NAS8-36555
P. 35

Finite Element Solver for 3–D Compressible Viscous Flows

K. C. Reddy and J. N. Reddy

The University of Tennessee Space Institute
Tullahoma, Tennessee 37388

Interim Report of Contract No. NAS8–36555
For Period: April 1986 – October 1986

Submitted to

NASA/MSFC
Marshall Space Flight Center, AL 35812

by

The University of Tennessee Space Institute
Tullahoma, Tennessee 37388

December 1986

(NASA-CR-179182) FINITE ELEMENT SOLVER FOR 3-D COMPRESSIBLE VISCOUS FLOWS Interim Report, Apr. - Oct. 1986 (Tennessee Univ. Space Inst.) 35 p Avail: NTIS HC A03/MF A01 CSCL 20D G3/34

N87-29763
Unclas
0063976

Table of Contents

1. INTRODUCTION

The space shuttle main engine (SSME) has extremely complex internal flow structure. The geometry of the flow domain is three-dimensional with complicated topology. The flow is compressible, viscous and turbulent with large gradients in flow quantities and regions of recirculations. In recent years computer codes are being developed[1–4] to solve the flow equations in different regions of the SSME such as the hot gas manifold (HGM) region. The analysis of the flow field in SSME involves several tedious steps. One is the geometrical modelling of the particular zone of the SSME being studied. It is usually available in the form of engineering drawings, in terms of algebraic equations for different pieces of the surfaces or in a CAD (computer aided design) system. Accessing the geometry definition, digitizing it and developing surface interpolations suitable for an interior grid generator requires considerable amount of manual effort. There are several types of grid generators available with some general-purpose finite element programs, such as NASTRAN, ADINA, ABQUS, etc. However, these programs require considerable amount of effort on the part of the user to input the geometry to the grid generators; also, the grid generated by those programs are not always the most appropriate grids for the flows being modelled. Next, an efficient and robust computational scheme for solving 3D Navier–Strokes equations has to be implemented for this class of problems. Post processing software has to be adapted to visualize and analyze the computed 3D flow field. Different elements of the above process have been studied in the past and other parts are yet to be developed. The current report discusses the progress made in a project to develop software for the analysis of the flow in the space shuttle main engine and similar complex internal flows.

A CFD code for practical applications should have the following features. It should be reasonably accurate for the class of problems it is designed to solve, with grids that can be accommodated on the present day computers. It should be robust in the sense that it is numerically stable for a broad range of initial and boundary conditions and geometrical parameters. and tolerate some variations in the grid resolution and structure. It should be computationally efficient for obtaining accurate solutions with reasonable computational and human resources. Standard of efficiency, however, is relative and it can only be measured against the current CFD software or which can be foreseen in the immediate future. Another important aspect of a CFD code is its usability, as to how much effort a user has to expend to solve practical problems with it.

For computing the viscous compressible flow inside the main engine where the flow undergoes complex turns through various chambers and ducts, it is necessary to discretize the physical space with several competing requirements. The geometry of the internal surfaces is typically represented in a CAD system or in some equivalent form by the designer. The surface data representation should be interfaced with suitable interpolation software. The refined spline surface representation of the flow boundaries will be the input for the grid generation routines. The topology of the grid structure depends on the flow solver algorithm to be used. Finite difference codes usually impose constraints on the grid structure such as the separability of the indices for efficient computational procedures, while the finite element method can be implemented with less stringent requirements on the grid structure. The grid should provide reasonable resolution of the flow field within the limits of the grid selected by the user. This requires providing more grid points and/or special methods in regions of large gradients of flow quantities, such as the viscous zones near solid boundaries. The gird should meet certain smoothness requirements so that the metrics of the curvilinear grid can be computed numerically and the computed metrics are nonsingular. Unreasonably skewed grid cells or elements, and singular points in the grid where the local transformation of the physical space to computational space has very small or very large Jacobians, should be avoided if at all possible. Otherwise such grids will require special handling by the flow solver algorithm and also may give rise to numerical inaccuracies and instabilities.

There are several grid generation techniques and special purpose codes which can generate reasonable grids for simple two-dimensional and three-dimensional geometries, for both internal and external flows. These techniques fall under two classes: algebraic generators and elliptic generators. Algebraic generators use various interpolation and stretching functions while elliptic generators solve a set of elliptic partial differential equations. While both techniques are effective for simple geometric regions, it is usually difficult to use them to develop a composite grid over a complex internal flow domain. Finite element community have developed extensive amount of software for generating algebraic grid suitable for finite element solvers. For example, NASTRAN (a general purpose finite element program primarily developed for structural analysis) contains grid generators for 2D and 3D structures. Also, the program PATRAN (developed by PDA Engineering) contains 2D and 3D grid generators and pre– and post processing capabilities. In the current project some parts of the software such as PATRAN will be adapted and developed to generate body conforming, curvilinear finite element meshes of the flow domains inside the SSME.

Computation of the flow field inside the space shuttle main engine requires the application of the state-of-the-art CFD technology. Several computer codes[1-4] are under development to solve three dimensional Navier–Stokes equations with different turbulence models for analyzing the SSME internal flow, such as the flow through the how gas manifold (HGM). The computational methods[5-6] used in the Navier–Stokes codes fall into two major categories: finite difference and finite element methods. Some of the algorithms are designed to solve the unsteady compressible Navier–Stokes equations, either by explicit or by implicit factorization methods, using several hundred or thousands of time steps to reach a steady-state solution asymptotically. Other algorithms attempt to solve the steady-state equations by relaxation methods. All of them require body-fitting curvilinear grids with sufficient resolution. Grid requirements, however, differ greatly with the region being modelled and the algorithm used. Implicit factorization based on finite differences typically use global numerical transformations whereby the transformed grid in the computational space is uniform and rectilinear. This requires the grid to have indices which are separable in the three directions for three dimensional problems, and also be reasonably smooth. However, such requirements may introduce grid singularities when complicated domains are discretized. Flow solver algorithm will have to deal with such grid singularities. Explicit schemes and finite element algorithms have less stringent requirements on the grid structure. However, explicit schemes are slow to converge because of the stability limitations on time step, particularly for large scale viscous problems.

The finite element method is characterized by three basic features which are credited for the enormous success, the method has enjoyed in the solution of practical engineering problems[6]. The first feature is that every computational domain is viewed as a collection of simple subdomains, called finite elements. This feature allows us to represent complicated geometries as assemblages of simple parts. It is a desirable feature in the solution of flow problems in complex configurations, not only to describe the complex geometry but also to choose the most suitable computational grid for a particular flow. This feature also allows us to place or remove any obstructions routinely into the flow field. The second feature is that over each element the solution is represented by polynomials of desired degree. This allows us to compute the solution as a continuous function of position instead of at selected few points. Desired degree of approximation (e.g., linear, quadratic, etc.) can be easily and routinely specified without rewriting the whole or parts of the program. The third feature is that the relationship (i.e., the algebraic equations) between the solution and its dual variables (i.e., velocities and forces) is developed using a variational method, such as the Galerkin method. The boundary conditions are then applied on the algebraic equations directly before solving. The three features of the finite element method also allow the easy development and interfacing of pre– and post–processors, and user-defined subroutines for equations for state and turbulence models.

The Galerkin finite element method (i.e., the weight functions are the same as the approximation functions) applied to flow problems always results in implicit schemes. The weighted-residual (or Petrov–Galerkin) method, in which the weight functions are different from the approximation functions, can be used in conjunction with explicit schemes to obtain explicit final equations. For example, by selecting the weight functions to be orthogonal to the approximation functions, the mass matrix can be diagonalized. However, such considerations are entirely in the interest of obtaining explicit schemes and not necessarily in the interest of accuracy or even computational efficiency. In the current project implicit finite element scheme with suitable dissipation terms for stability is being developed. A relaxation procedure, known as the locally implicit scheme is being developed to solve the coupled set of algebraic equations efficiently.

In the following sections we discuss the technical approach to the development of the finite element scheme and the relaxation procedure. Appendix I contains the details of the equations derived and Appendix II has a listing of the three dimensional finite element code for the compressible Navier–Stokes equations. Future reports will discuss the numerical results for specific problems.

2. TECHNICAL APPROACH

2.1 GOVERNING EQUATIONS

In an Eulerian description, used most extensively in fluid dynamics, the coordinate system is fixed in space rather than in the body, and measurements of density, velocity, pressure, etc. are made for the material particle that happens to be in a given location at that particular time. The basic equations of a continuous medium in the Eulerian description are:

Continuity Equation. – The law of conservation of mass leads to

$$\frac{\partial}{\partial t}(\rho) + \underline{\nabla} \cdot (\rho \underline{v}) = 0 \tag{1}$$

where ρ is the density of the medium, $\underline{v}$ is the velocity vector and $\underline{x} = (x_1, x_2, x_3)$ the spatial coordinates.

Equations of Motion. – The law of balance of linear momentum leads to the celebrated Eulerian equation of motion,

$$\frac{\partial}{\partial t}(\rho \underline{v}) + \underline{\nabla} \cdot (\rho \underline{v}\underline{v}) = \underline{\nabla} \cdot \underline{\sigma} + \underline{F} \tag{2}$$

Here $\underline{F}$ the body force vector (measured per unit volume) and $\underline{\sigma}$ the total stress tensor, which can be divided into hydrostatic and viscous parts:

$$\underline{\sigma} = -p\underline{I} + \underline{\tau} \tag{3}$$

Here p denotes the hydrostatic pressure, $\underline{\tau}$ the viscous (or shear) stress tensor, and $\underline{I}$ denotes the unit tensor.

An application of the law of balance of angular momentum and neglect of microstructural effects such as couple stresses lead to the symmetry of stress tensor,

$$\sigma_{ij} = \sigma_{ji}, \; \tau_{ij} = \tau_{ji} \; \left(\underline{\sigma} = \underline{\sigma}^T\right) \tag{4}$$

Energy Equation. – The law of conservation of energy (the first law of thermodynamics) leads to

$$\frac{\partial}{\partial t}(\rho e) + \underline{\nabla} \cdot (\rho e \underline{v}) = \underline{\nabla} \cdot (\underline{\sigma} \cdot \underline{v}) + \underline{F} \cdot \underline{v} + \rho S - \underline{\nabla} \cdot \underline{q} \tag{5}$$

where e is the total energy per unit mass,

$$e = \varepsilon + \frac{1}{2}\underline{v} \cdot \underline{v}$$

ε being the specific internal energy, S is the rate of internal heat generation per unit mass, and $\underline{q}$ is the heat flux vector or the rate of heat flow per unit area across the surface in the direction of its unit outward normal.

Constitutive Equations. – The thermodynamic pressure p is related to the specific internal energy ε and the density ρ through an equation of state,

$$p = p(\varepsilon, \rho) \tag{6}$$

and the viscous stress is related to the deformation rate tensor $\underline{d}$ through a constitutive equation of the form

$$\underline{\tau} = \underline{\tau}(\underline{d}, \underline{c}) \tag{7}$$

where

$$\underline{\underline{d}} = \frac{1}{2}\left[\underline{\nabla}\underline{v} + (\underline{\nabla}\underline{v})^T\right] \tag{8}$$

and $\underline{\underline{c}}$ is the tensor of viscosities.

For isotropic fluids obeying linear stress-strain relations (i.e. Newtonian fluids) we have

$$\tau_{ij} = 2\mu d_{ij} \tag{9}$$

where μ is the viscosity.

Initial Conditions. – At time $t = 0$, values of all the dependent variables $(\rho, \underline{v}, e, p)$ must be specified in the entire domain. It is not essential to specify all of these quantities at the same set of points.

Boundary Conditions. – Depending on the type of the boundary (e.g., rigid boundary, free surface, interface, plane of symmetry, etc.), there are different kinds of boundary conditions in a problem. At a rigid boundary, the normal component of the particle velocity must coincide with the normal component of the velocity of the rigid boundary. For a fixed (in time) boundary, the normal component of the particle velocity must be zero at that boundary. A plane of symmetry can be interpreted as a fixed boundary. On a free surface, the pressure must vanish. At an interface (and at a contact discontinuity) the pressure and the normal component of particle velocity must be continuous, and the density, internal energy and the tangential component of particle velocity may be discontinuous (i.e. jumps may occur). Across moving shock fronts, the Rankine–Hugoniot relations must be satisfied.

2.2 FINITE ELEMENT MODEL

Writing the governing equations in terms of the velocities, pressure, density and internal energy, we obtain

$$\frac{\partial \rho}{\partial t} - \underline{\nabla} \cdot (\rho \underline{v}) = 0$$

$$\frac{\partial}{\partial t}(\rho \underline{v}) + \underline{\nabla} \cdot (\rho \underline{v}\underline{v}) + \underline{\nabla} p = \mu \underline{\nabla} \cdot \underline{\underline{d}} + \underline{F}$$

$$\frac{\partial}{\partial t}(\rho e) + \underline{\nabla} \cdot (\rho e \underline{v}) + \underline{\nabla} \cdot (p \underline{v}) = \mu \underline{\nabla} \cdot (\underline{\underline{d}} \cdot \underline{v}) + \underline{F} \cdot \underline{v} - \underline{\nabla} \cdot \underline{q} \tag{10}$$

and p is given by the equation of state. If we assume that the body force, heat flux, and the internal heat generation are zero, the last two terms in the energy equation dropout.

For simplicity and computational convenience, we denote

$$\rho \underline{v} = \underline{V}, \quad \rho e = E$$

so that (10) become

$$\frac{\partial}{\partial t}(\rho) + \underline{\nabla} \cdot \underline{V} = 0$$

$$\frac{\partial}{\partial t}(\underline{V}) + \underline{\nabla} \cdot (\underline{v}\underline{V}) + \underline{v}P = \mu \underline{\nabla} \cdot \underline{d}$$

$$\frac{\partial}{\partial t}(E) + \underline{\nabla} \cdot (E\underline{v}) + \underline{\nabla} \cdot (P\underline{v}) = \mu \underline{\nabla} \cdot (\underline{d} \cdot \underline{v}) \tag{11}$$

We seek approximate solutions to Eq. (11) using the finite element method.

Spatial Approximation. - Finite element approximations to Eq. (11) are sought over a typical element Ω^e:

$$\rho = \sum_{j=1}^{N} \rho_j \psi_j(\underline{x})$$

$$V_i = \sum_{j=1}^{N} V_i^j(t)\psi_j(\underline{x})$$

$$E = \sum_{j=1}^{N} E_j(t)\psi_j(\underline{x}) \tag{12}$$

where $\psi_j(\underline{x})$ are the interpolation functions in space, ρ_j, V_i^j, and E_j are the unknown, time-dependent, nodal values to be determined. In Eq. (12) we have assumed for simplicity the same type (linear or quadratic) of interpolation functions for all the variables. The Galerkin approximation amounts to seeking solutions to Eq. (11) in the form (12) by making the errors in Eq. (11) orthogonal to the trial functions. This leads to the following local set of nonlinear ordinary differential equations in time.

$$[A]\{\dot{\rho}\} + [B]\{V\} = 0$$

$$[A]\{\dot{V}\} + [N]\{V\} = \{Q\}$$

$$[A]\{\dot{E}\} + [M]\{E\} = \{R\} \tag{13}$$

Here the superposed dot denote total differentiation with respect to time, and

$$A_{ij} = \int_{\Omega^e} \psi_\alpha \psi_\beta \, d\underline{x}, \; B_{ij} = \int_{\Omega^e} \psi_i \frac{\partial \psi_j}{\partial x_k} \, d\underline{x}$$

$$N_{ij} = \int_{\Omega^e} \psi_i \sum_{k=1}^{3} \frac{\partial}{\partial x_k} (v_k \psi_j)\, d\underline{x} + \int_{\Omega^e} \mu \underline{\nabla} \psi_i \cdot \underline{\nabla} \psi_j d\underline{x},$$

$$M_{ij} = \int_{\Omega^e} \psi_i \sum_{k=1}^{3} \frac{\partial}{\partial x_k} (v_k \psi_j)\, d\underline{x}$$

$$Q_{ik} = -\int_{\Omega^e} \psi_i \frac{\partial p}{\partial x_k} d\underline{x},\ R_i = -\int_{\Omega^e} \psi_i \sum_{k=1}^{3} \frac{\partial}{\partial x_k} (p v_k)\, d\underline{x} + \int_{\Omega^e} \mu \underline{\nabla} \psi_i \underline{d} d\underline{x} \tag{14}$$

where $\int_{\Omega^e}$ denotes integration over the element volume.

Equations (13) are to be further approximated (or numerically integrated with respect to time) to obtain a set of simultaneous algebraic equations.

<u>Temporal Approximations</u>. - Equations (13) are of the general form

$$[A]\{\dot{U}\} + [B]\{U\} = \{Q\}, \tag{15}$$

We approximate $U(t)$ by

$$U(t) = \sum_{j=1}^{n} U_j \phi_j(t)\ , \qquad m = 1, 2, \ldots, M \tag{16}$$

where $\phi_j(t)$ are approximation functions in time. Here we assume that ϕ_j are linear in t (i.e., $n = 2$):

$$\phi_1(t) = \left(1 - \frac{t}{\Delta t}\right),\ \ \phi_2(t) = \frac{t}{\Delta t},\quad 0 \le t \le \Delta t$$

where Δt denotes the time increment. Then the time derivative of U is given by

$$\dot{U} = (U_2 - U_1)\,/\Delta t \tag{17}$$

It can be readily interpreted that U_1 is the value of U at time $t = n(\Delta t)$, and U_2 is the value of U at $t = (n+1)\Delta t$. Substituting Eq. (16) and (17) into Eq. (15), multiplying with $\phi_2(t)$ and integrating over 0 to Δt, we obtain

$$\left[A + \frac{2}{3}\Delta t B\right]\{U_{n+1}\} = \Delta t\{Q\} + \left[A - \frac{\Delta t}{3} B\right]\{U_n\} \tag{18}$$

Thus the unknown vector $\{U_{n+1}\}$ can be solved in terms of the known vector $\{U_n\}$. It should be noted that the temporal approximations (18) can be applied to the local set (13). There are other methods of time integration which can be incorporated into the code.

Equations (18) can be assembled in the usual manner to obtain the global equations, which must be solved iteratively (after imposing the initial and boundary conditions of the problem) for the nodal values, as the resulting algebraic equations are nonlinear. A flow chart of the computer program based on the formulation presented above is shown in Fig. 1.

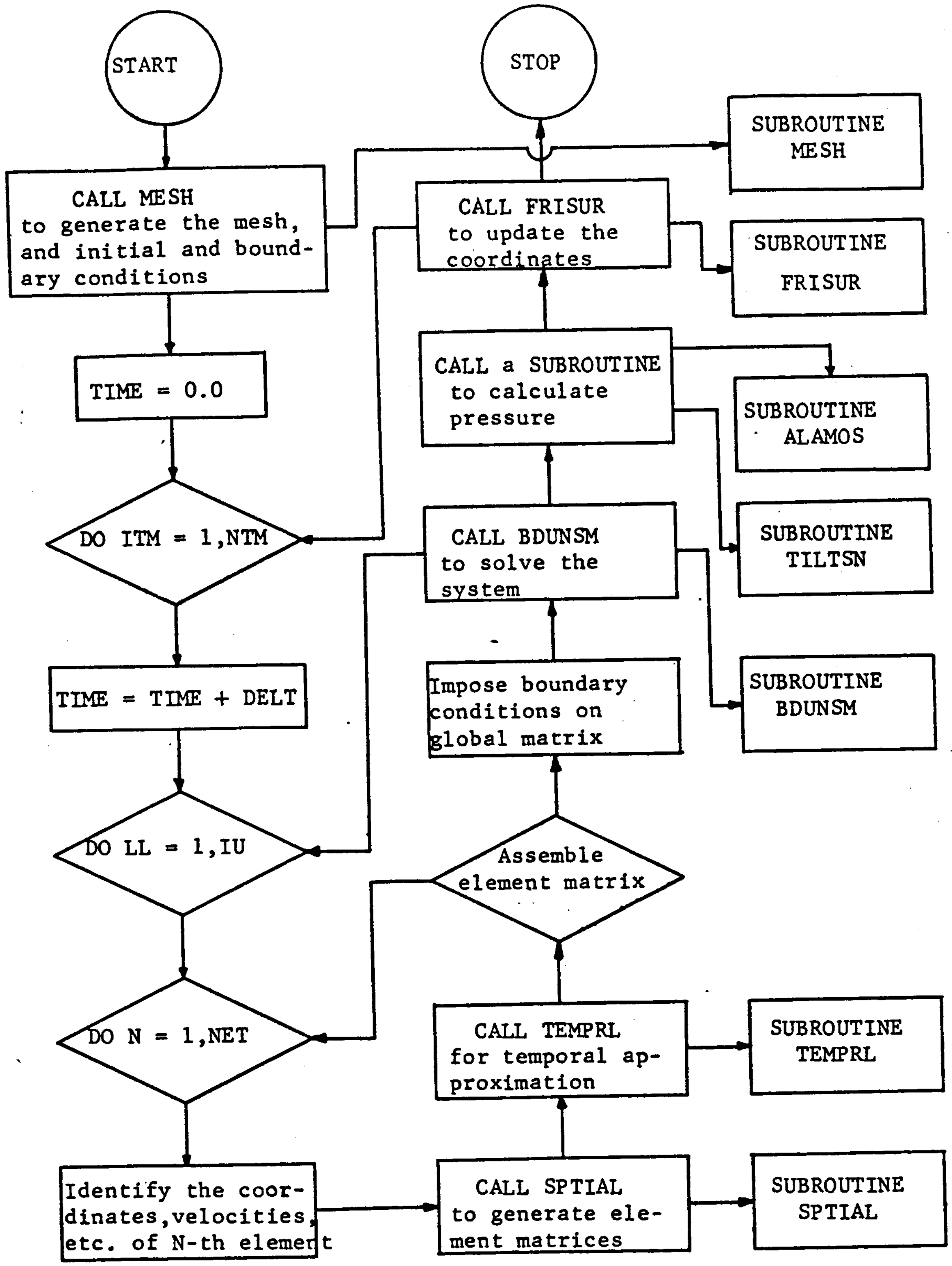

Fig. 1. Flow Chart of the Computer Program

2.3 LOCALLY IMPLICIT APPROXIMATIONS

For large problems, it is not possible to solve the global (linearized) equations by direct methods. An efficient iterative method of solution has been formulated and it is known as the locally implicit method[7]. This is based on a modified Gauss–Seidel iteration technique with a symmetric inner iteration.

The linearized equations (18) for an element j can be written in the form

$$L_j \ \Delta U_j = \ Res \ (U^n) + \sum_{K \neq j} L_k \ \Delta U_k + Q \tag{19}$$

where $\Delta U_j = U_j^{n+1} - U_j^n$ and the summation on the right hand side of (19) is limited to the elements surrounding the element j, with which the finite element equations over the element j are coupled with. Equations (19) are solved by an iteration

$$LM_j \ \delta \ \Delta U_j = \ Res \ (U^n) + \sum_{k \neq j} L_k \ \Delta U_k^{(\,)} - L_j \ \Delta U_j^{(m)} + Q \tag{20}$$

where $\Delta U_j^{m+1} = \Delta U_j^{(m)} + \delta \ \Delta U_j$, LM_j is a modification to the matrix L_j so as to achieve stability and rapid convergence of the iteration process. $\Delta U_k^{(\,)}$ denotes either $\Delta U_k^{(m+1)}$ or $\Delta U_k^{(m)}$ depending on the latest available iterates for ΔU_k. The iteration process of the equation (20) is carried out starting at a different corner of the computational space for each iteration. Eight such iterations complete one symmetric modified Gauss–Seidel iteration per time step for 3–dimensional problems. This is a stable process with fast convergence properties in a local sense. It amounts to solving the equations (15) implicitly in a local sense for each node. It is not necessary to achieve full convergence at each time step if we need only the steady state solution. One symmetric sweep per time step is adequate. This process has been tested over a variety of model equations such as the 3–dimensional Poisson equation and one dimensional Burger's equation. The same procedure has also been shown to work for two dimensional Euler equations with finite volume discretizations and artificial dissipation terms.

3. REFERENCES

1. Chang, L. C., Kwak, D., Dao, S. C., and Rosen, R., "A Three-Dimensional Incompressible Flow Simulation Method and Its Application to the Space Shuttle Main Engine, Part 1 – Laminar Flow", AIAA-85-0175, AIAA 23rd Aerospace Sciences Meeting, Reno, January 1985.

2. Spradley, L. W., "Application of Computational Methods to Internal and External Fluid Flows", Proceedings of a Workshop on Computational Fluid Dynamics in Aerospace Design, UTSI, Tullahoma, TN, June 1985.

3. Mukherjee, T., Przekwas, A. J., Tam, L. T., Holland, R. L. and Costes, N. C., "A Multidomain Global-Model Analysis of SSME, Workshop on Computational Fluid Dynamics, NASA Marshall Space Flight Center, Huntsville, AL, April 1986.

4. Rhie, J., "A General Purpose Navier-Stokes Solver for Internal Propulsion System Flows", AIAA- 86-0207, Aerospace Sciences Meeting, January 1986.

5. Reddy, K. C. and J. S. Steinhoff, (eds.), Computational Fluid Dynamics, proceedings of a workshop held at The University of Tennessee Space Institute, Tullahoma, TN, March 12-16, 1984.

6. Reddy, J. N., An Introduction to the Finite Element Method, McGraw-Hill, New York, 1984.

7. Reddy, K. C., "A Locally Implicit Scheme for Elliptic Problems", A Workshop on Computational Fluid Dynamics, NASA Marshall Space Flight Center, Huntsville, AL, April 1986.

Appendix I

Finite Element equations for Navier–Stokes Equations

Variational formulation over an element for the Navier–Stokes equations in non-conservation form:

$$0 = \int_{\Omega^e} w_1 \left[\frac{\partial \rho}{\partial t} + u\frac{\partial \rho}{\partial x} + v\frac{\partial \rho}{\partial y} + w\frac{\partial \rho}{\partial z} + \rho\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right] dV \tag{1}$$

$$\begin{aligned}
0 = \int_{\Omega^e} & \left\{ \rho w_2 \frac{\partial u}{\partial t} + w_2 \rho \left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z}\right) - p\frac{\partial w_2}{\partial x} + 2\mu\frac{\partial w_2}{\partial x} + 2\mu\frac{\partial w_2}{\partial x}\frac{\partial u}{\partial x} \right.\\
& \left. +\mu\frac{\partial w_2}{\partial y}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) + \mu\frac{\partial w_2}{\partial z}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) - \lambda\frac{\partial w_2}{\partial x}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right\} dV \\
& - \oint_{\Gamma^e} t_x w_2 ds
\end{aligned} \tag{2}$$

$$\begin{aligned}
0 = \int_{\Omega^e} & \left\{ \rho w_3 \frac{\partial v}{\partial t} + w_3 \rho \left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z}\right) - p\frac{\partial w_3}{\partial y} + \mu\frac{\partial w_3}{\partial x}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \right.\\
& \left. +2\mu\frac{\partial w_3}{\partial y}\frac{\partial v}{\partial y} + \mu\frac{\partial w_3}{\partial z}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) - \lambda\frac{\partial w_3}{\partial y}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right\} dV \\
& - \oint_{\Gamma^e} t_y w_3 ds
\end{aligned} \tag{3}$$

$$\begin{aligned}
0 = \int_{\Omega^e} & \left\{ \rho w_4 \frac{\partial w}{\partial t} + w_4 \rho \left(u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z}\right) - p\frac{\partial w_4}{\partial z} + \mu\frac{\partial w_4}{\partial x}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) \right.\\
& \left. +\mu\frac{\partial w_4}{\partial y}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) + 2\mu\frac{\partial w_4}{\partial z}\frac{\partial w}{\partial z} - \lambda\frac{\partial w_4}{\partial z}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right\} dV \\
& - \oint_{\Gamma^e} t_z w_4 ds
\end{aligned} \tag{4}$$

$$\begin{aligned}
0 = \int_{\Omega^e} & \left\{ \rho c_v w_5 \frac{\partial T}{\partial t} + \rho c_v w_5 \left(u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} + w\frac{\partial T}{\partial z}\right) - w_5 \rho Q \right.\\
& + k_x\frac{\partial w_5}{\partial x}\frac{\partial T}{\partial x} + k_y\frac{\partial w_5}{\partial y}\frac{\partial T}{\partial y} + k_z\frac{\partial w_5}{\partial z}\frac{\partial T}{\partial z} + w_5 p\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) \\
& - w_5\left(\frac{\partial u}{\partial x}\tau_{xx} + \frac{\partial u}{\partial y}\tau_{xy} + \frac{\partial u}{\partial z}\tau_{xz} + \frac{\partial v}{\partial x}\tau_{xy} + \frac{\partial v}{\partial y}\tau_{yy} + \frac{\partial v}{\partial z}\tau_{yz} \right.\\
& \left.\left. + \frac{\partial w}{\partial x}\tau_{xz} + \frac{\partial w}{\partial y}\tau_{yz} + \frac{\partial w}{\partial z}\tau_{zz}\right)\right\} dV \\
& - \oint_{\Gamma^e} q w_5 ds
\end{aligned} \tag{5}$$

where

$$t_x = \sigma_x n_x + \sigma_{xy} n_y + \sigma_{xz} n_z, \qquad t_y = \sigma_{xy} n_x + \sigma_y n_y + \sigma_{zy} n_z$$

$$t_z = \sigma_{xz} n_x + \sigma_{yz} n_y + \sigma_z n_z, \qquad q = K_x \frac{\partial T}{\partial x} n_x + K_y \frac{\partial T}{\partial y} n_y + K_z \frac{\partial T}{\partial z} n_z$$

FINITE ELEMENT FORMULATION

Let $\rho = \sum_{j=1}^{n} \rho_j \psi_j(x,y,z), \quad u = \sum_{j=1}^{n} U_j \psi_j(x,y,z),$ etc.

Equations (1) – (5) can be formulated as

$$\begin{aligned} [M^1]\{\dot{\rho}\} + [K^1]\{\rho\} &= \{F^1\} \\ [M^2]\{\dot{U}\} + [K^2]\{U\} &= \{F^2\} \\ [M^2]\{\dot{V}\} + [K^3]\{V\} &= \{F^3\} \\ [M^2]\{\dot{W}\} + [K^4]\{W\} &= \{F^4\} \\ [M^3]\{\dot{T}\} + [K^5]\{T\} &= \{F^5\} \end{aligned}$$

where

$$M_{ij}^1 = \int_{\Omega^e} \psi_i \psi_j dV, \;\; K_{ij}^1 = \int_{\Omega^e} \psi_i \left(u \frac{\partial \psi_j}{\partial x} + v \frac{\partial \psi_j}{\partial y} + w \frac{\partial \psi_j}{\partial z} \right) dV$$

$$F_i^1 = - \int_{\Omega^e} \rho \psi_i \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) dV$$

$$\begin{aligned} M_{ij}^2 = \int_{\Omega^e} \rho \psi_i \psi_j dV, \quad K_{ij}^2 = \int_{\Omega^e} \Bigg[& \rho \psi_i \left(u \frac{\partial \psi_j}{\partial x} + v \frac{\partial \psi_j}{\partial y} + w \frac{\partial \psi_j}{\partial x} \right) + 2\mu \frac{\partial \psi_i}{\partial x} \frac{\partial \psi_j}{\partial x} \\ & + \mu \frac{\partial \psi_i}{\partial y} \frac{\partial \psi_j}{\partial y} + \mu \frac{\partial \psi_i}{\partial z} \frac{\partial \psi_j}{\partial z} - \lambda \frac{\partial \psi_i}{\partial x} \frac{\partial \psi_j}{\partial x} \Bigg] dV \end{aligned}$$

$$F_i^2 = \int_{\Omega^e} \left[p\frac{\partial \psi_i}{\partial x} - \mu \left(\frac{\partial \psi_i}{\partial y}\frac{\partial v}{\partial x} + \frac{\partial \psi_i}{\partial z}\frac{\partial w}{\partial x} \right) + \lambda \frac{\partial \psi_i}{\partial x} \left(\frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) \right] dV + \oint_{\Gamma^e} t_x \psi_i ds$$

$$K_{ij}^3 = \int_{\Omega^e} \left[\rho\psi_i \left(u\frac{\partial \psi_j}{\partial x} + v\frac{\partial \psi_j}{\partial y} + w\frac{\partial \psi_j}{\partial x} \right) + \mu \left(\frac{\partial \psi_i}{\partial x}\frac{\partial \psi_j}{\partial x} + 2\frac{\partial \psi_i}{\partial y}\frac{\partial \psi_j}{\partial y} + \frac{\partial \psi_i}{\partial z}\frac{\partial \psi_j}{\partial z} \right) - \lambda \frac{\partial \psi_i}{\partial y}\frac{\partial \psi_j}{\partial y} \right] dV$$

$$F_i^3 = \int_{\Omega^e} \left[p\frac{\partial \psi_i}{\partial y} + \mu \left(\frac{\partial \psi_i}{\partial x}\frac{\partial u}{\partial y} + \frac{\partial \psi_i}{\partial z}\frac{\partial w}{\partial y} \right) - \lambda \frac{\partial \psi_i}{\partial y} \left(\frac{\partial u}{\partial x} + \frac{\partial w}{\partial z} \right) \right] dV - \oint_{\Gamma^e} t_y \psi_i ds$$

$$K_{ij}^4 = \int_{\Omega^e} \left[\psi_i \rho \left(u\frac{\partial \psi_i}{\partial x} + v\frac{\partial \psi_i}{\partial y} + w\frac{\partial \psi_i}{\partial z} \right) + \mu \left(\frac{\partial \psi_i}{\partial x}\frac{\partial \psi_j}{\partial x} + \frac{\partial \psi_i}{\partial y}\frac{\partial \psi_j}{\partial y} + 2\frac{\partial \psi_i}{\partial z}\frac{\partial \psi_j}{\partial z} \right) - \lambda \frac{\partial \psi_i}{\partial z}\frac{\partial \psi_j}{\partial z} \right] dV$$

$$F_i^4 = \int_{\Omega^e} \left[p\frac{\partial \psi_i}{\partial z} + \mu \left(\frac{\partial \psi_i}{\partial x}\frac{\partial u}{\partial z} + \frac{\partial \psi_i}{\partial y}\frac{\partial v}{\partial z} \right) - \lambda \frac{\partial \psi_i}{\partial z} \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) \right] dV - \oint_{\Gamma^e} t_z \psi_i ds$$

$$K_{ij}^5 = \int_{\Omega^e} \left[\rho c_v \psi_i \left(u\frac{\partial \psi_j}{\partial x} + v\frac{\partial \psi_j}{\partial y} + w\frac{\partial \psi_j}{\partial z} \right) + K_x \frac{\partial \psi_i}{\partial x}\frac{\partial \psi_j}{\partial x} + K_y \frac{\partial \psi_i}{\partial y}\frac{\partial \psi_j}{\partial y} + K_z \frac{\partial \psi_i}{\partial z}\frac{\partial \psi_j}{\partial z} \right] dV$$

$$F_i^5 = \int_{\Omega^e} \left[\psi_i \rho Q - \psi_i p \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) + \psi_i \left(\frac{\partial u}{\partial x}\tau_{xx} + \cdots \right) \right] dV + \oint_{\Gamma^e} q \psi_i ds$$

$$M_{ij}^3 = \int_{\Omega^e} \rho c_v \psi_i \psi_j dV$$

ALTERNATIVE (CONSERVATION) FORM OF EQUATIONS

$$\begin{aligned}\text{Let}\quad \vec{V} &= \rho\vec{v} \quad (U = \rho u,\ \ V = \rho v,\ \ W = \rho w)\\ E &= \rho\epsilon,\quad \vec{f} = \vec{0},\quad Q = 0\end{aligned}$$

The governing equations are

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) + \frac{\partial}{\partial y}(\rho v) + \frac{\partial}{\partial z}(\rho w) &= 0\\
\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}(Uu) + \frac{\partial}{\partial y}(Uv) + \frac{\partial}{\partial z}(Uw) &= -\frac{\partial p}{\partial x} + \frac{\partial \sigma_x}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z}\\
\frac{\partial V}{\partial t} + \frac{\partial}{\partial x}(Vu) + \frac{\partial}{\partial y}(Vv) + \frac{\partial}{\partial z}(Vw) &= -\frac{\partial p}{\partial y} + \frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z}\\
\frac{\partial W}{\partial t} + \frac{\partial}{\partial x}(Wu) + \frac{\partial}{\partial y}(Wv) + \frac{\partial}{\partial z}(Ww) &= -\frac{\partial p}{\partial z} + \frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_z}{\partial z}\\
\frac{\partial E}{\partial t} + \frac{\partial}{\partial x}(uE) + \frac{\partial}{\partial y}(vE) + \frac{\partial}{\partial z}(wE) &= -\vec{\nabla}\cdot\vec{q} + \vec{\sigma} : \vec{D}
\end{aligned}$$

The finite–element equations are

$$\begin{aligned}
[M]\{\dot{\rho}\} + [K]\{\rho\} &= \{F^1\}\\
[M]\{\dot{U}\} + [K]\{U\} &= \{F^2\}\\
[M]\{\dot{V}\} + [K]\{V\} &= \{F^3\}\\
[M]\{\dot{W}\} + [K]\{W\} &= \{F^4\}\\
[M]\{\dot{E}\} + [K]\{E\} &= \{F^5\}
\end{aligned}$$

where

$$M_{ij} = \int_{\Omega^e} \psi_i \psi_j dV, \quad K_{ij} = \int_{\Omega^e} \psi_i \left[\frac{\partial}{\partial x}(u\psi_j) + \frac{\partial}{\partial y}(v\psi_j) + \frac{\partial}{\partial z}(w\psi_j) \right] dV$$

$$= \int_{\Omega^e} \psi_i \left[u\frac{\partial \psi_j}{\partial x} + v\frac{\partial \psi_j}{\partial y} + w\frac{\partial \psi_j}{\partial z} + \psi_j \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) \right] dV$$

$$F_i^1 = 0, \quad F_i^2 = \int_{\Omega^e} \psi_i \left[-\frac{\partial p}{\partial x} + \frac{\partial \sigma_x}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} \right] dV$$

$$F_i^3 = \int_{\Omega^e} \psi_i \left[-\frac{\partial p}{\partial y} + \frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} \right] dV$$

$$F_i^4 = \int_{\Omega^e} \psi_i \left[-\frac{\partial p}{\partial z} + \frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_z}{\partial z} \right] dV$$

$$F_i^5 = \int_{\Omega^e} \left(-\vec{\nabla} \cdot \vec{q} + \vec{\sigma} : \vec{D} \right) \psi_i dV$$

This formulation is a natural extension of the finite element model for inviscid flows and is applicable for compressible viscous flows from low subsonic to supersonic flows with suitable addition of stabilizing terms (artificial viscosity). For highly viscous, low Mach number internal flows there is no need for the addition of artificial viscosity. This formulation is coded in the computer program COMPR3D and is listed in Appendix II.

```
C                      *************************
C                      PROGRAM    C O M P R 3 D
C                      *************************
C
C
C     AN IN-CORE  FINITE-ELEMENT ANALYSIS COMPUTER PROGRAM  FOR THE
C     ANALYSIS OF  UNSTEADY NAVIER-STOKES  EQUATIONS  GOVERNING THE
C     FLOW OF A  VISCOUS   COMPRESSILE FLUID  IN  THREE-DIMENSIONAL
C     ENCLOSURES.  THE CONSERVATION FORM OF THE EQUATIONS  IS  USED
C     TO DEVELOP THE FINITE ELEMENT MODEL.    THE PROGRAM IS  UNDER
C     DEVELOPMENT BY J. N. REDDY, 505 CRANWELL CIRCLE, BLACKSBURG.
C
C  ******************************************************************
C
C          D E S C R I P T I O N   O F   T H E   V A R I A B L E S
C
C     AMU.......VISCOSITY OF THE FLUID (=1.0/REYNALDS NUMBER HERE)
C     BETA......ACCELERATION PARAMETER IN THE NONLINEAR ITERATION
C     DT........TIME STEP FOR THE UNSTEADY ANALYSIS
C     DX,DY,DZ..ARRAYS OF COORDINATES OF THE NODES ALONG THE X, Y
C               AND Z COORDINATES, RESPECTIVELY (FOR MESH ONLY)
C     CV........CONSTANT IN THE EQUATION OF STATE (GAS CONSTANT)
C     DRE.......ARRAY OF THE INCREMENTS OF REYNALDS NUMBER
C     EPS.......ERROR PERMITTED BETWEEN THE SOLUTIONS OF TWO CONSECU-
C               TIVE TIMES (A CHECK FOR STEADY STATE SOLUTION)
C     ELF.......ELEMENT SOURCE VECTOR
C     GSTIF.....ASSEMBLED COEFFICIENT MATRIX; THE LAST COLUMN OF THE
C               MATRIX CONTAINS THE SOURCE VECTOR BEFORE GOING INTO
C               SUBROUTINE 'BDUNSM', AND THE SOLUTION WHEN RETURNS
C     GC........SOLUTION FROM THE CURRENT ITERATION
C     GF........CONVERGED SOLUTION FOR THE PREVIOUS TIME
C     GP........SOLUTION FROM THE PREVIOUS ITERATION
C     IEL.......ELEMENT TYPE: IEL=1, LINEAR; IEL=2, QUADRATIC
C     IMESH.....INDICATOR FOR MESH GENERATION (ONLY FOR RECTANGULAR
C               DOMAINS):IMESH=0, NOT GENERATE; IMESH=1, GENERATE
C     ISBC......ARRAY OF NODE NUMBER AND DEGREE OF FREEDOM THAT IS
C               SPECIFIED:  ISBC(I,1)=NODE NUMBER
C                           ISBC(I,2)=DEGREE OF FREEDOM
C     ISBF......ARRAY OF NODE NUMBER AND D.O.F. OF SPECI. 'FORCES'
C     ITEM......INDICATOR FOR UNSTEADY (ITEM=1) AND STEADY (ITEM=0)
C               SOLUTION
C     ITMAX.....MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR CONVERGENCE
C               IN THE NONLINEAR (N-S EQUATIONS) ANALYSIS; IT ALSO
C               SERVES AS AN INDICATOR FOR LINEAR (ITMAX=1) ANALYSIS
C     NBW.......FULL BAND WIDTH OF 'GSTIF' FOR VELOCITIES
C     NCMAX.....COLUMN DIMENSION OF GSTIF IN THE DIMENSION STATEMENT
C     NDF.......DEGREES OF FREEDOM AT EACH NODE (RHO,U,V,W,E)
C     NEM.......NUMBER OF ELEMENTS IN THE MESH
C     NEQ.......NUMBER OF EQUATIONS IN THE MODEL (=NNM)
C     NHBW......HALF BAND WIDTH OF 'GSTIF' FOR VELOCITIES
C     NNM.......NUMBER OF NODES IN THE MESH
C     NOD.......CONNECTIVITY MATRIX
C     NOSTRS....INDICATOR FOR THE STRESS CALCULATION: NOSTRS=0, DO
C               NOT COMPUTE STRESSES; NOSTRS=1, COMPUTE STRESSES
C     NPRNT.....INDICATOR FOR PRINTING (NPRNT=1) OF ELEMENT SIFFNESS
C               MATRIX (NPRNT=0, DO NOT PRINT)
C     NRENLD....NUMBER OF REYNALDS NUMBERS FOR WHICH THE SOLUTION IS
C               TO BE CALCULATED (FOR NONLINEAR CASE ONLY)
C     NSBC......NUMBER OF SPECIFIED BOUNDARY CONDITIONS (ESS. B.C.)
```

```
C       NSBF......NUMBER OF SPECIFIED SOURCE VALUES (NATURAL B.C.)
C       NTIME.....NUMBER OF TIME STEPS FOR THE UNSTEADY CASE
C       NX,NY,NZ..NUMBER OF ELEMENTS ALONG THE X,Y AND Z DIRECTIONS
C                 (TO GENERATE THE MESH)
C       NRMAX.....ROW DIMENSION OF GSTIF IN THE DIMENSION STATEMENT
C       R.........CONSTANT IN THE EQUATION OF STATE
C       RENLDS....REYNALDS NUMBER
C       STIF......ELEMENT COEFFICIENT MATRIX
C       THETA.....PARAMETER IN THE TIME APPROXIMATION
C       TLR.......ERROR TOLERANCE BETWEEN THE SOLUTIONS OF TWO CONSECU-
C                 TIVE ITERATIONS (FOR CONVERGENCE OF NONLIN. SOL.)
C       V0........ELEMENT DEGREES OF FREEDOM AT THE PREVIOUS TIME
C       V.........ELEMENT DEGREES OF FREEDOM AT THE CURRENT ITERATION
C       VSBC......ARRAY OF THE VALUES OF SPECIFIED DEGREES OF FREEDOM
C       VSBF......VALUES OF THE SPECIFIED FLUXES
C       X,Y.......THE X AND Y COORDINATES OF GLOBAL NODES IN THE MESH
C
C
C    ................................................................
C
C             D I M E N S I O N   R E Q U I R E M E N T S
C
C       THE DIMENSIONS OF ARRAYS  'GSTIF',  'GF', ' GC', 'GP', 'NOD',
C       'ISBC', 'VSBC', 'DRE',  'X',  'Y', 'DX'  AND  'DY'  SHOULD BE
C       SUCH THAT THEY MEET THE REQUIREMENTS OF PROBLEM BEING SOLVED.
C       THE DIMENSION REQUIREMENTS OF THE ARRAYS ARE AS FOLLOWS:
C
C         GSTIF(NRMAX,NCMAX), GF(NRMAX,5) GC(NRMAX,5), GP(NRMAX,5)
C                  NRMAX.GE.NEQ     AND    NCMAX.GE.NBW
C                  NOD(N,I) WITH  N.GE.NEM  AND  I.EQ.8
C              ISBC(I,J), VSBC(I) WITH I.GE.NSBC AND J.EQ.2
C              ISBF(I,J), VSBF(I) WITH I.GE.NSBF AND J.EQ.2
C         DRE(I) WITH I.GE.NRENLD;  X(I), Y(I) AND Z(I) WITH I.GE.NNM
C           DX(I) WITH I.GE.IEL*NX+1; DY(J) WITH J.GE.IEL*NY+1, ETC.
C    ...............................................................
C
C                 S U B R O U T I N E S   U S E D
C
C       MESH3D....GENERATES NX BY NY BY NZ MESH FOR PRISMATIC DOMAINS
C       BDUNSM....SOLVES UNSYMMETRIC SYSTEM OF BANDED EQUATIONS
C       INVDET....COMPUTES THE INVERSE OF JACOBIAN & ITS DETERMINANT
C       MATMLT....GIVES THE PRODUCT OF TWO MATRICES
C       SHP3D.....EVALUATES THE SHAPE FUNCTIONS AND THEIR DERIVATIVES
C       STF3D.....CALCULATES THE ELEMENT STIFFNESS MATRIX
C       STRS3D....CALCULATES STRESSES AND PRESSURE IN EACH ELEMENT
C
C
C    ................................................................
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION GSTIF(500,200),GF(500,5),GP(500,5),GC(500,5),TITLE(20),
     *          ISBC(160,2),VSBC(160),ISBF(100,2),VSBF(100),IBDS(5),
     *          DRE(10),IBDF(5)
      COMMON/STF/ELXYZ(8,3),STIF(8,8),ELF(8),V(8,5),V0(8,5)
      COMMON/MSH/X(360),Y(360),Z(360),NOD(224,8),DX(10),DY(10),DZ(10)
      DATA NRMAX,NCMAX,NDF/500,200,5/
C
C    ---------------------------------------------------------------
C          P R E P R O C E S S O R   O F   T H E   P R O G R A M
C    ---------------------------------------------------------------
C
C             I N P U T   D A T A   T O   T H E   P R O G R A M
```

```
C
      READ 600, TITLE
      READ 610, AMU,CV,R
      READ 620, IEL,NPE,IMESH,ITEM,ITMAX,NPRNT,NOSTRS
      IF(IMESH.EQ.1)GOTO 30
C
C     IF  THE  DOMAIN  IS  NONRECTANGULAR,  READ THE MESH  INFORMATION
C
      READ 620, NEM,NNM
      DO 20 N=1,NEM
   20 READ 620, (NOD(N,I),I=1,NPE)
      READ 610, (X(I),Y(I),Z(I),I=1,NNM)
      GOTO 40
C
C     IF THE DOMAIN IS RECTANGULAR, READ THE NUMBER OF ELEMENTS  AND
C     COORDINATES OF THE NODES ALONG THE COORDINATE LINES (WHICH ARE
C     ASSUMED TO BE PARALLEL TO THE SIDES OF THE RECTANGULAR DOMAIN)
C
   30 READ 620, NX,NY,NZ
      NX1=IEL*NX+1
      NY1=IEL*NY+1
      NZ1=IEL*NZ+1
      READ 610, (DX(I),I=1,NX1)
      READ 610, (DY(I),I=1,NY1)
      READ 610, (DZ(I),I=1,NZ1)
      READ 620, (NOD(1,I),I=1,NPE)
      CALL MESH3D(NX,NY,NZ,NPE,NNM,NEM)
C
   40 NEQ=NNM
C
C     READ THE NUMBER OF SPECIFIED VELOCITIES (NSBC), THE NODE NUMBERS
C     (ISBC(I,1)) AND THE DEGREE OF FREEDOM (ISBC(I,2))  SPECIFIED  AT
C     THE NODE  AND  THE SPECIFIED  VALUES  (VSBC)  OF THE VELOCITIES.
C     SIMILARLY, READ THE SPECIFIED BOUNDARY 'FORCES' AT THE NODES AND
C     SPECIFIED BOUNDARY TEMPERATURES IN  CASE OF  COUPLED CONVECTION.
C
      READ 620, NSBC
      READ 620, ((ISBC(I,J),J=1,2),I=1,NSBC)
      READ 610, (VSBC(I),I=1,NSBC)
      READ 620, NSBF
      IF(NSBF.EQ.0)GOTO 45
      READ 620, ((ISBF(I,J),J=1,2),I=1,NSBF)
      READ 610, (VSBF(I),I=1,NSBF)
C
C     FOR NONLINEAR ANALYSIS (I.E., THE SOLUTION OF  THE NAVIER-STOKES
C     EQUATIONS),READ THE NUMBER OF INCREMENTS OF THE  REYNALDS NUMBER
C     (NRENLD),  THE ARRAY OF THE INCREMENTS  (DRE(I)),  THE ALLOWABLE
C     PERCENTAGE OF ERROR (TLR)  BETWEEN  THE VELOCITY VECTORS  OF TWO
C     CONSECUTIVE ITERATIONS,  AND THE  ACCELERATION PARAMETER (BETA).
C
   45 NRENLD=1
      IF(ITMAX.LE.1)GOTO 50
      READ 620, NRENLD
      READ 610, (DRE(I),I=1,NRENLD)
      READ 610, TLR,BETA
C
C     READ THE NUMBER OF TIME STEPS (NTIME),  THE TIME STEP (DT)   AND
C     THE PARAMETER (THETA)  IN THE  TIME-APPROXIMATION  FOR  UNSTEADY
C     CASE.    ZERO  INITIAL CONDITIONS  ON  THE VELOCITY ARE ASSUMED.
C
```

```
   50 IF(ITEM.EQ.0)GOTO 60
      READ 620, NTIME
      READ 610, DT,THETA,EPS
   60 CONTINUE
C
C
C     E N D   O F   T H E   I N P U T   D A T A   T O   P R O G R A M
C
C     COMPUTE THE HALF BAND WIDTH (NHBW) OF GLOBAL  COEFFICIENT MATRIX
C
      NHBW=0
      DO 70 N=1,NEM
      DO 70 I=1,NPE
      DO 70 J=1,NPE
      NW=(IABS(NOD(N,I)-NOD(N,J))+1)
   70 IF (NHBW.LT.NW) NHBW=NW
      NBW=2*NHBW
C
C     P R I N T  T H E  I N P U T  D A T A  T O  T H E  P R O G R A M
C
      PRINT 900
      PRINT 600, TITLE
      IF(ITEM.EQ.0)PRINT 870
      IF(ITEM.GE.1)PRINT 880
      IF(ITMAX.GT.1)PRINT 890
      IF(IMESH.EQ.1)PRINT 910, NX,NY,NZ
      PRINT 630, IEL,NDF,NEM,NNM,NBW,NSBC,NSBF
      PRINT 640
      DO 100 I=1,NNM
      DO 80 K=1,NDF
      IBDF(K)=0
   80 IBDS(K)=0
      DO 90 J=1,NSBC
      NODE=ISBC(J,1)
      NBC=ISBC(J,2)
      IF(NODE.NE.I)GOTO 90
      IBDS(NBC)=NBC
   90 CONTINUE
      IF(NSBF.EQ.0)GOTO 100
      DO 95 J=1,NSBF
      NODE=ISBF(J,1)
      NBF=ISBF(J,2)
      IF(NODE.NE.I)GOTO 95
      IBDF(NBF)=NBF
   95 CONTINUE
  100 PRINT 650, I,X(I),Y(I),Z(I),(IBDS(K),K=1,NDF),(IBDF(K),K=1,NDF)
      PRINT 660
      DO 110 I=1,NEM
  110 PRINT 625, I,(NOD(I,J),J=1,NPE)
      IF(ITEM.GE.1)PRINT 920, NTIME,DT,THETA,EPS
      IF(ITMAX.GT.1)PRINT 930, NRENLD,TLR,BETA
C
C        S O M E   B A S I C  C H E C K S   O N   T H E   D A T A
C
      IF(IEL.EQ.0)GOTO 570
      IF(NSBC.EQ.0)GOTO 560
      IF(NEQ.GT.NRMAX)GOTO 580
      IF(NBW.GT.NCMAX)GOTO 590
      IF(IEL.EQ.1 .AND. NPE.NE.8)GOTO 550
C
```

```
C     ------------------------------------------------------------------
C        P R O C E S S O R   U N I T   O F   T H E   P R O G R A M
C     ------------------------------------------------------------------
C
C     L O O P   O N   T H E   R E Y N A L D S   N U M B E R   B E G I N S
C
      NCOUNT=0
      NRE=0
      PRINT 670, RENLDS
  130 NRE=NRE+1
      AMU=1.0/RENLDS
      IF(NRE.GT.NRENLD)GOTO 460
C
C     T I M E   L O O P   F O R   U N S T E A D Y   C A S E   B E G I N S
C
  140 TIME=0.0
      NT=0
C
C     INITIALIZE THE SOLUTION VECTORS: GSTIF(I,NBW), GF(I,5) & GC(I,5)
C
      DO 150 I=1,NEQ
      GSTIF(I,NBW)=0.0
      DO 150 J=1,NDF
      GC(I,J)=0.0
      GP(I,J)=0.0
  150 GF(I,J)=0.0
  160 IF(ITEM.EQ.0)GOTO 175
  170 NT=NT+1
      TIME=TIME+DT
      PRINT 680, TIME
      IF(NT.GT.NTIME)GOTO 450
  175 NCOUNT=NCOUNT+1
      ITER=0
C
C     COUNTER ON  THE ITERATIVE CYCLE  ON THE NONLINEARITY BEGINS HERE
C
  180 ITER=ITER+1
      IF (ITER.GT.ITMAX)GOTO 520
      DO 200 I=1,NEQ
      DO 200 J=1,NBW
  200 GSTIF(I,J)=0.0
C
      DO 350 NDOF=1,NDF
C
      DO 250 I=1,NPE
      DO 250 J=1,NDF
  250 V(I,J)=0.0
C
C     DO-LOOP ON NUMBER OF ELEMENTS BEGINS HERE;  ELEMENT CALCULATIONS
C     AND ASSEMBLY OF ELEMENT MATRICES WILL TAKE PLACE IN THE DO-LOOP.
C     THE ELEMENT COORDINATES,  AND THE CURRENT ITERATION AND PREVIOUS
C     TIME  VELOCITIES  ARE  TRANSFERRED  TO  THE  SUBROUTINE 'STF2D'.
C
      DO 280 N=1,NEM
      DO 220 I=1,NPE
      NI=NOD(N,I)
      ELXYZ(I,1)=X(NI)
      ELXYZ(I,2)=Y(NI)
      ELXYZ(I,3)=Z(NI)
      IF(ITEM.EQ.0)GOTO 210
```

```
      V0(I,NDOF)=GF(NI,NDOF)
  210 IF(ITMAX.LE.1)GOTO 220
      V(I,NDOF)=BETA*GP(NI,NDOF)+(1.0-BETA)*GC(NI,NDOF)
  220 CONTINUE
C
      IT=0
      CALL  FLUX3D(NPE,ELXYZ,V,AMU,IEL,CV,R,ELF)
      CALL  STF3D(IEL,NPE,AMU,IT,THETA,ITEM,DT)
C
C     PRINT (IF NPRNT=1)  THE  ELEMENT  COEFFICIENT  MATRIX: STIF(I,J)
C
      IF(NPRNT.EQ.0)GOTO 240
      IF(N.GT.1)GOTO 240
      IF(ITER.GT.1)GOTO 240
      IF(NTIME.GT.1)GOTO 240
      PRINT 700
      DO 230 I=1,NN
  230 PRINT 610, (STIF(I,J),J=1,NN)
  240 CONTINUE
C
C     ASSEMBLY OF ELEMENT MATRICES INTO GLOBAL MATRICES IN BANDED FORM
C
      DO 260 I=1,NPE
      NR= NOD(N,I)
      GSTIF(NR,NBW)=GSTIF(NR,NBW)+ELF(I)
      DO 260 J=1,NPE
      NCL= NOD(N,J)
      NC=NCL-NR+NHBW+1
      IF (NC) 260,260,250
  250 GSTIF(NR,NC)=GSTIF(NR,NC)+STIF(I,J)
  260 CONTINUE
  280 CONTINUE
C
C     MODIFY THE 'FORCE'  VECTOR TO INCLUDE  SPECIFIED BOUNDARY VALUES
C
      IF(NSBF.EQ.0)GOTO 300
      DO 290 I=1,NSBF
      IF(ISBF(I,2).NE.NDOF)GOTO 290
      NF=ISBF(I,1)
      VF=VSBF(I)
      GSTIF(NF,NBW)=GSTIF(NF,NBW)+VF
  290 CONTINUE
C
C     IMPOSITION OF  THE SPECIFIED BOUNDARY CONDITIONS  ON  VELOCITIES
C
  300 DO 320 I=1,NSBC
      IF(ISBC(I,2).NE.NDOF)GOTO 320
      DO 310 J=1,NBW
  310 GSTIF(IE,J)=0.0
      GSTIF(IE,NHBW)=1.0
      GSTIF(IE,NBW)=VSBC(I)
  320 CONTINUE
C
C
C     SOLUTION OF  THE ASSEMBLED EQUATIONS  FOR THE VELOCITIES USING A
C     BANDED EQUATION SOLVER. THE SOLUTION IS STORED IN 'GSTIF(I,NBW)'
C
      CALL BDUNSM(GSTIF,NRMAX,NCMAX,NEQ,NHBW,IER)
C
      DO 330 I=1,NEQ
```

```
      GP(I,NDOF)=GC(I,NDOF)
  330 GC(I,NDOF)=GSTIF(I,NBW)
  350 CONTINUE
      NFLAG=0
      IF(ITMAX.LE.1)GOTO 420
      IF(NCOUNT.LE.1 .AND. ITER.LE.1)GOTO 420
C
C     CHECK FOR CONVERGENCE OF VELOCITIES FOR A GIVEN  REYNALDS NUMBER
C
      NFLAG=1
      DO 400 NDOF=1,NDF
      ERR=0.0
      DNORM=0.0
      DO 380 I=1,NEQ
      DNORM=DNORM+GC(I)**2
  380 ERR=ERR+(GP(I)-GC(I))**2
      ERROR=DSQRT(ERR/DNORM)
      IF (ERROR.GT.TLR)GOTO 180
  400 PRINT 760, RENLDS,ITER,ERROR
C
C     PRINT THE LINEAR  OR  CONVERGED NONLINEAR SOLUTION FOR EACH NODE
C
  420 PRINT 710
      DO 430 I=1,NNM
  430 PRINT 730, I,GC(I,1),GC(I,2),GC(I,3),GC(I,4),GC(I,5)
C
      IF(ITMAX.LE.1)GOTO 440
      IF(NFLAG.EQ.0)GOTO 180
  440 IF(ITEM.EQ.0)GOTO 460
C
C     CHECK TO SEE IF THE SOLUTION HAS REACHED THE STEADY STATE
C
      DIFFT=0.0
      DNRMT=0.0
      DO 450 I=1,NEQ
      DNRMT=DNRMT+GF(I)**2
      DIFFT=DIFFT+(GF(I)-GC(I))**2
      IF(ICONV.EQ.0)GOTO 450
      IF(I.GT.NNM)GOTO 450
      TF(I)=TC(I)
  450 GF(I)=GC(I)
      DIFF=1.0
      IF(NT.EQ.1)GOTO 460
      DIFF=DSQRT(DIFFT/DNRMT)
C
C     ------------------------------------------------------------
C      P O S T P R O C E S S O R  P A R T  O F  T H E  P R O G R A M
C     ------------------------------------------------------------
C
C     CALL SUBROUTINE 'STRS3D' TO CALCULATE STRESSES AND PRESSURE
C
  460 IF(NOSTRS.EQ.0)GOTO 490
      PRINT 740
      PRINT 750
      PRINT 740
      DO 480 N=1,NEM
      DO 470 I=1,NPE
      NI=NOD(N,I)
      DO 465 J=1,NDF
  465 V(I,J) = GC(NI,J)
```

```
      ELXYZ(I,1)=X(NI)
      ELXYZ(I,2)=Y(NI)
  470 ELXYZ(I,3)=Z(NI)
  480 CALL STRS3D(NPE,ELXYZ,V,AMU,IEL,CV,R)
  490 PRINT 740
C
      IF(ITEM.EQ.0)GOTO 500
      IF(DIFF.LT.EPS)GOTO 530
      GOTO 170
  500 IF(ITMAX.LE.1)GOTO 510
      RENLDS=RENLDS+DRE(NRE)
      IF(NRENLD.GT.1)GOTO 130
  510 CONTINUE
      GOTO 599
  520 PRINT 800, ITER,ERROR
      GOTO 599
  530 PRINT 790, TIME
      GOTO 599
  550 PRINT 860
      GOTO 599
  560 PRINT 850
      GOTO 599
  570 PRINT 840
      GOTO 599
  580 PRINT 810
      PRINT 830
      GOTO 599
  590 PRINT 820
      PRINT 830
  599 STOP
C
C     ----------------------------------------------------------------
C                          F   O   R   M   A   T   S
C     ----------------------------------------------------------------
C
  600 FORMAT (20A4)
  610 FORMAT (8F10.5)
  620 FORMAT (16I5)
  625 FORMAT (I5,3X,8I5)
  630 FORMAT (/,5X,'ELEMENT TYPE (1, LINEAR; 2, QUADRATIC) =',I5,
     *        /,5X,'NUMBER OF DEGREES OF FREEDOM PER NODE..=',I5,
     *        /,5X,'NUMBER OF ELEMENTS IN THE MESH.........=',I5,
     *        /,5X,'NUMBER OF NODES IN THE MESH............=',I5,
     *        /,5X,'FULL BAND WIDTH OF THE ASSEMBLED MATRIX=',I5,
     *        /,5X,'NUMBER OF SPECIFIED PRIMARY UNKNOWNS...=',I5,
     *        /,5X,'NUMBER OF SPECIFIED FLUXES.............=',I5)
  640 FORMAT (/,2X,'NODE',10X,'X',16X,'Y',16X,'Z',10X,'SPEC. VEL.',5X,
     *   'SPEC. FORCES',2X,'SPEC. TEMP.',/,70X,'0 INDICATES UNSPEC.',/)
  650 FORMAT (I5,3(5X,E12.5),3I5,2X,3I5,2X,I5)
  660 FORMAT (/,' ELEM NO.   BOOLEAN (CONNECTIVITY) MATRIX NOD(I,J) ',/)
  670 FORMAT (/,5X,'R E Y N A L D S  N U M B E R =',E12.4)
  680 FORMAT (/,5X,'T I M E =',E12.4,/)
  700 FORMAT (/,5X,'ELEMENT STIFFNESS MATRIX:',/)
  710 FORMAT (/,8X,'NODE',2X,'DENSITY',5X,'X-VELOCITY',5X,'Y-VELOCITY',
     *          5X,'Z-VELOCITY',5X,'TEMPERATURE',/)
  730 FORMAT (5X,I5,5(2X,E13.5))
  740 FORMAT (5X,120('-'))
  750 FORMAT (10X,'X',12X,'Y',12X,'Z',7X,'XX-STRESS',3X,'YY-STRESS',3X,
     *'ZZ-STRESS',3X,'XY-STRESS',3X,'XZ-STRESS',3X,'YZ-STRESS',3X,'PRESS
     *URE')
```

```
  760 FORMAT (/,5X,'R E Y N A L D S   N U M B E R =',E12.4,5X,
     *'R A L E I G H   N U M B E R =',E12.4,/,5X,'ITERATION NO. =',I4,5X,
     *'ERROR =',E12.4)
  790 FORMAT (/,5X,'STEADY-STATE IS REACHED AT TIME =',E12.4)
  800 FORMAT (/,5X,'CONVERGENCE CRITERION IS NOT SATISFIED FOR ITMAX = '
     *,I3,2X,' ERROR = ',E13.5,'***',/)
  810 FORMAT (5X,'**DATA ERROR**: THE ROW-DIMENSION OF GSTIF IS LESS THA
     *N THE NUMBER OF EQUATIONS**')
  820 FORMAT (5X,'**DATA ERROR**: THE COLUMN-DIMENSION OF GSTIF IS LESS
     *THAN THE FULL BANDWIDTH**')
  830 FORMAT (5X,'THE DIMENSIONS OF GSTIF SHOULD MATCH WITH NRMAX  AND
     *NCMAX IN THE DATA STATEMENT',/,5X,'AND NRMAX AND NCMAX SHOULD BE
     *SUCH THAT NRMAX.GE.NEQ AND NCMAX.GE.NBW')
  840 FORMAT (5X,'**INPUT ERROR**: THE VARIABLE IEL IS READ AS ZERO.  IT
     * SHOULD BE EITHER 1 OR 2')
  850 FORMAT (5X,'**INPUT ERROR**: THE VALUE OF NSBC IS EQUAL TO ZERO')
  860 FORMAT (5X,'**INPUT ERROR**: VALUES OF IEL AND NPE DO NOT MATCH.
     *IF IEL=1 THEN NPE=4, AND IF IEL=2 THEN NPE=8 OR 9')
  870 FORMAT (5X,'S T E A D Y - S T A T E   S O L U T I O N')
  880 FORMAT (5X,'U N S T E A D Y   S O L U T I O N')
  890 FORMAT (5X,'N O N L I N E A R   A N A L Y S I S')
  900 FORMAT (/,5X, 75('-'),/,15X,'P R O G R A M   FLOPEN3D  (DEVELOPED
     *BY J. N. REDDY)',/,5X,75('-'),//)
  910 FORMAT (/,5X,'FINITE ELEMENT MESH (NX BY NY BY NZ)...=',3I5)
  920 FORMAT (/,5X,'NUMBER OF TIME STEPS (NTIME)...........=',I5,/,
     *          5X,'VALUE OF THE TIME STEP (DT)............=',E12.4,/,
     *          5X,'TIME PARAMETER (THETA).................=',E12.4,/,
     *          5X,'TOLERANCE FOR STEADY STATE (EPS).......=',E12.4,/)
  930 FORMAT (/,5X,'NUMBER OF  *RE* OR *RA*   INCREMENTS...=',I5,/,
     *          5X,'CONVERGENCE TOLERANCE (TLR)............=',E12.4,/,
     *          5X,'ACCELERATION PARAMETER (BETA)..........=',E12.4,/)
      END
      SUBROUTINE STF3D(IEL,NPE,AMU,IT,THETA,ITEM,DT)
C
C     ..................................................................
C     PROGRAM 'STF3D' GENERATES ELEMENT COEFFICIENT MATRIX 'STIF'  AND
C     SOURCE VECTOR  'ELF'  FOR THE LINEAR  (EIGHT-NODE) ISOPARAMETRIC
C     PRISM ELEMENT.
C     ..................................................................
C
C        GAUSS.....ARRAY OF GAUSS POINTS
C        GDSF......GLOBAL DERIVATIVES OF THE SHAPE FUNCTIONS
C                  GDSF(I,J)=DERIVATIVE OF SF(J) W.R.TO X(I)
C        WT........ARRAY OF WEIGHTS CORRESPONDING TO GAUSS POINTS
C        SF........ELEMENT SHAPE FUNCTIONS
C        STIF......ELEMENT COEFFICIENT MATRICES
C     ..................................................................
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION GAUSS(4,4),WT(4,4),S(8,8)
      COMMON/STF/ELXYZ(8,3),STIF(8,8),ELF(8),V(8,5),V0(8,5)
      COMMON/SHP/SF(8),GDSF(3,8)
C
      DATA GAUSS/4*0.0D0,-.57735027D0,.57735027D0,2*0.0D0,-.77459667D0,
     *0.0D0,.77459667D0,0.0D0,-0.86113631D0,-0.33998104D0,0.33998104D0,
     *0.86113631D0/
      DATA WT/2.0D0,3*0.0D0,2*1.0D0,2*0.0D0,0.55555555D0,0.88888888D0,
     *0.55555555D0,0.0D0,0.34785485D0,2*0.65214515D0,0.34785485D0/
C
      NGP=IEL+1
```

```
      NGP1=IEL
C
C     INITIALIZE THE ARRAYS, SX, SY, ETC. (FOR PENALTY TERMS)
C
      DO 50 I = 1,NPE
      ELF(I) = 0.0
      DO 50 J = 1,NPE
      S(I,J)=0.0
   50 STIF(I,J)=0.0
      IF(IT.EQ.1) GO TO 145
C
C     COMPUTE THE COEFFICIENT MATRICES FOR EACH VARIABLE
C
  145 DO 300 NI = 1, NGP
      DO 300 NJ = 1, NGP
      DO 300 NK = 1, NGP
      XI = GAUSS(NI,NGP)
      ETA = GAUSS(NJ,NGP)
      ZETA = GAUSS(NK,NGP)
      CALL SHP3D(NPE,DET,XI,ETA,ZETA,ELXYZ)
      CONST=DET*WT(NI,NGP)*WT(NJ,NGP)*WT(NK,NGP)
      RHO=0.0
      T=0.0
      V1 = 0.0
      V2 = 0.0
      V3 = 0.0
      DIV = 0.0
      DO 150 I=1,NPE
      RHO=RHO+V(I,1)*SF(I)
      T=T+V(I,5)*SF(I)
      V1 = V1+V(I,2)*SF(I)
      V2 = V2+V(I,3)*SF(I)
      V3 = V3+V(I,4)*SF(I)
  150 DIV=DIV+V(I,2)*GDSF(1,I)+V(I,3)*GDSF(2,I)+V(I,4)*GDSF(3,I)
      DO 200 I = 1, NPE
  160 DO 200 J = 1, NPE
      STIF(I,J)=STIF(I,J)+((V1*GDSF(1,J)+V2*GDSF(2,J)+V3*GDSF(3,J))*
     *          SF(I)+SF(I)*SF(J)*DIV)*CONST
      IF(ITEM.EQ.0)GOTO 200
      S(I,J)=S(I,J)+SF(I)*SF(J)*CONST
  200 CONTINUE
  300 CONTINUE
      IF(ITEM.EQ.0)RETURN
      THETA1=1.0-THETA
      DO 600 I=1,NN
      DO 600 J=1,NN
      ELF(I)=ELF(I)+(S(I,J)-DT*THETA*STIF(I,J))*V0(J)
  600 STIF(I,J)=S(I,J)+DT*THETA*STIF(I,J)
      RETURN
      END
      SUBROUTINE FLUX3D(NPE,ELXYZ,V,AMU,IEL,CV,R,ELF,NDOF)
C
C     ..................................................................
C     PROGRAM 'FLUX3D' COMPUTES TOTAL STRESSES, PRESSURE AND DIVERGENCE
C     OF THE VELOCITY FIELD  AND THE FLUX VECTORS FOR EACH EQUATION.
C             P = PRESSURE AT THE CURRENT GAUSS POINT
C           DPX = DERIVATIVE OF P WITH RESPECT TO X
C           DPY = DERIVATIVE OF P WITH RESPECT TO Y
C           DPZ = DERIVATIVE OF P WITH RESPECT TO Z
C          DIVQ = DIVERGENCE OF THE FLUX
```

```
C         DISPN = DISSIPATION, SIGMA:STRAIN RATE
C     ..............................................................
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ELXYZ(8,3),V(8,5),GAUSS(4,4)
      COMMON/SHP/SF(8),GDSF(3,8)
      DATA GAUSS/4*0.0D0,-.57735027D0,.57735027D0,2*0.0D0,-.77459667D0,
     *0.0D0,.77459667D0,0.0D0,-0.86113631D0,-0.33998104D0,.33998104D0,
     *0.86113631D0/
C
      NGP=IEL
      DO 100 II=1,NGP
      DO 100 JJ=1,NGP
      DO 100 KK=1,NGP
      XI=GAUSS(II,NGP)
      ETA=GAUSS(JJ,NGP)
      ZETA=GAUSS(KK,NGP)
      CALL SHP3D(NPE,DET,XI,ETA,ZETA,ELXYZ)
      X = 0.0
      Y = 0.0
      Z = 0.0
      SUM1 = 0.0
      SUM2 = 0.0
      SUM3 = 0.0
      SUM12=0.0
      SUM13=0.0
      SUM23=0.0
      RHO=0.0
C
C     CALCULATE STRAIN-RATES
C
      DO 40 I=1,NPE
      X = X + ELXYZ(I,1)*SF(I)
      Y = Y + ELXYZ(I,2)*SF(I)
      Z = Z + ELXYZ(I,3)*SF(I)
      RHO=RHO+W(I,1)*SF(I)
      SUM1=SUM1+W(I,2)*GDSF(1,I)
      SUM2=SUM2+W(I,3)*GDSF(2,I)
      SUM3=SUM3+W(I,4)*GDSF(3,I)
      SUM12=SUM12+W(I,2)*GDSF(2,I)+W(I,3)*GDSF(1,I)
      SUM13=SUM13+W(I,2)*GDSF(3,I)+W(I,4)*GDSF(1,I)
   40 SUM23=SUM23+W(I,3)*GDSF(3,I)+W(I,4)*GDSF(2,I)
C
C     CALCULATE STRESSES AND PRESSURE
C
C     USE PROPER CONSTITUTIVE LAWS TO COMPUTE THE TEMPERATURE AND
C     PRESSURE
C
      CALL STATE(XI,ETA,P,DPX,DPY,DPZ,DIVQ,V,CA,R)
C
      SIGMA1 = -P+AMU*(4.0*SUM1-2.0*(SUM2+SUM3))/3.0
      SIGMA2 = -P+AMU*(4.0*SUM2-2.0*(SUM1+SUM3))/3.0
      SIGMA3 = -P+AMU*(4.0*SUM3-2.0*(SUM1+SUM2))/3.0
      SGMA12=AMU*SUM12
      SGMA13=AMU*SUM13
      SGMA23=AMU*SUM23
      IF(NDOF.EQ.2)ELF(I)=ELF(I)+SF(I)*(-DPX+SGMA1X+SGM12Y+SGMA13Z)*
     *              CONST
      IF(NDOF.EQ.3)ELF(I)=ELF(I)+SF(I)*(-DPY+SGMA12X+SGMA2Y+SGMA23Z)*
     *              CONST
```

```
      IF(NDOF.EQ.4)ELF(I)=ELF(I)+SF(I)*(-DPZ+SGMA13X+SGMA23Y+SGMA3Z)*
     *           CONST
      IF(NDOF.EQ.5)ELF(I)=ELF(I)+SF(I)*(-DIVQ+DISPN)*CONST
   50 CONTINUE
  100 CONTINUE
  200 FORMAT (5X,10E12.4)
      RETURN
      END
      SUBROUTINE SHP3D(NPE,DET,XI,ETA,ZETA,ELXYZ)
C
C     ..................................................................
C     PROGRAM 'SHP3D' EVALUATES  SHAPE FUNCTIONS AND THEIR DERIVATIVES
C     AT THE GUASSIAN POINTS OF THE EIGHT-NODE ISOPARAMETRIC PRISMATIC
C     ELEMENT.  SHAPE FUNCTIONS FOR HIGHER-ORDER ELEMENTS CAN BE ADDED
C     ..................................................................
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION XNODE(8,3),DSF(3,8),ELXYZ(8,3),GJ(3,3),GJINV(3,3)
      COMMON/SHP/SF(8),GDSF(3,8)
      DATA XNODE/2*1.0D0,2*-1.0D0,2*1.0D0,3*-1.0D0,2*1.0D0,
     *   2*-1.0D0,2*1.0D0,5*-1.0D0,4*1.0D0/
C
      FNC(A,B,C) = 0.125*A*B*C
      DO 50 I = 1, NPE
      XP=XNODE(I,1)
      YP=XNODE(I,2)
      ZP = XNODE(I,3)
      XIO = 1.0 + XI*XP
      ETAO = 1.0 + ETA*YP
      ZETAO = 1.0 + ZETA*ZP
      SF(I) = FNC(XIO,ETAO,ZETAO)
      DSF(1,I) = FNC(XP,ETAO,ZETAO)
      DSF(2,I) = FNC(XIO,YP,ZETAO)
      DSF(3,I) = FNC(XIO,ETAO,ZP)
   50 CONTINUE
      CALL MATMLT(DSF,3,NPE,ELXYZ,3,GJ)
      CALL INVDET(GJ,GJINV,DET)
      CALL MATMLT(GJINV,3,3,DSF,NPE,GDSF)
      RETURN
      END
      SUBROUTINE STRS3D(NPE,ELXYZ,W,AMU,IEL,CV,R)
C
C     ..................................................................
C     PROGRAM 'STRS3D' COMPUTES TOTAL STRESSES, PRESSURE AND DIVERGENCE
C     OF THE VELOCITY FIELD IN EACH ELEMENT (TO CHECK CONSERV. OF MASS)
C     ..................................................................
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ELXYZ(8,3),W(8,5),GAUSS(4,4)
      COMMON/SHP/SF(8),GDSF(3,8)
      DATA GAUSS/4*0.0D0,-.57735027D0,.57735027D0,2*0.0D0,-.77459667D0,
     *0.0D0,.77459667D0,0.0D0,-0.86113631D0,-0.33998104D0,.33998104D0,
     *0.86113631D0/
C
      NGP=IEL
      DO 50 II=1,NGP
      DO 50 JJ=1,NGP
      DO 50 KK=1,NGP
      XI=GAUSS(II,NGP)
      ETA=GAUSS(JJ,NGP)
```

```
      ZETA=GAUSS(KK,NGP)
      CALL SHP3D(NPE,DET,XI,ETA,ZETA,ELXYZ)
      X = 0.0
      Y = 0.0
      Z = 0.0
      SUM1 = 0.0
      SUM2 = 0.0
      SUM3 = 0.0
      SUM12=0.0
      SUM13=0.0
      SUM23=0.0
      RHO=0.0
C
C     CALCULATE STRAIN-RATES
C
      DO 40 I=1,NPE
      X = X + ELXYZ(I,1)*SF(I)
      Y = Y + ELXYZ(I,2)*SF(I)
      Z = Z + ELXYZ(I,3)*SF(I)
      RHO=RHO+W(I,1)*SF(I)
      SUM1=SUM1+W(I,2)*GDSF(1,I)
      SUM2=SUM2+W(I,3)*GDSF(2,I)
      SUM3=SUM3+W(I,4)*GDSF(3,I)
      SUM12=SUM12+W(I,2)*GDSF(2,I)+W(I,3)*GDSF(1,I)
      SUM13=SUM13+W(I,2)*GDSF(3,I)+W(I,4)*GDSF(1,I)
   40 SUM23=SUM23+W(I,3)*GDSF(3,I)+W(I,4)*GDSF(2,I)
C
C     CALCULATE STRESSES AND PRESSURE
C
C     USE PROPER CONSTITUTIVE LAWS TO COMPUTE THE TEMPERATURE AND
C     PRESSURE
C
      T = CV*RHO
      P = RHO*R*T
      SIGMA1 = -P+AMU*(4.0*SUM1-2.0*(SUM2+SUM3))/3.0
      SIGMA2 = -P+AMU*(4.0*SUM2-2.0*(SUM1+SUM3))/3.0
      SIGMA3 = -P+AMU*(4.0*SUM3-2.0*(SUM1+SUM2))/3.0
      SGMA12=AMU*SUM12
      SGMA13=AMU*SUM13
      SGMA23=AMU*SUM23
      PRINT 200,X,Y,Z,SIGMA1,SIGMA2,SIGMA3,SGMA12,SGMA13,SGMA23,P
   50 CONTINUE
  100 CONTINUE
  200 FORMAT (5X,10E12.4)
      RETURN
      END
      SUBROUTINE INVDET (A,B,DET)
C
C     ..................................................................
C     PROGRAM 'INVDET' COMPUTES THE INVERSE OF THE JACOBIAN MATRIX AND
C     DETERMINANT  OF  THE JACOBIAN MATRIX FOR  THE EIGHT-NODE ELEMENT
C     ..................................................................
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(3,3),B(3,3)
      G(Z1,Z2,Z3,Z4)=Z1*Z2-Z3*Z4
      F(Z1,Z2,Z3,Z4)=G(Z1,Z2,Z3,Z4)/DET
      C1=G(A(2,2),A(3,3),A(2,3),A(3,2))
      C2=G(A(2,3),A(3,1),A(2,1),A(3,3))
      C3=G(A(2,1),A(3,2),A(2,2),A(3,1))
```

```
      DET=A(1,1)*C1+A(1,2)*C2+A(1,3)*C3
      B(1,1)=F(A(2,2),A(3,3),A(3,2),A(2,3))
      B(1,2)=-F(A(1,2),A(3,3),A(1,3),A(3,2))
      B(1,3)=F(A(1,2),A(2,3),A(1,3),A(2,2))
      B(2,1)=-F(A(2,1),A(3,3),A(2,3),A(3,1))
      B(2,2)=F(A(1,1),A(3,3),A(3,1),A(1,3))
      B(2,3)=-F(A(1,1),A(2,3),A(1,3),A(2,1))
      B(3,1)=F(A(2,1),A(3,2),A(3,1),A(2,2))
      B(3,2)=-F(A(1,1),A(3,2),A(1,2),A(3,1))
      B(3,3)=F(A(1,1),A(2,2),A(2,1),A(1,2))
      RETURN
      END
      SUBROUTINE MATMLT (A,M,N,B,L,C)
C
C     ..............................................................
C     PROGRAM 'MATMLT' GIVES THE PRODUCT OF TWO MATRICES:    [C]=[A][B]
C         THE PROGRAM MULTIPLIES A(M,N) BY B(N,L) TO GIVE C(M,L)
C     ..............................................................
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(M,N),B(N,L),C(M,L)
      DO 10 I=1,M
      DO 10 J=1,L
      C(I,J)=0.
      DO 10 K=1,N
   10 C(I,J)=C(I,J)+A(I,K)*B(K,J)
      RETURN
      END
      SUBROUTINE MESH3D(NEX,NEY,NEZ,NPE,NNM,NEM)
C
C     ..............................................................
C     PROGRAM  'MESH3D'  GENERATES MESH FOR  RECTANGULAR DOMAINS USING
C     THE EIGHT-NODE (LINEAR) PRISMATIC ELEMENTS. THE PROGRAM COMPUTES
C     THE COORDINATES OF GLOBAL NODES X,Y, AND Z, BOOLEAN CONNECTIVITY
C     MATRIX 'NOD' RELATING ELEMENT NODES TO GLOBAL NODES OF THE MESH.
C     ..............................................................
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/MSH/X(360),Y(360),Z(360),NOD(224,8),DX(10),DY(10),DZ(10)
      NPEH=NPE/2
      NEXY=NEX*NEY
      NEM = NEXY*NEZ
      NPX=NEX+1
      NPY=NEY+1
      NPZ=NEZ+1
      NNM = NPX*NPY*NPZ
C
C     GENERATE THE CONNECTIVITY MATRIX, 'NOD'
C
      N=1
      N1=NPX
      N2=NPX
      N3=NPX*NPY
      NN = N3*NPZ
      KK = 1
      IF (NEX .EQ. 1)GOTO 30
      DO 20 I=2,NEX
      N=N+1
      DO 10 K=1,NPE
   10 NOD(I,K)=NOD(I-1,K)+KK
```

```
   20 CONTINUE
   30 IF(NEY .EQ. 1)GOTO 50
      DO 40 J=2,NEY
      DO 40 I=1,NEX
      N=N+1
      M=N-NEX
      DO 40 K=1,NPE
      NPIY=N2
      IF(K .EQ. 0)NPIY = N1
      NOD(N,K)=NOD(M,K)+NPIY
   40 CONTINUE
   50 IF(NEZ .EQ. 1)GOTO 70
      DO 60 L=2,NEZ
      DO 60 K=1,NEXY
      N=N+1
      M=N-NEXY
      DO 60 I=1,NPE
   60 NOD(N,I)=NOD(M,I)+N3
   70 CONTINUE
C
C     COMPUTE THE COORDINATES X, Y, AND Z  OF THE GLOBAL NODES
C
      DO 80 K=1,NPZ
      DO 80 J=1,NPY
      DO 80 I=1,NPX
      N=1+(I-1)*KK+(J-1)*N2+(K-1)*N3
      X(N)=DX(I)
      Y(N)=DY(J)
   80 Z(N)=DZ(K)
C
      RETURN
      END
      SUBROUTINE BDUNSM(A,NRMAX,NCMAX,N,ITERM,IER)
C
C     ..................................................................
C     PROGRAM 'BDUNSM' IS AN  EQUATION  SOLVER FOR  BANDED UNSYMMETRIC
C     SET OF EQUATIONS. THE SOLUTION IS STORED IN THE LAST COLUMN OF A
C     ..................................................................
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(NRMAX,NCMAX)
      CERO=1.D-10
      PARE=CERO**2
      NBND=2*ITERM
      NBM=NBND-1
C
C     BEGINS ELIMINATION OF THE LOWER LEFT
C
      DO 80 I=1,N
      IF (DABS(A(I,ITERM))  .LT.  CERO)  GO TO 10
      GO TO 20
   10 IF(DABS(A(I,ITERM)).LT.PARE) GO TO 110
   20 JLAST=MIN0(I+ITERM-1,N)
      L=ITERM+1
      DO 40 J=I,JLAST
      L=L-1
      IF (DABS(A(J,L))  .LT.  PARE)  GO TO 40
      B=A(J,L)
      DO 30 K=L,NBND
   30 A(J,K)=A(J,K)/B
```

```
      IF (I.EQ.N) GO TO 90
   40 CONTINUE
      L=0
      JFIRST=I+1
      IF (JLAST.LE.I) GO TO 80
      DO 70 J=JFIRST,JLAST
      L=L+1
      JL=J-L
      ITL=ITERM-L
      IF (DABS(A(J,ITL)) .LT. PARE) GO TO 70
      DO 50 K=ITERM,NBM
      KL=K-L
   50 A(J,KL) = A(JL,K) - A(J,KL)
      A(J,NBND)= A(JL,NBND)-A(J,NBND)
      IF (I.GE.N-ITERM+1) GO TO 70
      DO 60 K=1,L
      NBK=NBND-K
   60 A(J,NBK) = - A(J,NBK)
   70 CONTINUE
   80 CONTINUE
   90 L=ITERM-1
      DO 100 I=2,N
      N1I=N+1-I
      DO 100 J=1,L
      N1IJ=N1I+J
      ITJ=ITERM+J
      IF (N+1-I+J.GT.N) GO TO 100
      A(N1I,NBND) = A(N1I,NBND) - A(N1IJ,NBND)*A(N1I,ITJ)
  100 CONTINUE
      IER=0
      RETURN
  110 PRINT 130
      PRINT 120, I,A(I,ITERM)
      IER=1
      RETURN
  120 FORMAT (10X,'THE EQUATION NUMBER AND THE COEFFICIENT =',I5,E13.4)
  130 FORMAT (5X,'COMPUTATION STOPPED IN BDUNSM BECAUSE ZERO APPEARED ON
     * THE MAIN DIAGONAL.   *** CHECK YOUR MATRIX ***')
      END
```

THE UNIVERSITY OF TENNESSEE SPACE INSTITUTE

CONTRACT NO.: NASA NAS8-36555
UT ACCT NO.: R02-400450

PRINCIPAL INVEST: DR. K.C. REDDY
EFFECTIVE DATE: 03/13/86 - 03/11/87

MONTH &YEAR	PERSONNEL SERVICES	FRINGE BENEFITS	TRAVEL	COMMUNI-CATIONS	COMPUTER USAGE	SUPPLIES & OTHER	EQUIPMENT	OVERHEAD	TOTAL EXPDITURES	CHARGES OUTSDING	BUDGET ADDITIONS	FREE BALANCE
BUDGET	$23,930	$1,337	$4,000	$400	$2,000	$0	$3,000	$15,327	$49,994		$49,994	
BAL FWRD	$8,156	$1,442	$833	$0	$0	$34	$0	$5,390	$15,855			$34,139
AUG 86	$1,867	$276	$0	$0	$0	$0	$0	$1,104	$3,247			
TO DATE	$10,023	$1,718	$833	$0	$0	$34	$0	$6,493	$19,102			$30,892
SEPT 86	$1,287	$74	$0	$0	$0	$0	$0	$701	$2,063			
TO DATE	$11,310	$1,793	$833	$0	$0	$34	$0	$7,195	$21,164			$28,830
OCT 86	$1,890	$251	$48	$0	$0	$0	$0	$1,127	$3,316			
TO DATE	$13,200	$2,044	$880	$0	$0	$34	$0	$8,322	$24,480			$25,514