system to undergo rigid body motion. However, irrespective of which mode of instability is encountered, the problem of stability can be formulated as the eigenproblem given by

$$([K] - \lambda[K_G]) = 0 \tag{10}$$

However, as has been pointed out by Gallagher (ref. 24), in the case of rigid body motion only, the total stiffness matrix will have eigenvalues of zero magnitude, and the corresponding eigenvectors will represent the rigid body modes. This fact is used to advantage in the present study to check for the potential of occurrence of rigid body motion whenever the determinant of the regenerated total stiffness matrix, $[K_T]$, approaches zero. If such rigid body motion occurs, the parts of the structure undergoing such motion are removed from consideration for the remaining time/load increments of the study.

Deformations are also of great importance as a criterion for determination of acceptable structural behavior. Traditionally, the formation of sufficient number of plastic hinges has been used as a measure of structural failure. However, since displacements and plastic deformations generally become extremely large before a structure becomes a true mechanism, failure criteria based on a count of plastic hinges are unsatisfactory. On the other hand, it is known that the distortions (displacements and rotations) in the structure increase a great deal just before the collapse load is reached. Therefore, a failure criterion based on the magnitude of structural distortions is more appropriate, especially if account is taken of the ability of the structure to strain-harden. This is indeed the approach adopted in this study. Unfortunately, very limited quantitive information is available in the literature on this subject; rather the investigators in the field have stressed the need for the experimental determination of such limits. In the present study, this problem is circumvented by requiring the input of the above information for a given structure based on the best available data and professional judgement.

The effect of inelastic deformations in a member or part of the structure can also be taken into account through the concept of the ductility factor or ductility ratio. Different definitions for the ductility ratio with respect to curvilinear and bilinear hysteresis curves have been reported in the literature (ref. 25-28). In the present study, the ductility factor is defined as the ratio of the maximum permissible or useful strain (or generalized strain/displacement) to the corresponding value at first yield. This factor is then used as a measure of failure in a structural component.

NUMERICAL RESULTS

To demonstrate the applicability of the proposed method, solutions to several structures have been obtained. Some typical results are reported herein. It should be mentioned that although the precedure is applicable to both beam and plate type structures, currently, only the beam elements have been fully incorporated in the computer program.

Example 1

The first example considered is a two story, two bay skeletal frame with dimensions as shown in Figure 2. All the girders are W 10x11.5 steel sections while the columns, with exception of the lower level interior column, are made up of W 8x20 sections. The lower level interior column is M 7x5.5 and all the steel is assumed to have a bilinear stress-strain relation with a yield point of 249 MN/m^2(36 ksi) with the slope of the inelastic branch being 0.01 times the corresponding value for the elastic branch. The columns are modelled by 3 equal elements per story and the girders are subdivided into 4 equal elements per bay. The loading consists of a uniform dead load, W1, distributed over the girders and a live load, W2, as shown in Figure 2. In the analysis, the distributed loads are replaced with equivalent nodal forces. The dead load is applied in 3 increments of 7.78 kN/m(0.53 kip/ft) each. This is then followed by application of live load increments of the same magnitude until the structure fails completely. The failure limits are set at 15.25 cm(6 in) and 0.2 radians for nodal displacement and rotation, respectively.

Figures 3A-F depict the sequence of structural modification due to propagation of failure. In Figure 3A the lower story inner column fails due to its ultimate strength being exceeded. Upon further loading, the girders start to fail due to strength requirements (Fig. 3B,C). Continuation of loading the structure leads to excessive displacements which necessitate the removal of a node (Fig. 3D) which in turn leads to rigid body motion and removal of further portions of the structure (Fig. 3E) and ultimate failure (Fig. 3F).

Example 2

The second example considered demonstrates the effect of a weak exterior column coupled with lateral loads. In this example the same frame as in the previous case is used except that the lower story right-hand side is considered to be a weak column (i.e., M 7x5.5 section) instead of the middle column. In addition, concentrated loads as shown in Figure 4A are applied to the structure. The loading sequence consists of 3 load increments of W1 = 7.78 kN/m(0.53 kip/ft) followed by 11 live load increments, W2, of the same magnitude. This is then followed by 6 load increments of H = 1.34 kN(0.3 kip). The failure pattern of this structure is depicted in Figures 4B to 4E. As can be observed, in this case the failure is not as extensive as in the previous example.

REFERENCES

1. Argyris, J.H., "Continua and Discontinua," Proceeding of Conference on Matrix Methods in Structural Mechanics, Wright Patterson Air Force Base, Ohio, October 1965.

2. Bergen, Pal G., "Non-Linear Analysis of Plates Considering Geometric and Material Effects," Report No. UCSESM 71-7, University of California, Berkeley, April 1971.

3. Burgmann, J.B., and Rawlings, B., "Dynamic Plastic Analysis of Pin-Jointed Frames," International Journal of Mechanical Science, Vol. 10, No. 12, December 1968, pp. 967-80.

4. Huang, H.K., "Large Deformations of Beam-Plate Assemblages," Doctor of Science Dissertation, The George Washington University, February 1972.

5. Lin, T.H., Lin, S.R., and Mazelsky, B., "Elastoplastic Bending of Rectangular Plates with Large Deflection," Journal of Applied Mechanics, Vol. 39, December 1972, pp. 978-982.

6. Nigam, N.C., "Yielding in Framed Structures Under Dynamic Loads," Journal of the Engineering Mechanics Division, ASCE, Vol. 96, No. EM5, October 1970, pp. 687-710.

7. Smith, J.H., "Nonlinear Beam and Plate Elements," Journal of the Structural Division, ASCE, Vol. 98, No. ST3, March 1972, pp. 553-568.

8. Toridis, T.G., "Dynamic Analysis of Frame and Plate Structures with Geometric and Material Nonlinearities," Report 3988, NSRDC, Bethesda, May 1973.

9. Akyuz, F.A., and Merwin, J.E., "Solution of Nonlinear Problems of Elasto-plasticity by Finite Element Method," AIAA Journal, Vol. 6, No. 10, October 1968, pp. 1825-31.

10. Argyris, J.H., Scharpf, D.W., and Spooner, J.B., "The Elasto-Plastic Calculation of General Structures and Continua," Proceedings of Third Conference on Dimensioning, Budapest, 1968, Akademiai Kiado, pp. 345-384.

11. Marcal, P.V., "Finite Element Analysis with Material Nonlinearity-Theory and Practice," Recent Advances in Matrix Methods of Structural Analysis and Design, Proceedings of U.S. - Japan Seminar on Matrix Methods in Structural Analysis and Design, Tokyo, pp. 257-282.

12. Armen, Jr. H., et al, "Finite Element Analysis of Structures in the Plastic Range," NASA CR-1649, February 1971.

13. Oden, J.T., "Finite Element Applications in Nonlinear Structural Analysis," Proceedings of the Symposium on Application of Finite Element Methods in Civil Engineering, Vanderbilt University, November 13-14, 1969, pp. 419-456.

14. Stricklin, J.S., Haisler, W.E., and Von Riesemann, W.A., "Evaluation of Solution Procedures for Material and/or Geometrically Nonlinear Structural Analysis," AIAA Journal, Vol. 11, No. 3, March 1973, pp. 292-9.

15. Hibbitt, H.D., Marcal, P.V., and Rice, J.R., "A Finite Element Formulation for Problems of Large Strain and Large Displacements," International Journal of Solids and Structures, Vol. 6, 1970, pp. 1069-86.

16. Bathe, K., Ramm, E., and Wilson, E.L., "Finite Element Formulations for Large Deformation Dynamic Analysis," International Journal for Numerical Methods in Engineering, Vol. 9, 1975, pp. 353-386.

17. Zienkiewicz, O.C., and Nayak, G.C., "A General Approach to Problems of Large Deformations and Plasticity Using Iso-Parametric Elements," Proceedings of Third Conference on Matrix Methods in Structural Mechanics, Wright-Patterson Air Force Base, Ohio, 1971.

18. Toridis, T.G., and Khozeimeh, K., "Inelastic Response of Frames to Dynamic Loads," Journal of the Engineering Mechanics Division, ASCE, Vol. 97, No. EM3, June 1971, pp. 847-864.

19. Khozeimeh, K. and Toridis, T.G., "Models for Inelastic Response of Beam-Plate Assemblages," Journal of the Engineering Mechanics Division, ASCE, Vol. 104, No. EM5, October 1978.

20. Akkoush, E.A., Toridis, T.G. and Khozeimeh, K., "Bifurcation, Pre- and Post-Buckling Analysis of Frame Structures," Computers and Structures, Vol. 8, No. 6, pp. 667-678.

21. Marcal, P.V., and McNamara, J.F., "Incremental Stiffness Method for Finite Element Analysis of the Nonlinear Dynamic Problems," Paper presented at International Symposium on Numerical and Computer Methods in Structural Mechanics, Urbana, Illinois, September 1971.

22. McNeice, G.M., "An Elastic-Plastic Finite Element Analysis for Plates with Edge Beams," Proceedings of the Symposium on Application of Finite Element Methods in Civil Engineering, Vanderbilt University, November 13-14, 1969, pp. 529-566.

23. Khozeimeh, K., "Inelastic Response of Beam-Plate Assemblages Subjected to Static and Dynamic Loads," Doctor of Science Dissertation, The George Washington University, May 1974.

24. Gallagher, R., "Finite Element Analysis," Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1975.

25. Bresler, B., "Behavior of Structural Elements - A Review," Building Practices for Disaster Mitigation, Building Science Series 46, NBS, U.S. Department of Commerce, February 1973, pp. 286-351.

26. Giberson, M.F., "Nonlinear Beams with Definition of Ductility," Journal of Structural Division, ASCE, Vol. 95, No. ST2, February 1969, pp. 137-158.

27. Newmark, N.M., and Hall, W.J., "Procedure and Criteria for Earthquake Resistant Design," Building Practices for Disaster Mitigation, Building Science Series 46, NBS, U.S. Department of Commerce, February 1973, pp. 209-236.

28. Santhakumar, A.R., "Ductile Behavior of Coupled Shear Walls Subjected to Reversed Cyclic Loading," Proceedings of International Symposium on Earthquake Structural Engineering, August 19-21, 1976, University of Missouri, Rolla, Vol. 1, pp. 501-507.

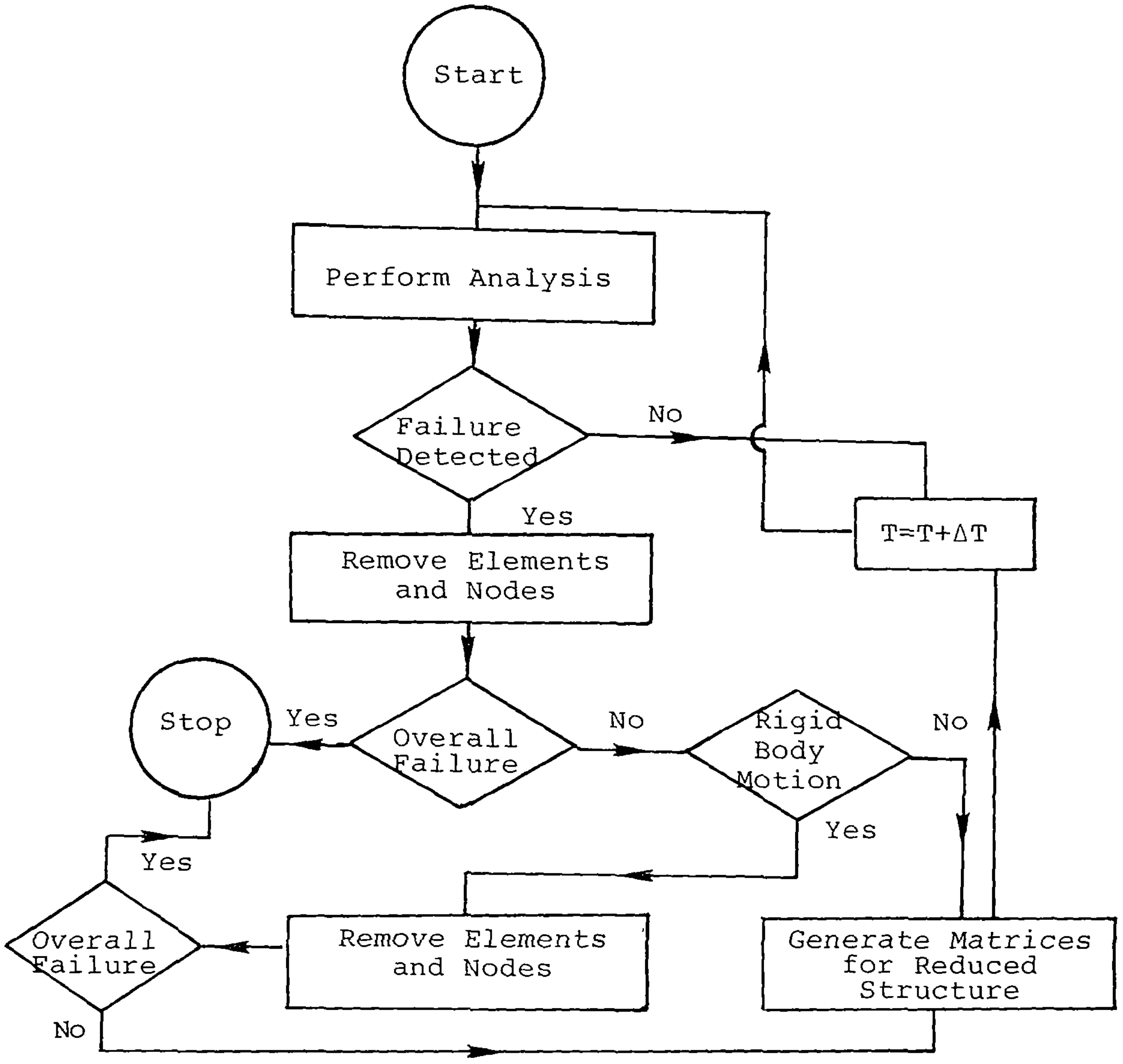


Figure 1.- Macro flow chart for progressive failure.

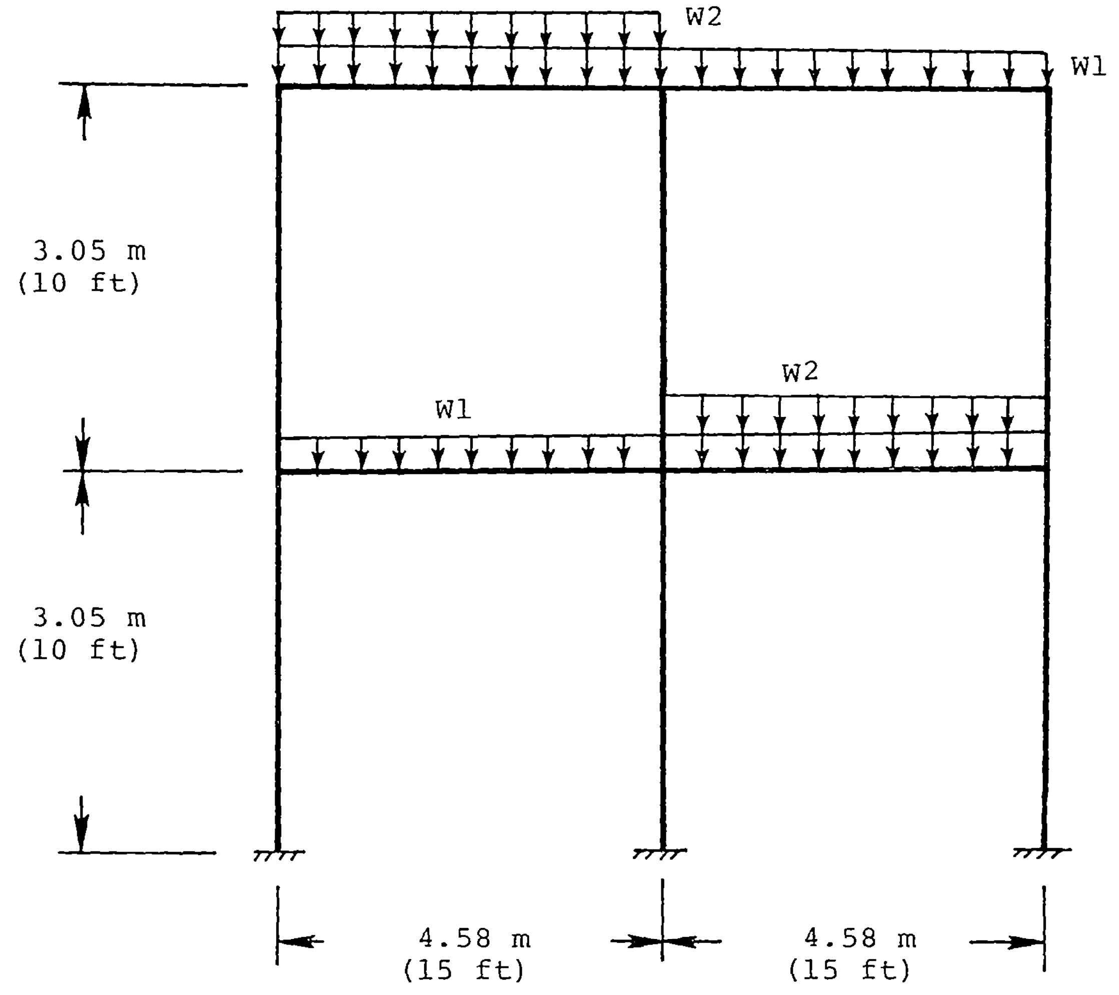
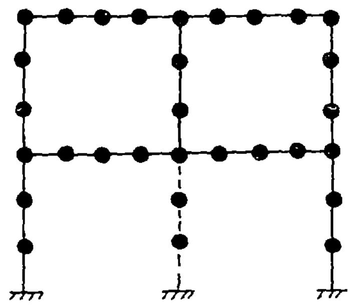
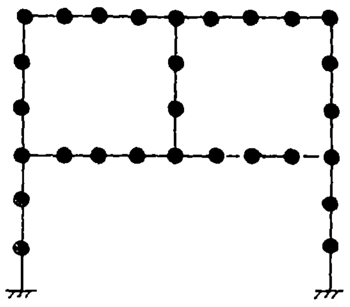
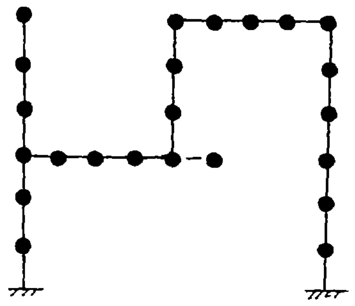
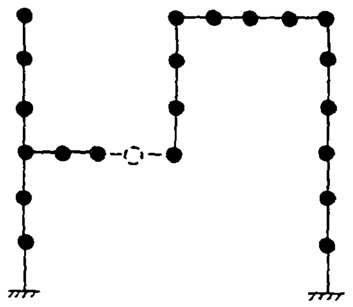


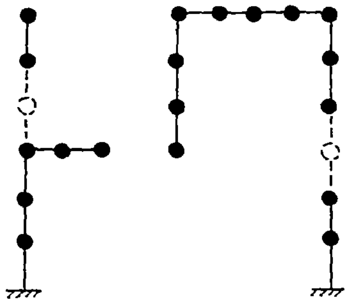
Figure 2.- Example structure.


(A) W1 = 23.34 kN/m; W2 = 46.68 kN/m.


(B) W1 = 23.34 kN/m; W2 = 54.46 kN/m.


(C) W1 = 23.34 kN/m; W2 = 85.58 kN/m.


(D) W1 = 23.34 kN/m; W2 = 101.1 kN/m.


(E) W1 = 23.34 kN/m; W2 = 108.9 kN/m.

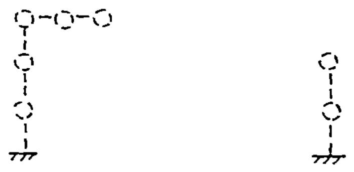
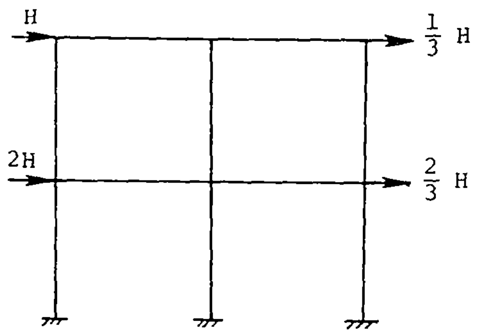
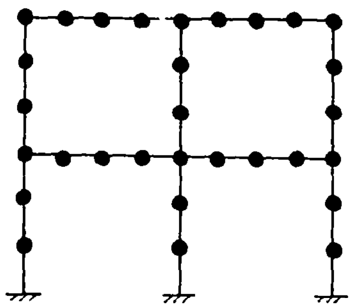

(F) W1 = 23.34 kN/m; W2 = 116.7 kN/m.

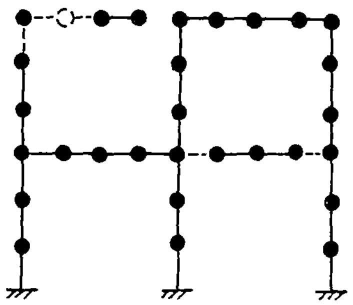
Figure 3.- Progression of failure in example 1.



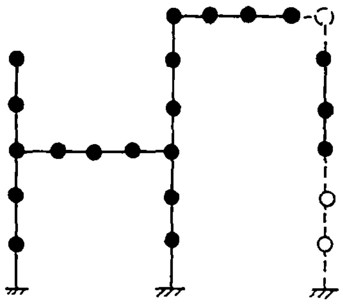
(A) Lateral loads.



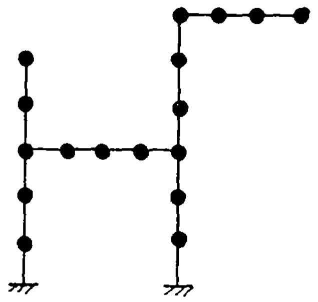
(B) W1 = 23.34 kN; W2 = 70.02 kN.



(C) W1 = 23.34 kN; W2 = 77.8 kN; H = 0 kN.



(D) W1 = 23.34 kN; W2 = 85.58 kN; H = 0 kN.



(E) W1 = 23.34 kN; W2 = 85.58 kN; H = 8.01 kN.

Figure 4.- Pattern of failure in example 2.

THE LANCZOS ALGORITHM WITH SELECTIVE ORTHOGONALIZATION*

B.N. Parlett
Mathematics Department, University of California, Berkeley

D.S. Scott
Union Carbide

ABSTRACT

A new stable and efficient implementation of the Lanczos algorithm is presented.

The Lanczos algorithm is a powerful method for finding a few eigenvalues and eigenvectors at one or both ends of the spectrum of a symmetric matrix A. The algorithm is particularly effective if A is large and sparse in that the only way in which A enters the calculation is through a subroutine which computes Av for any vector v. Thus the user is free to take advantage of any sparsity structure in A and A need not even be represented as a matrix at all.

The simple Lanczos algorithm procedes as follows. Choose q_1, an arbitrary unit vector, and define $\beta_0 = 0$ and $q_0 = 0$. Then for $j = 1,2,\ldots$ do 1 to 5.

```
1. $u_j = Aq_j - q_{j-1}\beta_{j-1}$
2. $\alpha_j = q_j^* u_j$
3. $r_j = u_j - q_j\alpha_j$
4. $\beta_j = \|r_j\|$
5. if $\beta_j = 0$ stop, else $q_{j+1} = r_j/\beta_j$
```

One cycle through 1-5 is a Lanczos step. Note that only q_{j-1} and q_j are needed to compute q_{j+1} which is another attractive feature of the algorithm.

In exact arithmetic, if $Q_j = (q_1,q_2,\ldots,q_j)$ then it can be shown (cf. ref. 1) that Q_j is an orthonormal matrix, i.e. $1-Q_j^*Q_j = 0$, and in addition $Q_j^*AQ_j = T_j$ where

*Research supported by Office of Naval Research Contract N00014-76-C-0013.

$$T_j = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \cdot & \cdot & \\ & & \cdot & \cdot & \beta_{j-1} \\ & & & \beta_{j-1} & \alpha_j \end{bmatrix}$$

is tridiagonal. Furthermore if $T_j = S_j \Theta_j S_j^*$ is the spectral decomposition of T_j with $\Theta_j = \mathrm{diag}(\theta_1^{(j)}, \theta_2^{(j)}, \ldots, \theta_j^{(j)})$ and if $Y_j \equiv (y_1^{(j)}, y_2^{(j)}, \ldots, y_j^{(j)}) = Q_j S_j$ then $(\theta_i^{(j)}, y_i^{(j)})$, $i = 1,2,\ldots,j$, are the (optimal) Rayleigh-Ritz approximations to eigenpairs of A derivable from span(Q_j), the subspace spanned by $q_1, q_2, \ldots, q_j$.

Finally and remarkably, the residual norm of (y_i, θ_i) can be computed without computing the vector y_i. Namely

$$\|Ay_i - y_i\theta_i\| = \beta_{ji}$$

where $\beta_{ji} = \beta_j |s_{ji}|$ and s_{ji} is the (j,i) element of S_j. The quantities β_{ji} show how it is possible for <u>some</u> of the Ritz values (θ's) to be accurate without the appearance of a small β_j. If s_{ji} is tiny then $\theta_i^{(j)}$ will be accurate even if β_j is not small at all.

By construction span(Q_j) = span$(q_1, Aq_1, \ldots, A^{j-1}q_1)$, a Krylov subspace. It can be shown (refs. 2 and 3) that Rayleigh-Ritz approximations converge rapidly as j increases to well separated extreme eigenpairs of A (those near either end of the spectrum).

Unfortunately, as was known to Lanczos when he introduced the algorithm (ref. 4), finite precision causes the computed quantities to diverge completely from their theoretical counterparts. The Lanczos vectors (the q's) inevitably lose their mutual orthogonality and approach linear dependence. This is the infamous "loss of orthogonality" in the Lanczos algorithm.

Lanczos himself recommended that the simple Lanczos algorithm be augmented by a full reorthogonalization of each newly computed q_{j+1}. That is, q_{j+1} is explicitly orthogonalized against all preceding Lanczos vectors (q_i, for $i \leq j$). This not only greatly increases the number of computations required to compute q_{j+1}, it also requires that all the q's be kept in fast store.

This poses a serious dilemma. For large problems it will be too costly to take more than a few steps using full reorthogonalization but linear independence will surely be lost without some sort of corrective procedure. Selective Orthogonalization (hereafter called SO) interpolates between full reorthogonalization and simple Lanczos to obtain the best of both worlds. Robust linear independence is maintained among the columns of Q_j at a cost which is close to that of simple Lanczos.

SO is based on the analysis of the simple Lanczos algorithm in finite arithmetic given by C. Paige in his doctoral thesis (ref. 3). Paige showed that the loss of orthogonality among the columns of Q_j is highly structured when viewed in the basis of Ritz vectors, the columns of $Y_j = Q_j S_j$, rather than in the basis of Q_j itself.

Theorem (Paige). For any step j of the simple Lanczos algorithm and any $i \leq j$, $$\lvert y_i^{(j)*} q_{j+1} \rvert = \epsilon \lVert A \rVert \gamma_{ji} / \beta_{ji}$$ where $\gamma_{ji} \doteq 1$ and ϵ is the working precision.

A proof is given in reference 5.*

It can also be shown (cf. ref. 5 or 6) that β_{ji} is a very good estimate of the residual norm of $y_i^{(j)}$ despite rounding errors. Thus Paige's Theorem shows that q_{j+1} will lose orthogonality only in the direction of Ritz vectors with small β_{ji}, that is those Ritz vectors which are converging to eigenvectors. This can be stated as

$$\boxed{\text{loss of orthogonality} \Leftrightarrow \text{convergence}}$$

To maintain orthogonality among the Lanczos vectors below some threshold value $\tau \leq 1$ it is only necessary to orthogonalize q_{j+1} against those Ritz vectors which satisfy

$$\lvert y_i^{(j)*} q_{j+1} \rvert > \tau \tag{1}$$

By Paige's Theorem equation (1) holds only if $\beta_{ji} \leq \epsilon \lVert A \rVert \gamma_{ji} / \tau \doteq \epsilon \lVert A \rVert / \tau$. Thus it is possible to determine which Ritz vectors achieve the threshold (1) merely by inspecting the β_{ji} which can be computed via a small $(j \times j)$ eigenproblem. There are strong theoretical arguments in favor of the choice $\tau = \sqrt{\epsilon}$ (ref. 5 or 6).

Thus SO modifies the simple Lanczos algorithm by explicitly orthogonalizing q_{j+1} against all the Ritz vectors which satisfy

$$\beta_{ji} \leq \sqrt{\epsilon} \lVert A \rVert \tag{2}$$

We call any Ritz vector which satisfies (2) a <u>good</u> Ritz vector. Good Ritz vectors are already rather well converged and few (if any) of the Ritz vectors at step j will be good, which explains the computational efficiency of the scheme.

In practice it is possible to implement SO even more efficiently. It is not necessary to recompute the good Ritz vectors for orthogonalizing q_{j+1} at each step j as Ritz vectors computed at earlier steps can be used instead.

*The detailed rounding error analysis needed to complete the proof of Paige's Theorem is given in reference 3.

Furthermore it is not necessary to orthogonalize against a particular good Ritz vector at every step.

In conclusion, SO is an efficient way to maintain robust linear independence among the columns of Q_j and so allow the Lanczos algorithm to be run almost as originally conceived. SO points the way to a high quality subroutine package which can be used off the shelf for large sparse symmetric eigenvalue problems.

REFERENCES

1. Kahan, W. and Parlett, B.: How Far Should You Go with the Lanczos Algorithm? In Sparse Matrix Computations, J. Bunch and D. Rose, eds. Academic Press, New York, 1976.

2. Kaniel, S.: Estimates for Some Computational Techniques in Linear Algebra. Mathematics of Computation, vol. 20, no. 95, July 1966, pp. 369-378.

3. Paige, C. C.: The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices. Ph.D. Thesis, University of London, 1971.

4. Lanczos, C.: An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators. J. Res. Nat. Bur. Stand., vol. 45, 1950, pp. 255-282.

5. Scott, D. S.: Analysis of the Symmetric Lanczos Process. Ph.D. Thesis, Mathematics Department, University of California, Berkeley, 1978.

6. Parlett, B. N.: The Symmetric Eigenvalue Problem. Prentice-Hall, Englewood Cliffs, N.J., 1979.

1. Report No. NASA CP-2059	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle RESEARCH IN COMPUTERIZED STRUCTURAL ANALYSIS AND SYNTHESIS		5. Report Date October 1978
		6. Performing Organization Code
7. Author(s) Harvey G. McComb, Jr., compiler		8. Performing Organization Report No. L-12507
		10. Work Unit No. 505-02-33-01
9. Performing Organization Name and Address NASA Langley Research Center Hampton, VA 23665		11. Contract or Grant No.
		13. Type of Report and Period Covered Conference Publication
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		14. Sponsoring Agency Code

15. Supplementary Notes
Most of the papers presented at the symposium are contained in

Noor, Ahmed K.; and McComb, Harvey G., Jr. (eds.): Trends in Computerized Structural Analysis and Synthesis. Pergamon Press, Ltd., 1978.

16. Abstract

A Symposium on Future Trends in Computerized Structural Analysis and Synthesis was held at Washington, D.C., on October 30 - November 1, 1978. NASA Langley Research Center and The George Washington University sponsored the symposium in cooperation with the National Science Foundation and the American Society of Civil Engineers. The purpose of the symposium was to provide a forum for structural technologists, computer hardware experts, and computer software experts to examine recent developments and discuss trends in this very rapidly advancing technology area. This conference publication includes 17 symposium papers; 13 were presented at research-in-progress sessions and 4 at other sessions. The papers deal with five subjects: (1) Potential of new computing systems and artificial intelligence (5 papers), (2) Advances in numerical analysis (3 papers), (3) Structural engineering software systems (1 paper), (4) Adaptive finite element analysis and mesh design (2 papers), and (5) Structural applications (6 papers).

17. Key Words (Suggested by Author(s)) Structural analysis, Structural synthesis, Computer methods, Advanced computer hardware, Structural software systems	18. Distribution Statement Unclassified - Unlimited; Subject Category 39

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	228	$9.50

* For sale by the National Technical Information Service, Springfield, Virginia 22161