1

National Library
of Canada

Bibliothèque nationale
du Canada

Canadian Theses Service

Services des thèses canadiennes

Ottawa, Canada
K1A 0N4

CANADIAN THESES

THÈSES CANADIENNES

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

National Library
of CanadaBibliothèque nationale
du Canada

Canadian Theses Division

Division des thèses canadiennes

Ottawa, Canada
K1A 0N4**PERMISSION TO MICROFILM — AUTORISATION DE MICROFILMER**

• Please print or type — Ecrire en lettres moulees ou dactylographier

Full Name of Author — Nom complet de l'auteur

GURURAJA RAO VENAGANTI

Date of Birth — Date de naissance

21 / 07 / 57

Country of Birth — Lieu de naissance

INDIA

Permanent Address — Residence fixe

3900, GOSNOLD AVE
NORFOLK Va 23508
USA

Title of Thesis — Titre de la these

FINITE ELEMENT SOLUTION OF
THE NAVIER-STOKES EQUATIONS

University — Université

CARLETON UNIVERSITY

Degree for which thesis was presented — Grade pour lequel cette these fut présentée

M Engg

Year this degree conferred — Année d'obtention de ce grade

1984

Name of Supervisor — Nom du directeur de these

Dr S. SJOLANDER

Permission is hereby granted to the NATIONAL LIBRARY OF
CANADA to microfilm this thesis and to lend or sell copies of
the filmThe author reserves other publication rights, and neither the
thesis nor extensive extracts from it may be printed or other-
wise reproduced without the author's written permission.L'autorisation est, par la présente, accordée à la BIBLIOTHÈ-
QUE NATIONALE DU CANADA de microfilmer cette thèse et de
prêter ou de vendre des exemplaires du filmL'auteur se réserve les autres droits de publication, ni la thèse
ni de longs extraits de celle-ci ne doivent être imprimés ou
autrement reproduits sans l'autorisation écrite de l'auteur

Date

10 / 08 / 84

Signature

Finite Element Solution of the Navier-Stokes Equations

by

© G.R.Vemaganti, M.Tech(Mech.)

*A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the
Degree of Master of Engineering*

DEPARTMENT OF MECHANICAL AND AERONAUTICAL ENGINEERING
CARLETON UNIVERSITY
OTTAWA, ONTARIO
CANADA.

July 1984

The undersigned recommend to the Faculty of Graduate Studies acceptance of the thesis 'Finite Element Solution of the Navier-Stokes Equations' submitted by Gururaja Rao Vemaganti, M.Tech.(Mech.) in partial fulfilment of the requirements for the degree of Master of Aeronautical Engineering.

Thesis Supervisor

Chairman

Department of Mechanical
and Aeronautical Engineering

ABSTRACT

The application of the finite element method in the field of fluid mechanics using the primitive variables is reviewed. A finite element algorithm for solving Navier-Stokes equations governing steady viscous flow is formulated based on the Galerkin's weighted residual method. A computer program is developed as a part of the ASGARD finite element package to be used to solve fluid flow problems. Different features associated with this program such as the solver, convergence criterion etc. are discussed. The package is tested on several problems to demonstrate its capabilities. The results are presented. At present the program can be applied only to laminar flow problems and its three-dimensional capabilities are yet to be demonstrated. The program documentation along with sample input and output files are presented as appendices.

ACKNOWLEDGEMENTS

I would like to extend my deepest appreciation to my thesis supervisors Dr. Steen A. Sjolander and Dr. John A. Goldak for their continued interest, encouragement and guidance provided during this project. Their patient cooperation and understanding nature are gratefully acknowledged. I would like to thank the chairman, Dr. H.I.H. Saravanamuttoo and the Department of Mechanical and Aeronautical Engineering, Carleton University for their financial support which helped me carry-out this research.

I wish to thank Dr. Neal M. Holtz who helped me a number of times in word-processing. I would like to thank all my friends at Carleton University for their help and cooperation, with a special mention to Mr.B.Patel and Mr.J.C. Bose for their assistance in developing the software.

Finally I thank my God Sri Raghavendra Swami who always helps me in all my pursuits in life.

Table of Contents

Abstract	iii
Acknowledgements	vi
Table of Contents	v
List of Figures and Tables	vii
Nomenclature	ix
1 INTRODUCTION	1
1.1 Finite Elements versus Finite Differences	2
1.2 Scope of This Project	2
2 FINITE ELEMENT METHOD	4
2.1 Galerkin's Weighted Residual Approach	5
2.2 Velocity-Pressure Formulation	6
2.3 Previous Work with V-P Formulation	7
2.4 Choice of the Element	9
2.5 Structure of a FEM Program	11
3 FINITE ELEMENT FORMULATION	13
3.1 Governing Equations	14
3.2 Initial and Boundary Conditions	16
3.4 Finite Element Equations	17
3.3.1 Subroutine VFL3D20	20
4 MAIN FEATURES OF THE PROGRAM	22
4.1 The Solver	22
4.1.1 Program Details	24
4.3 Convergence Criterion	25

5	TEST PROBLEMS AND RESULTS	28
5.1	Test Problem Requirements	29
5.1.1	Plane-Poiseuille Flow	29
5.1.2	Couette Flow	30
5.1.3	Flow Down an Inclined Plane	30
5.1.4	Flow Between Porous Plates	31
5.2	Fully Developed Laminar and Turbulent Flow	32
5.3	Further Observations	34
6	CONCLUSIONS	36
6.1	Future Work	37
	References	39
	Figures and Tables	43
	Appendix I	69
	Appendix II	76
	Appendix III	78
	Appendix IV	84

List of Figures and Tables

Figures

2.1 One-dimensional elements and associated hierarchical shape functions and variables of (a) linear, (b) quadratic, (c) cubic form. Ref. to Zienkiewicz and Morgan(1988)	43
2.2 Unit hierarchical brick element	44
2.3 Flow diagram for the ASGARD package for fluid mechanics	45
3.1 Initial and boundary conditions for the problem of flow over a circular cylinder	46
5.1 Velocity profile for the flow between parallel plates with one element in the FEM mesh	47
5.2 Non-dimensional velocity profile for the flow between parallel plates with two elements in the FEM mesh	48
5.3a Non-dimensional velocity profile for Couette flow with negative pressure gradient	49
5.3b Non-dimensional velocity profile for Couette flow with zero pressure gradient	50
5.3c Non-dimensional velocity profile for Couette flow with positive pressure gradient	51
5.4 Geometrical representation of the problem of flow down an inclined plane	52
5.5 Non-dimensional velocity profile for the flow down an inclined plane	53

5.6 Non-dimensional velocity profile for the flow between porous plates, with two elements in the FEM mesh, at a wall Reynolds number of 2	54
5.7 A three element mesh for the flow between porous plates problem	55
5.8a Non-dimensional velocity profile for the flow between porous plates, at wall $Re=2$, with a three element mesh	56
5.8b Non-dimensional velocity profile for the flow between porous plates, at wall $Re=3$, with a three element mesh	57
5.8c Non-dimensional velocity profile for the flow between porous plates, at wall $Re=5$, with a three element mesh	58
5.9 A three element mesh for the flow through a circular pipe	59
5.10 Non-dimensional velocity profile for the laminar pipe flow	60
 Tables	
2.1 Previous Work With V-P Formulation	62
2.2 Shape Functions For The Hierarchical Brick Element	63
5.1 Summary of The Test Cases	64

NOMENCLATURE

a	A constant
A	Element stiffness matrix
B	Body force vector
C	Nodal value of a variable ϕ
f	A constant in a differential equation
F	Body force per unit volume of the fluid, $Kg/m^2 - sec^2$
g	Acceleration due to gravity
h	Half the distance between the parallel plates
k	Karman constant
K	A coefficient in the stiffness matrix
l	Mixing length
L	A differential operator
m	Number of corner nodes in an element
M	Shape function at a corner node in an element
n	Total number of nodes in an element
N	Shape function at any node in an element
p	Pressure, $Kg/m - sec^2$
R	Residual obtained in the Galerkin's method
Re	Reynolds number
u	'x' component of velocity, m/sec
U	Velocity of the moving plate in a Couette flow

v	'y' component of velocity, m/sec
V	Volume, m^3
w	'z' component of velocity, m/sec
W	Weighting function or a weighting factor
x	Coordinate in 'x' direction
y	Coordinate in 'y' direction
z	Coordinate in 'z' direction

Greek Symbols

ϕ	A variable
ξ	A local coordinate
τ	Stress tensor
μ	Dynamic viscosity of a fluid, $Kg/m - sec$
ρ	Density of a fluid, Kg/m^3
δ	Kronecker delta
Δ	Deviation in the velocity
ν	Kinematic viscosity of the fluid, m^2/sec
λ	The degree of freedom vector
ϵ	Tolerance limit for convergence
θ	Angle of inclination

Subscripts

e	Effective
i	An index for a tensor or a node

j Another index for a tensor

n The n^{th} iteration

x In the 'x' direction

y In the 'y' direction

z In the 'z' direction

0,1 etc Reference points

w Location at the wall

t Turbulent flow

Superscript

e Element

Chapter 1

INTRODUCTION

The objective of this research is to develop an engineering tool for analyzing the flow of viscous fluids in complex internal geometries. Such a tool could be used to design minimum length contractions, to predict both the static pressure recovery and the total pressure loss in diffusers, and to analyze the blade-to-blade and tip leakage flows in turbomachinery.

A variety of approaches can and have been used to analyze the flows cited. In the case of the contraction a combined inviscid/boundary layer calculation is feasible since the boundary layers remain thin; to allow for non-uniform inlet flow a rotational inviscid calculation could be applied to the mainstream. On the other hand, the validity of using the boundary layer assumptions in the diffuser flow is more doubtful since the viscous wall layers grow rapidly along the length and may eventually meet and interact on the centreline. Likewise, the complex and highly interactive flows in turbomachinery tend not to lend themselves to the use of boundary-layer techniques. Thus, to have a tool with maximum generality and flexibility it would seem desirable to solve essentially the full Navier-Stokes equations.

As to the numerical method for solving the equations, there are again several possibilities. For example, Moore and Moore (1979) have predicted a variety of complex flows using the parabolized Navier-Stokes equations and finite-difference techniques. However, since a large body of programs for the finite-element method (ASGARD) has already been developed at Carleton, it was decided instead to extend these programs to the calculation of fluid flows. In this way it will be possible to draw on much existing technology and thus reduce the overall development time and effort. In any case, the finite-element method (FEM) seems particularly suitable for handling the complicated geometries which will arise in internal flows of interest here.

1.1 Finite Elements vs Finite Differences

It is a well known fact that the method of finite differences has long been a driving force in the field of computational fluid dynamics. The finite difference method (FDM) still dominates computational fluid dynamics and has been quite successful in many respects. The question naturally arises of where the finite element method stands relative to the finite difference method. There are many features in the finite element method which are analogous to those of the finite difference method but there exist certain advantages and disadvantages of one method over the other.

An effective comparison of relative computational aspects between FEM and FDM is not a simple matter. For a rigorous comparison, we may choose a level of error and compute with as fine a mesh as necessary to obtain that error. Then the relative efficiency can be determined by comparing the computing time and the amount of work necessary to solve the same problem. Though this work was not undertaken in this project, other researchers such as Chung (1981) have carried-out a similar analysis. They concluded as follows:

- 1) Mathematical error estimates are more rigorously handled in FEM than FDM.
- 2) More complex geometries are better interpreted in a FEM mesh than in the case of a uniform finite difference mesh
- 3) The computational work required to obtain the same level of error by FEM and FDM varies depending on problems and various schemes employed in FEM and FDM.

Like any other numerical technique, FEM too suffers from certain disadvantages which will be discussed later on in this thesis in the last chapter on conclusions.

1.2 Scope of This Project

It should be emphasized that the objective is to develop a useful engineering tool rather than to advance the state of the art in the finite element calculation of fluid flows. Therefore, whenever possible an effort will be made to draw on the successful (or unsuccessful) experience of others working in the field: for example, in the method of formulation and in the choice of the solver.

For the time being, only steady, incompressible flows will be considered. Initially, the formulation for turbulent flows is based on the mixing length/eddy viscosity approach since Moore and Moore (1979) have demonstrated that this can produce results of acceptable accuracy. Development of the turbulence model required for generating an eddy-viscosity distribution in the flow would be an area of future investigation.

It is necessary to test any program on some simple test problems to demonstrate its capabilities. This project is limited to testing the package on some simple test cases so as to establish its validity. The selection of the test problems is based on their simplicity, fewer number of elements required for the mesh and the availability of the exact solutions. Once the program is established for simpler problems it can be used for more complicated problems with confidence.

Chapter 2

FINITE ELEMENT METHOD

The finite element method is an approximate method of solving differential equations of boundary and/or initial value problems in engineering. In this method, a continuum is divided into many small cells called 'elements' of convenient shapes and sizes linked through suitable points called 'nodes'. The variable in the differential equation is written as a linear combination of appropriately selected interpolation functions and the values of the variable or its various derivatives specified at its nodes. Using variational principles or weighted residual methods, the governing differential equations are transformed into finite element equations governing all the elements. These local elements are finally assembled together to form a global system of differential or algebraic equations with the proper boundary and/or initial conditions imposed. The nodal values of the variable are determined from this system of equations.

The close relationship of finite element analysis to the classical variational concept of the Rayleigh-Ritz method or the weighted residual methods modelled after the well-known method of Galerkin, has established the finite element method as a versatile tool for solving any set of differential equations. Unfortunately, variational principles often cannot be found for some engineering problems, particularly when the differential equations are not self-adjoint, that is when they give rise to non-symmetric matrix equations. Weighted residuals are applied in the methods of Galerkin, least squares or collocation. Zienkiewicz and Morgan (1983) have pointed out that the finite element Galerkin formulation always gives results which are at least as accurate as those produced by the equivalent finite difference equations. In applying the FEM to fluid mechanics, the Galerkin method is often considered the most convenient way for formulating finite element models since it requires no variational principles. The least squares method requires higher order interpolation functions in general even if the physical behaviour may be adequately described by

linear or lower order functions. For these reasons, Galerkin's method is employed here for solving fluid flow problems using finite elements.

2.1 Galerkin's Weighted Residual Approach

The method of weighted residuals is a technique for obtaining approximate solutions to linear and nonlinear partial differential equations. Applying the method of weighted residuals involves basically two steps. The first step is to assume the general functional behaviour of the dependent field variable in some way so as to approximately satisfy the given differential equation and the boundary conditions. Substitution of this approximation into the original differential equation and the boundary conditions then results in some error called a residual. The main aim of the weighted residual method is to minimize this residual. This is done in the second stage of the process, while solving the equations resulting from the first step by choosing a particular function for the general functional form, which then becomes the approximate solution sought.

If a field variable ϕ is governed by the differential equation

$$L(\phi) - f = 0 \quad (1)$$

in the domain ω then an approximate solution for this variable for an element in the domain could be written as

$$\bar{\phi} = \sum_{i=1}^n N_i C_i \quad (2)$$

where N are a set of trial functions written in terms of local coordinates associated with n discrete values within or on the boundary of an element and C_i are the values of the variable at the nodes.

The residual then becomes

$$R = L\left(\sum_{i=1}^n N_i C_i\right) - f \quad (3)$$

Minimizing the residual over the entire domain yields the Galerkin's FEM equation i.e.

$$\int_{\omega} \left(W_k \left(L \sum_{i=1}^n N_i C_i \right) - f \right) d\omega = 0 \quad k = 1, 2, \dots, n \quad (4)$$

where W_k are called the weighting functions. In the Bubnov-Galerkin method, the trial functions are chosen as the weighting functions which give the following equation.

$$\int_{\omega} N_k \left(L \left(\sum_{i=1}^n N_i C_i \right) - f \right) d\omega = 0 \quad (5)$$

The steady or transient flow of an incompressible viscous fluid is governed by the well-known Navier-Stokes equations and the continuity equation. In most of the cases of practical interest the Navier-Stokes equations exhibit elliptic nature. In other words the solution of such equations is both an initial and a boundary value problem.

In developing the FEM approximation for the Navier-Stokes equations there are mainly three different approaches.

These are the use of

- i) the primitive variables, u, v, w and p
- ii) the stream function and vorticity, and
- iii) the stream function alone.

Out of these three, the primitive variable formulation using the velocities and the pressures is most suited for the Galerkin's method. Its advantages over other approaches and its applicability in different flow problems is discussed in the next section.

2.2 Velocity-Pressure Formulation

The stream function approach offers the advantage that only one fourth order differential equation for the stream function ξ need be solved and the plate-bending analogy in stress analysis can be applied, but it suffers from the disadvantage that the element interpolation function needs a higher order compatibility and this is more complicated in its construction. Those elements achieving or approximating C^1 continuity, that is those elements which have the variable function and its first derivative continuous at the inter-element boundaries, must be employed. It is not possible to specify the boundary conditions in terms of the normal and tangential derivatives of ξ for C^0 elements where the variable function alone is continuous at the inter-element boundaries.

The primitive variables (Velocity-Pressure) formulation is the only one that is available for a three dimensional algorithm, since a stream function does not exist for a three dimensional flow. Specification of the boundary conditions also is much simpler in the case of the V-P formulation. As the primitive variables of the flow field are obtained as a solution, post processing is a simple task. It is also easier to relate to the problem physically with this method.

A significant difficulty arising in all finite element formulations of the primitive equations is the interaction of the pressure and the velocity fields. The difficulty arises from the fact that the pressure is coupled through the continuity equation which is merely a constraint on the divergence of velocities rather than a full third equation coupling the pressure to the velocities. As a result, it is difficult to develop solution methods which are capable of determining the primitive variables simultaneously, with the same degree of accuracy.

In formulating the primitive variables model, the critical point is the choice of the interpolations for velocity and pressure. Hood and Taylor (1973) noted that the interpolation for velocity should be one order higher than that for the pressure so that a consistently accurate result would then be obtained simultaneously for both velocity and pressure. The explanation of the choice was based on error consistency of the two coupled equations, momentum and continuity, for the two unknowns, u and p . The work done by the previous researchers using mixed interpolation is discussed in the next section.

2.3 Previous Work with V-P Formulation

In their two-dimensional formulation Taylor and Hood (1973) used eight-noded quadrilaterals using quadratic interpolation for the velocity and a linear interpolation for the pressure. They used the same integration schemes for velocity and the pressure terms in the stiffness matrix. Yamada et al (1975) used quadratic triangular elements for the velocity components and linear triangular elements with the same vertex nodes for the pressures. They were successful in obtaining the exact velocity profiles in a shear and pressure induced flow between parallel plates. Kawahara et al (1976), Nickell et al (1974), King et al (1975) used the same mixed interpolation, namely quadratic

for the velocity components and a linear variation for the pressures. Some examples of the different types of elements used by other researchers and their applications are given in the Table 2.1.

Nickell and his coworkers (1974) used the six noded triangles for solving free surface flow problems. They found that the solutions to the Poiseuille flow were giving errors of the order of 10^{-6} for the primitive variables compared with the exact solution. Their basic finite element was a 20 degree-of-freedom triangle. King et al (1975) also used a quadratic approximation for the velocities and linear variation for the pressures. They made this selection after considering the implication of the different order of the continuity equation. They were successful in predicting the two dimensional flow over a submerged weir and a flood flow of a river through a constriction using the same six noded, quadratic velocity and linear pressure elements.

In a few attempts at applying the FEM to flow problems utilizing the primitive equations made by researchers up to now, the utilization of the integrated method with mixed interpolation of pressure one order less than the velocity was the most successful. The main mixed interpolation schemes considered so far were linear velocity-constant pressure scheme and quadratic velocity-linear pressure interpolation schemes. Among these two, the quadratic velocity-linear pressure scheme gave the best results when compared with either the exact solutions or the approximate solutions obtained by the classical finite difference method. Olson and Tuann (1976) demonstrated that using mixed interpolation is in fact essential. They proved that equal interpolation always generated one or more eigen vectors which contained only pressure and corresponded to zero eigenvalues; they claimed that these were spurious and were the cause of the failure. Sani et al (1981) concluded that a two dimensional 8 noded serendipity velocity element, that is the element whose shape functions are computed directly, using only the exterior nodes with C^0 bilinear pressure exhibits no spurious modes. These were some of the observations made by the other researchers which we considered before starting this project.

2.4 Choice of the Element

Based on the success achieved by previous researchers it has been decided that the mixed interpolation, that is a quadratic variation for the velocity and a linear variation for the pressure, should be adopted for the solution of the Navier-Stokes equations.

A simple three dimensional element for this mixed interpolation could be a 20 noded brick. The main disadvantage with such an element is the extra computational cost associated with it in calculating eight more shape functions and their twenty four derivatives for the linear interpolation of the pressures. In other words we will be calculating the shape functions and their derivatives for two elements separately namely, a 20 noded brick and an 8 noded brick. To avoid the use of two elements of different interpolations, Goldak (1983) suggested the use of a 20 noded hierarchical brick element for solving the Navier-Stokes equations. A unit brick element of this type is shown in the Fig.2.2. Zienkiewicz et al (1983) discuss about the hierarchical finite elements, their advantages in the formulation and their application in the field of stress analysis. This element is still isoparametric and in a global coordinate system any curved boundary can be interpreted using this element. In a local coordinate system the coordinates vary from -1 to 1 between corner nodes. While solving a problem globally the coordinates can be specified in a cartesian coordinate system with any values given to them. In a hierarchical element for our purposes the absolute velocities are specified at the corner nodes and the deviations from the linear variation of the velocities are specified at the midedge nodes. Shape functions are computed in such a way that they are linear at the corner nodes and parabolic at the midedge nodes. The general expression for a velocity u could be written in terms of the shape functions and its nodal values as follows

$$u = \sum_{i=1}^8 N_i u_i + \sum_{i=1}^{12} M_i \Delta u_i \quad (6)$$

where u_i are the absolute velocities at the eight corner nodes and Δu_i are the deviations from the velocities at the midedge nodes. N_i are the linear shape functions at the corner nodes and M_i are the shape functions at the midedge nodes. In a similar way the expression for the pressure could be written as

$$p = \sum_{i=1}^8 N_i p_i \quad (7)$$

The additional advantages for such an element include ease in introducing higher order approximation, refinement of the mesh with fewer elements and better numerical conditioning because of flexibility to incorporate higher order interpolation.

The hierarchical finite elements are defined as those in which successive refinements are added in a similar way to the additional terms in a Fourier series. It follows that the stiffness matrix corresponding to the hierarchical element at a certain level of refinement is a submatrix of the 'stiffness' matrix corresponding to a higher order of refinement. As the shape functions for each order of approximation are not completely different, each level of approximation does not result in a completely new element matrix and thus the equation set need not be entirely reevaluated, if it is decided to resolve a problem using shape functions of a higher degree.

This fact can be demonstrated easily using a one dimensional element. A linear interpolation of a variable ϕ could be achieved with just two nodes 0 and 1 as follows

$$\phi^e = \phi_0 N_0^e + \phi_1 N_1^e \quad (8)$$

where ϕ_0 and ϕ_1 are the nodal values of the variable ϕ and N_0 , N_1 are the corresponding shape functions. A hierarchical format over this element could be achieved using a quadratic to modify this linear variation.

Introducing an intermediate node 2 between 0 and 1, which has a quadratic shape function in the local element coordinate ξ given by

$$N_2^e = \alpha_0 + \alpha_1 \xi + \alpha_2 \xi^2 \quad (9)$$

This expression takes the form

$$N_2^e = 1 - \xi^2 \quad \text{when} \quad \alpha_0 = 1, \alpha_1 = 0, \alpha_2 = -1 \quad (9a)$$

still the C^0 continuity of the approximation ϕ between the elements is preserved, provided N_2^e is zero at $\xi = \pm 1$. Now the quadratic approximation over an element 'e' could be written as

$$\phi^e = \phi_0 N_0^e + \phi_1 N_1^e + \phi_2 N_2^e \quad (10)$$

where $a_2^s N_2^s$ is the deviation from the linear approximation. In a similar manner, for a cubic interpolation one more term $a_3^s N_3^s$ has to be added to the quadratic expansion. This could be done easily by introducing another node 3 with a shape function of degree three. Zienkiewicz and Morgan(1983) have given the one dimensional hierarchical elements with different shape functions for higher order approximation of a variable as shown in Fig.2.1. The 20 noded brick used in this project is the three dimensional analogue of the one dimensional quadratic hierarchical element. A unit hierarchical brick in local coordinates with its nodes is shown in Fig.2.2. The hierarchical shape functions for this element are given in Table 2.2.

2.5 Structure of a FEM Program

The general structure of a FEM program consists of a main subroutine which calls the other subroutines in a sequence. The main routine also does the looping over each element for the assembly and loops over each iteration to check for the convergence. Most of the major FEM packages consist of powerful diagnostic subroutines which check the control data at different stages of processing.

The key role in any FEM program is played by the subprogram which calculates the element level stiffness matrix and the element level load vector. The program also calls a subroutine which computes the shape functions and their derivatives of a particular type of element used for that problem. In a general case, only this part of the package has to be replaced for different types of problems, where a general FEM program is already available.

Once the stiffness matrix coefficients are computed for all the elements in the solution domain, these values are input into the solver subroutine, which essentially assembles all the elements in the domain, thus forming a global stiffness matrix, and solves them for the global variables. These values are compared with some tolerance limit and reiterated. Once the results have converged, they are sent to a postprocessor to convert the raw output into a set of presentable desired results. The flow-chart of the package used for solving the 3-D Navier-Stokes equations is given in Fig 2.3.

The VFL3D20 subroutine used by the ASGARD program to compute the element level stiffness matrix, is written for a 20 noded brick element.

It forms the three momentum equations for each of the twenty nodes and the continuity equation for the eight corner nodes. The complete set of the governing equations and formulation of the element stiffness matrix is discussed in the next chapter.

Chapter 3

FINITE ELEMENT FORMULATION

In this chapter the basic governing equations of an incompressible fluid, the Navier-Stokes equations and the continuity equation are presented. The finite element discretization of the governing equations is done using the Galerkin-Bubnov approach. The Green's theorem was employed to reduce the second order derivatives to first order. The element stiffness matrix is formulated and all of its coefficients discussed. Finally the details of a Fortran subroutine to compute this element stiffness matrix are discussed.

3.1 Governing Equations

The governing equations for a steady 3-D, incompressible viscous flow are the full form of the Navier-Stokes equations including the turbulent Reynolds stress terms (Reynolds equations) and the continuity equation.

The equations could be written in an indicial notation as follows:

$$u_j \frac{\partial u_i}{\partial x_j} = F_i + \frac{1}{\rho} \frac{\partial}{\partial x_j} (\tau_{ij}) \quad (1)$$

And

$$\frac{\partial u_i}{\partial x_i} = 0 \quad (2)$$

where u is the velocity, F is the body force acting on the fluid, ρ is the density of the fluid and τ is the stress tensor. It is to be noted that equation (1) represents the time averaged momentum equations. In these equations the transport of momentum by turbulent motion is represented by correlations between the fluctuating quantities. The complete set of stress components for a

Newtonian fluid can be written as

$$\tau_{i,j} = -p\delta_{i,j} + \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \overline{\rho u_i u_j} \quad (3)$$

where p is the pressure, $\delta_{i,j}$ is the Kronecker delta ($\delta_{i,j} = 0$ for $i \neq j$, $\delta_{i,j} = 1$ for $i = j$). The second term corresponds to viscous stresses, μ being the absolute viscosity of the fluid and the last term corresponds to the Reynolds stresses consisting of fluctuating velocity components. Because of these Reynolds stress terms the mean flow equations are not closed, that is there are more unknowns in the governing equations than the number of equations available, and a turbulence model is required to evaluate these turbulent transport terms. Rodi (1982) describes several available models for calculating the turbulent stresses for incompressible flows. One of the first turbulence models proposed was Prandtl's (1925) mixing length hypothesis which is simpler and is one of the most widely used models. This hypothesis based on the eddy-viscosity concept, relates the Reynolds stresses to the local mean velocity gradients as follows:

$$-\overline{\rho u_i u_j} = \mu_e \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \quad (4)$$

where μ_e is the 'eddy viscosity'. The total stress may now be written as

$$\tau_{i,j} = -p\delta_{i,j} + \mu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \quad (5)$$

where μ_t is the effective or total viscosity, given by

$$\mu_t = \mu_e + \mu \quad (6)$$

μ_e may be assumed to be isotropic for a three-dimensional flow, though there is not enough experimental evidence in the literature to make this assumption. For a two-dimensional boundary layer, Prandtl's theory gives an expression for the turbulent viscosity as given by White (1974) and is given below

$$\mu_e = \rho l^2 \frac{\partial u}{\partial y} \quad (7)$$

where ' l ' is the mixing length whose distribution over the flow field has to be prescribed with the aid of empirical information. Again, there are several eddy-viscosity models available in the

literature which correlate the mixing length with other parameters. Galbraith et al (1976) have shown from experimental evidence that their model of eddy-viscosity is valid over a wide range of flow conditions. Their model was an improvement over the well known Cebeci and Smith (1974) model of eddy-viscosity. They correlated the mixing length l with the other parameters of the flow in the wall region of a boundary layer as follows:

$$l = ky \sqrt{\frac{\tau}{\tau_w}} \quad (8)$$

where k is the Korman constant, y is the normal distance from the wall, τ is the shear stress, the subscript w represents the location at the wall. They also modified Cebeci and Smith's (1974) model for the outer region of the boundary layer. This model could be our best choice to obtain closure for turbulent flows in our project.

The eddy viscosity hypothesis has been tested for and applied to a large variety of problems, especially those related to the boundary layer flows. Rodi (1982) also gives the limitations of this hypothesis such as its zero eddy-viscosity prediction where the velocity gradient is zero and its problems in predicting the turbulence effects due to buoyancy, rotation or streamline curvature. Although it is a very successful model for two dimensional flows, its validity for three dimensional flows is yet to be fully established. But because of its simplicity and fewer number of governing equations required to be solved, the eddy-viscosity model was chosen for modelling turbulence in our project.

With the inclusion of this 'eddy viscosity' concept, equation (1) yields the commonly used form for a turbulent flow

$$u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho} \frac{\partial p}{\partial x_j} \delta_{i,j} + \frac{\partial}{\partial x_j} \left(\nu_t \frac{\partial u_i}{\partial x_j} \right) + \frac{\partial u_j}{\partial x_i} \frac{\partial \nu_t}{\partial x_j} + F_i \quad (10)$$

The term $\frac{\partial \nu_t}{\partial x_i} \frac{\partial^2 u_j}{\partial x_j \partial x_i}$ goes to zero since $\frac{\partial u_i}{\partial x_j} = 0$ from continuity equation of an incompressible fluid. Since the three momentum equations are non-linear in nature an iterative procedure must be used to solve them. For convenience the velocity components are represented by u , v and w in the future discussion.

3.2 Initial and Boundary Conditions

The initial conditions may be assumed to be zero for a large range of problems. However, considerable savings in computer time can be made if reasonable values can be input to initiate the necessarily iterative procedure. Initial conditions do not influence a solution much for linear problems. For nonlinear problems a good initial guess is a must for faster convergence. A possible choice may be feeding the results of an inviscid solution as the initial condition for the viscous flow problem. The prescription of the boundary conditions becomes more difficult for an inviscid flow since the no-slip boundary condition does not apply at the solid wall and only normal component of velocity has to be made zero at the wall. The boundary conditions may be natural, where for the Navier-Stokes equations spatial derivatives of the variables are specified, or essential, where specific values are allocated to specified variables. Taylor and Hughes(1981) succeeded in solving the problem of two dimensional flow past a circular cylinder by imposing the proper boundary conditions as shown in Fig.3.1. Only one half of the flow field is considered because of symmetry of the problem. Specification of the boundary conditions is the major area where a clear perspective has yet to be obtained. Questions on the minimum number of boundary conditions required, and the appropriate set of boundary conditions on a specific problem are yet to be answered. A third type of boundary condition, the traction specification, that is specifying shear force on a boundary was suggested by Taylor et al (1981). The traction specification is found to be very effective for certain applications where the primitive variables are not available. These are not, however, as widely prescribed as the aforementioned boundary conditions and hence eliminated from the present analysis. Prescription of the essential boundary conditions alone need not make the problem fully posed in certain cases, for example in a problem where acceleration in a particular direction is to be made zero. The element has to be modified for this purpose, as discussed in section 5.3.

3.3 Finite Element Equations

Variation of the primitive variables is given in terms of shape functions of an isoparametric element in the following way. In the following notation the subscript i corresponds to the nodal value of a variable.

$$u = \sum_{i=1}^{i=n} N_i u_i \quad (9)$$

$$v = \sum_{i=1}^{i=n} N_i v_i \quad (11)$$

$$w = \sum_{i=1}^{i=n} N_i w_i \quad (12)$$

$$p = \sum_{i=1}^{i=m} M_i p_i \quad (13)$$

Where N_i correspond to the shape functions of all the 20 nodes and M_i correspond to the 8 corner nodes of the element. Applying the Galerkin's weighted residual approach to the set of governing equations (2,8), they will be transformed into the following finite element equations. As in the case of a hierarchical element the deviations in velocities act as the degrees of freedom at the midside nodes, the formulation is essentially similar to that of an ordinary element and is given as follows:

$$\begin{aligned} \sum_1^n \int_V N_i \left[\sum_1^n N_k u_k \sum_1^n \frac{\partial N_j}{\partial x} u_j + \sum_1^n N_k v_k \sum_1^n \frac{\partial N_j}{\partial y} u_j + \sum_1^n N_k w_k \sum_1^n \frac{\partial N_j}{\partial z} u_j \right. \\ \left. + \frac{1}{\rho} \sum_1^m \frac{\partial M_j}{\partial x} p_j - B_x - \nu_t \left(\sum_1^n \frac{\partial^2 N_j}{\partial x^2} u_j + \sum_1^n \frac{\partial^2 N_j}{\partial y^2} u_j + \sum_1^n \frac{\partial^2 N_j}{\partial z^2} u_j \right) \right. \\ \left. - \left(\sum_1^n \frac{\partial N_j}{\partial x} \frac{\partial \nu_t}{\partial x} u_j + \sum_1^n \frac{\partial N_j}{\partial y} \frac{\partial \nu_t}{\partial y} u_j + \sum_1^n \frac{\partial N_j}{\partial z} \frac{\partial \nu_t}{\partial z} u_j \right) \right. \\ \left. - \left(\sum_1^n \frac{\partial N_j}{\partial x} \frac{\partial \nu_t}{\partial x} u_j + \sum_1^n \frac{\partial N_j}{\partial x} \frac{\partial \nu_t}{\partial y} v_j + \sum_1^n \frac{\partial N_j}{\partial x} \frac{\partial \nu_t}{\partial z} w_j \right) \right] dV = 0 \end{aligned} \quad (14)$$

This equation represents the discretized form of the momentum equation in the X direction, for the entire domain. In this equation the summation sign outside the integral refers to the summation over all the elements. N_i in the beginning of the equation is a weighting function. For convenience, the shape functions are used as the weighting functions. $\sum_1^n N_k u_k \sum_1^n \frac{\partial N_j}{\partial x} u_j$ corresponds to $u \frac{\partial u}{\partial x}$. The summation $\sum_1^n N_k u_k$ represents the multiplying velocity and is obtained from the results of

the previous iteration. The value of u to be solved for in the current iteration only appears in the expression for $\frac{\partial u}{\partial x}$. This is necessary since otherwise the equation would be non-linear in the unknown u . For our chosen element, $n = 20$ and $m = 8$. The other two momentum equations in y and z directions are discretized in a similar way and are given below.

$$\begin{aligned} \sum_1^n \int_{V^e} N_i \left[\sum_1^n N_k u_k \sum_1^n \frac{\partial N_j}{\partial x} v_j + \sum_1^n N_k v_k \sum_1^n \frac{\partial N_j}{\partial y} v_j + \sum_1^n N_k w_k \sum_1^n \frac{\partial N_j}{\partial z} v_j \right. \\ \left. + \frac{1}{\rho} \sum_1^n \frac{\partial M_j}{\partial y} p_j - B_y - \nu_t \left(\sum_1^n \frac{\partial^2 N_j}{\partial x^2} v_j + \sum_1^n \frac{\partial^2 N_j}{\partial y^2} v_j + \sum_1^n \frac{\partial^2 N_j}{\partial z^2} v_j \right) \right. \\ \left. - \left(\sum_1^n \frac{\partial N_j}{\partial x} \frac{\partial \nu_t}{\partial x} v_j + \sum_1^n \frac{\partial N_j}{\partial y} \frac{\partial \nu_t}{\partial y} v_j + \sum_1^n \frac{\partial N_j}{\partial z} \frac{\partial \nu_t}{\partial z} v_j \right) \right. \\ \left. - \left(\sum_1^n \frac{\partial N_j}{\partial y} \frac{\partial \nu_t}{\partial x} u_j + \sum_1^n \frac{\partial N_j}{\partial y} \frac{\partial \nu_t}{\partial y} v_j + \sum_1^n \frac{\partial N_j}{\partial y} \frac{\partial \nu_t}{\partial z} w_j \right) \right] dV^e = 0 \end{aligned} \quad (15)$$

$$\begin{aligned} \sum_1^n \int_{V^e} N_i \left[\sum_1^n N_k u_k \sum_1^n \frac{\partial N_j}{\partial x} w_j + \sum_1^n N_k v_k \sum_1^n \frac{\partial N_j}{\partial y} w_j + \sum_1^n N_k w_k \sum_1^n \frac{\partial N_j}{\partial z} w_j \right. \\ \left. + \frac{1}{\rho} \sum_1^n \frac{\partial M_j}{\partial z} p_j - B_z - \nu_t \left(\sum_1^n \frac{\partial^2 N_j}{\partial x^2} w_j + \sum_1^n \frac{\partial^2 N_j}{\partial y^2} w_j + \sum_1^n \frac{\partial^2 N_j}{\partial z^2} w_j \right) \right. \\ \left. - \left(\sum_1^n \frac{\partial N_j}{\partial x} \frac{\partial \nu_t}{\partial x} w_j + \sum_1^n \frac{\partial N_j}{\partial y} \frac{\partial \nu_t}{\partial y} w_j + \sum_1^n \frac{\partial N_j}{\partial z} \frac{\partial \nu_t}{\partial z} w_j \right) \right. \\ \left. - \left(\sum_1^n \frac{\partial N_j}{\partial z} \frac{\partial \nu_t}{\partial x} u_j + \sum_1^n \frac{\partial N_j}{\partial z} \frac{\partial \nu_t}{\partial y} v_j + \sum_1^n \frac{\partial N_j}{\partial z} \frac{\partial \nu_t}{\partial z} w_j \right) \right] dV^e = 0 \end{aligned} \quad (16)$$

The continuity equation can be written as follows

$$\sum_1^n \int_{V^e} M_i \left[\sum_1^n \frac{\partial N_j}{\partial x} u_j + \sum_1^n \frac{\partial N_j}{\partial y} v_j + \sum_1^n \frac{\partial N_j}{\partial z} w_j \right] dV^e = 0 \quad (17)$$

Invoking Green's theorem the terms containing the second order derivatives in equations 14, 15 and 16 can be reduced to first order derivatives in the following way. For example the diffusion terms in equation (14) can be expanded as follows:

$$\begin{aligned} \nu_t \int_{V^e} N_i \left[\sum_1^n \frac{\partial^2 N_j}{\partial x^2} u_j + \sum_1^n \frac{\partial^2 N_j}{\partial y^2} u_j + \sum_1^n \frac{\partial^2 N_j}{\partial z^2} u_j \right] dV^e = \nu_t \int_{\Gamma^e} N_i \sum_1^n \frac{\partial N_j}{\partial n} u_j ds \\ - \nu_t \int_{V^e} \left(\frac{\partial N_i}{\partial x} \sum_1^n \frac{\partial N_j}{\partial x} u_j + \frac{\partial N_i}{\partial y} \sum_1^n \frac{\partial N_j}{\partial y} u_j + \frac{\partial N_i}{\partial z} \sum_1^n \frac{\partial N_j}{\partial z} u_j \right) dV^e \end{aligned}$$

Γ_e in the above expression represents the surface area over which the integral has to be evaluated. The surface integrals in the above expansion are eliminated from the formulation because when they are summed in adjacent elements, their net contribution is zero. They are significant only when a boundary acts as a limit to the domain under consideration.

The assembled matrix equations take the form

$$A\lambda = B \quad (18)$$

where A is the element stiffness matrix, λ is the degree of freedom vector and B is the external force vector. where λ is given by

$$\lambda_i = \begin{pmatrix} u_i \\ v_i \\ w_i \\ p_i \end{pmatrix} \quad (19)$$

Each submatrix in the global stiffness matrix A has the form,

$$a_{i,j} = \sum_1^{n_e} \int_{V^e} \begin{pmatrix} K_{1,1} & K_{1,2} & K_{1,3} & K_{1,4} \\ K_{2,1} & K_{2,2} & K_{2,3} & K_{2,4} \\ K_{3,1} & K_{3,2} & K_{3,3} & K_{3,4} \\ K_{4,1} & K_{4,2} & K_{4,3} & K_{4,4} \end{pmatrix} dV^e \quad (20)$$

and B is the body force vector given by

$$B_i = \sum_1^{n_e} \int_{V^e} \begin{pmatrix} B_{x,i} \\ B_{y,i} \\ B_{z,i} \\ 0 \end{pmatrix} dV^e \quad (21)$$

where, $B_{x,i} = N_i B_x$, $B_{y,i} = N_i B_y$ and $B_{z,i} = N_i B_z$. The coefficients of the stiffness matrix can be expressed as follows, where, as in the above equations, i is the number of the row in the global stiffness matrix and j is the column. If entry (i,j) corresponds to a $K_{1,1}$ of a submatrix then

$$K_{1,1} = +N_i \sum_1^n N_k u_k \frac{\partial N_j}{\partial x} + N_i \sum_1^n N_k v_k \frac{\partial N_j}{\partial y} + N_i \sum_1^n N_k w_k \frac{\partial N_j}{\partial z} + \nu_t \left(\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right) - \left(2N_i \frac{\partial N_j}{\partial x} \frac{\partial \nu_t}{\partial z} + N_i \frac{\partial N_j}{\partial y} \frac{\partial \nu_t}{\partial y} + N_i \frac{\partial N_j}{\partial z} \frac{\partial \nu_t}{\partial z} \right)$$

and similarly

$$\begin{aligned}
 K_{1,2} &= -N_i \frac{\partial N_j}{\partial x} \frac{\partial \nu_i}{\partial y} ; \quad K_{1,3} = -N_i \frac{\partial N_j}{\partial x} \frac{\partial \nu_i}{\partial z} ; \quad K_{1,4} = N_i \frac{\partial M_j}{\partial x} \\
 K_{2,1} &= -N_i \frac{\partial N_j}{\partial y} \frac{\partial \nu_i}{\partial x} ; \quad K_{2,2} = N_i \sum_1^n N_k u_k \frac{\partial N_j}{\partial x} + N_i \sum_1^n N_k v_k \frac{\partial N_j}{\partial y} + N_i \sum_1^n N_k w_k \frac{\partial N_j}{\partial z} \\
 &\quad + \nu_i \left(\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right) - N_i \frac{\partial N_j}{\partial x} \frac{\partial \nu_i}{\partial x} \\
 &\quad - \left(2N_i \frac{\partial N_j}{\partial y} \frac{\partial \nu_i}{\partial y} + N_i \frac{\partial N_j}{\partial z} \frac{\partial \nu_i}{\partial z} \right) \\
 K_{2,3} &= -N_i \frac{\partial N_j}{\partial y} \frac{\partial \nu_i}{\partial z} ; \quad K_{2,4} = N_i \frac{\partial M_j}{\partial y} \\
 K_{3,1} &= -N_i \frac{\partial N_j}{\partial z} \frac{\partial \nu_i}{\partial x} ; \quad K_{3,2} = -N_i \frac{\partial N_j}{\partial z} \frac{\partial \nu_i}{\partial y} \\
 K_{3,3} &= N_i \sum_1^n N_k u_k \frac{\partial N_j}{\partial x} + N_i \sum_1^n N_k v_k \frac{\partial N_j}{\partial y} + N_i \sum_1^n N_k w_k \frac{\partial N_j}{\partial z} + \nu_i \left(\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right) \\
 &\quad - \left(N_i \frac{\partial N_j}{\partial x} \frac{\partial \nu_i}{\partial x} + N_i \frac{\partial N_j}{\partial y} \frac{\partial \nu_i}{\partial y} + 2N_i \frac{\partial N_j}{\partial z} \frac{\partial \nu_i}{\partial z} \right) \\
 K_{3,4} &= N_i \frac{\partial M_j}{\partial z} ; \quad K_{4,1} = M_i \frac{\partial N_j}{\partial x} ; \quad K_{4,2} = M_i \frac{\partial N_j}{\partial y} ; \quad K_{4,3} = M_i \frac{\partial N_j}{\partial z} ; \quad K_{4,4} = 0
 \end{aligned}$$

The three velocity components u , v and w are assumed to be constant when formulating the matrices. In view of the convection terms such as $N_i N_k u_k \frac{\partial N_j}{\partial x}$, it is evident that the interchange of i and j results in an unsymmetric element stiffness and global stiffness matrix. Therefore an unsymmetric solution algorithm has to be employed to solve this system of equations. A direct elimination frontal solution method with diagonal pivoting can be incorporated in the solution process. Since diagonal pivoting is not always stable, sometimes it is necessary to search the columns and rows to establish the pivotal element for the elimination process. Hood (1976) suggests a frontal solution technique for unsymmetric matrices with off-diagonal pivoting. Other solvers were also considered as discussed in the next chapter.

3.3.1 Subroutine VFL3D20

This subroutine is Fortran code developed as a part of the ASGARD program. It computes the coefficients of the element stiffness matrix for a fluid element in exactly same way as discussed in the

above section. This subroutine uses the method of Gaussian integration for evaluating different integrals in the formulation. There is an option left to the user to choose different integration schemes for evaluating the velocity terms and the pressure terms in the momentum equations. It was observed in this regard that a higher order integration scheme was required for evaluating the velocity terms since they appear in higher order derivatives in the momentum equations.

There is a flag LCOUNT to be fixed by the user which determines the way the stiffness matrix is to be formed for inviscid and viscous flows, the latter including both laminar and turbulent flows. This has to be done in a different manner since the momentum equations are different for different flows; for example, the Reynolds stress terms appear only in turbulent flows. This subroutine is to be called for each of the elements in the domain in turn at each iteration. After computing the stiffness matrix for an element, VFL3D20 writes this information on a disc file which will be sent to the solver later on. The shape functions and their derivatives for the hierarchical brick element required for the formulation are obtained from another subroutine SFRH built into the ASGARD program. The listings of these subroutines are given in the Appendix IV.

Chapter 4

MAIN FEATURES OF THE PROGRAM

Once the stiffness matrices for the individual elements are computed in the flow domain, they are to be assembled into a grand global matrix and solved for the variables in the domain for the specified boundary conditions. As we have seen in the previous chapter the finite element formulation of the Navier-Stokes equations results in an asymmetric matrix and therefore a subroutine which can handle such matrices has to be employed for the equation solving process. This 'solver' subroutine should be economical as far as the computations are concerned and should need a minimum amount of core storage. It was also seen in the previous chapter that the convective terms make the Navier-Stokes equations non-linear, hence an iterative procedure has to be adopted for solving these non-linear equations. An initial estimate of the solution has to be given as an input to the program which is used to solve the matrix equations and the results are iterated until they converge to a tolerance limit. The method of successive approximation is used for updating the results obtained at a particular iteration before they are carried-over to the next iteration. These important features in the program along with some others are discussed in this chapter.

4.1 The Solver

Different solvers were considered prior to the starting of this project. One of them was developed at the National Research Council, Ottawa. It was discarded because of its inefficiency in handling the zero diagonal entries and predicting the singularity of a matrix. The next solver examined was a frontal solution algorithm used by Taylor and Hughes (1981). This particular solution used the diagonal pivots. Since it is not a general case that the most suitable pivot need not exist on the leading diagonal for a Navier-Stokes problem, a better algorithm was sought.

Finally it was decided to use a frontal solution program developed by Hood (1976) for solving the asymmetric equations. Few errors were found in Hood's program which were rectified in a subsequent Errata published by Hood (1977) himself. It is based on the Gaussian elimination algorithm. Hood observed that in solving the asymmetric matrices, there is no guarantee that using diagonal pivots would be stable. Consequently he incorporated a form of column and row search to select the pivot.

In formulating the program structure, Hood considered three factors :

- 1) The amount of computer storage required,
- 2) The number of non-zero entries encountered in the matrix equations ,
- 3) The choice of the pivot.

Ideally it is desirable to have all the non-zero entries in core for the pivot search in the interests of accuracy and search time, but the available core requirements often prohibit this approach. Therefore Hood developed the alternative strategy of retaining only part of the matrix in the core at any given time, and choosing the largest pivot from this particular area. Hood tested both banded and frontal algorithms using this strategy and found that, although the frontal routines are slightly more complex, they are preferred in general, because they are faster and require less core space as long as active variables can be kept in core. An additional advantage is that no stringent node numbering scheme is needed, though an element numbering helps to minimize the frontwidth.

Considering the fact that in a general Navier-Stokes formulation the diagonal entry need not be the largest, a total search other than a diagonal search is made for the pivot. The solution algorithm could be easily modified for the problems where the largest entry is always at the diagonal; whereas converting the diagonal pivoting to off-diagonal pivoting is difficult. It has been found that for three dimensional problems the frontal routines are often faster. Thus Hood's frontal solver was chosen for the equation solving process in the present work.

The overall solution technique consists of

- 1) A pre-frontal operation,

- 2) An assembly of all the element stiffness matrices into a global stiffness matrix,
- 3) Reduction of the global stiffness matrix using Gaussian elimination,
- 4) Application of the boundary conditions and
- 5) A back substitution process.

Though the solver could be treated in isolation or as a blackbox in the main package, some of its coding details are given in the next section.

4.1.1 Program Details

A subroutine FRONT is called first in the solver program which does the pre frontal assembly and the Gaussian elimination process in a sequence. The pre frontal operation mainly consists of labelling the node numbers negative when they appear for the last time in the mesh, for the assembly purposes. Some modifications were made in the pre frontal operation to create the arrays indexing the value of the first degree-of-freedom at each node and the specification of the values of the boundary conditions wherever they are prescribed. Along with this the array that indexes every fixed node in the mesh is computed using the specification of the boundary conditions given as an input to the program.

After the 'pre front' process is carried out, the assembly process is started by calling the subroutine ABFIND which reads the element stiffness matrix stored on disc for the first element. The ABFIND subroutine then assembles this data into the appropriate locations in the global stiffness matrix and forms the right hand side vector. Then the program assembles the second element in a similar way. The assembly process is carried out entering all the element level information into core, until the core area fixed by the user is filled.

Once this process is over, from within the assembled part of the global matrix (partially filled), a pivotal search is made to determine the largest entry amongst those rows and columns which are fully summed, that is rows and columns to which no further contributions will arise in subsequent assembly of element stiffness matrices. The pivotal row is then used to eliminate all of the coefficients in the pivotal column, after which it is placed on a disc file. When sufficient

coefficients have been eliminated, the next element stiffness matrix is assembled, after which further elimination takes place.

When finally all the coefficients have been eliminated the solution is obtained by a back-substitution routine, BACSUB. This routine starts by inserting the values of the assigned boundary conditions into a vector which gives the final values of the degrees-of-freedom as output. The disc containing the pivotal rows is read for each row in turn, and the back-substitution process is carried out. For more than one right-hand side the entry point to the program for the second and subsequent right-hand sides is via subroutine RESOL, but this feature is not used in this project.

Whenever a part of a package is taken from some other source, it has to be tested thoroughly before implementation. Since the solver could be used for solving any set of algebraic equations, testing becomes simpler. A number of simultaneous equations which form asymmetric matrices, were solved and tested for this purpose before coupling the solver into the main package. Few transfer errors were detected in the program which occurred while transferring the solver from the original source. Problems were tested in which a few of the diagonal entries were zeroes since this is the case with our FEM formulation of the Navier-Stokes equations. It was also tested whether the solver could predict the singularity of a matrix, if it exists. The solver was shown to handle essential boundary conditions correctly.

4.2 Convergence Criterion

Solving non-linear problems using the FEM involves an iterative procedure. The solution obtained in any particular iteration should be compared with that of the previous iteration. When the solution changes by less than a specified amount from one iteration to the next, it is said to have converged. Two popular iterative schemes used in practice in the non-linear FEM analysis are :

- 1) Successive approximation,
- 2) Newton-Raphson method.

Because of the lesser complexity involved, the successive approximation method has been employed in the present project. The successive approximation technique in general involves a consistent way of updating the variables from the previous and current iterations for the next iteration. The entire procedure can be summarized as follows:

- 1) Initial values of the variables are input as a data.
- 2) Results obtained from the solver in the first iteration are transferred to the second iteration as they are, since these are the first set of results, an alternative could be to input a trial solution to compare the results from the first iteration.
- 3) The results obtained from the second iteration are compared with the results of the first iteration.
- 4) If the difference in the value of any individual variable is more than the tolerance limit specified by the user, the values of all the variables are updated.
- 5) The updated values of the variables are transferred to the next iteration.
- 6) This cycle is repeated until the specified tolerance limit is reached, which then gives a converged solution.

The convergence criterion holds a key-role in an iterative solution technique. A tolerance limit is defined as follows,

$$\frac{U_n - U_{n-1}}{U_n} \leq \epsilon \quad (1)$$

where U is a variable and subscripts $n-1$ and n correspond to the $n-1^{th}$ and n^{th} iterations respectively. Considerable amount of care was taken in fixing the tolerance limit ϵ since this is a critical factor in determining the convergence. This value could be varied from problem to problem. For example, if only a rough estimate of a solution is required, the tolerance limit can be fixed to a higher value. When once the solution is obtained the tolerance limit can be changed to a smaller value to obtain a more accurate result. A good starting point can be 0.1, that is the difference in the value of a variable between any two iterations is to be within 10% of the current iteration. Attention was also given in choosing the element size, discretization of the mesh and shape of the element so that the discretization errors are minimized. The aspect ratio or the ratio of any two

adjacent sides of an element is within a ratio of 1:7. The internal angles between any two sides of an element were kept between 45° and 135° . These are the common features for any convergence scheme.

If at any particular iteration the value of any variable exceeds the tolerance limit the whole process is to be re-iterated. While the results of the current iteration are carried-out to the next iteration they are to be updated. The scheme used for updating the variables in the present work is as follows;

$$U_{n+1} = W * U_n + (1 - W) * U_{n-1} \quad (2)$$

where U_{n+1} is the value of a variable U to be input for the $n + 1^{th}$ iteration, U_n and U_{n-1} are its values at the n^{th} and $n - 1^{th}$ iterations respectively. W is the weighting factor whose optimum value depends on a particular problem. Before tackling a big non-linear problem this weighting factor has to be determined. In most of the test problems considered in this project, the value of the weighting factor is fixed at 0.5, so that the updated variable is always the arithmetic mean of the present and previous iterations.

Chapter 5

TEST PROBLEMS AND RESULTS

Once a finite element package is developed it is essential to test it against some simple problems for which exact analytical solutions exist. In some cases these exact solutions may not be available, but good experimental results may be available. For some more complex problems approximate solutions may be available which can be used to demonstrate qualitatively correct behaviour of the FEM solution.

5.1 Test Problem Requirements

The test problems selected should be such that they not only test the correct performance of a program but also demonstrate its capabilities. In general a test problem should meet the following requirements:

- 1) Either an exact analytical solution or an approximate numerical solution should be available to the user beforehand. Well documented experimental results could be a substitute for this purpose.
- 2) The test problem should be such that it could be tested with a minimum number of elements, preferably one. This will reduce the computational cost during the 'debugging' stage.
- 3) The boundary conditions for these test problems should be well defined.
- 4) The test problem should be such that a part of the code could be isolated and tested separately, that is results could be checked in any particular stage in the program

In view of these factors the following problems have been tested with the FEM program developed to solve fluid mechanics problems. For convenience a summary of the geometries and

boundary conditions for the test problems considered is given in Table 5.1. They are described in more detail in the following sections.

5.1.1 Plane-Poiseuille Flow

This is a simple flow between two stationary parallel plates. For a fully developed flow condition Schlichting (1976) gives an exact solution for this problem. The pressure forces balance the viscous forces in the momentum equation. Under a given pressure gradient and with the no-slip boundary condition being applied at the solid boundaries, the unidirectional velocity of the flow takes a parabolic profile between the two parallel plates. This problem was chosen as a test case not only because an exact solution exists but also the FEM domain can be easily discretized in terms of the three dimensional brick elements and the boundary conditions can be easily specified. By making the velocities on the surfaces that act as the solid boundaries go to zero, the no-slip boundary condition is easily imposed.

The velocity u can be expressed as a function of y , distance from the centerline as follows

$$u = -\frac{1}{2\mu} \frac{dp}{dx} \left(\frac{h^2}{4} - y^2 \right) \quad (1)$$

where h is the distance between the plates and dp/dx is the pressure gradient in x direction. As the order of interpolation for the velocity profile is the same as the exact solution, one element is enough to obtain the same solution using FEM. This fact is demonstrated in the Fig.5.1 A Reynolds number is defined based on the centerline velocity and the velocity profiles for different Reynolds numbers using two elements were obtained. When these velocity profiles were non dimensionalized with respect to the centerline velocity, they all fell on a single curve, which is given in Fig.5.2. It was clear from the observations that when the pressure gradient is zero there was no flow taking place. There are no convection terms in the momentum equations governing this flow and hence the problem is totally linear in nature. Agreement with the exact solution to within $10^{-4}\%$ was obtained in a single iteration. It took two iterations to converge since the convergence check is made after the second iteration is over. Because of the linearity of the problem the initial conditions did not have any effect on the solution. It was also observed that under a positive pressure gradient the flow reversed. Since the Poiseuille flow problem is the simplest test case

available and the geometry could be easily interpreted in terms of a FEM mesh, different sets of boundary conditions were imposed as summarized in Table 5.1, to test the program's capability to produce a converged solution.

5.1.2 Couette Flow

Couette flow is another simple flow taking place between two parallel plates, one of which is at rest, and the other moving with a velocity 'U' parallel with the fixed plate. The momentum equations remain the same as in the previous case. This is a shear flow problem with a moving boundary. This boundary condition was imposed for our FEM solution by making the v and w components of velocity zero and u component equal to a specified value on the top plate. Unlike in the case of Plane-Poiseuille flow, even under zero pressure gradient flow takes place, varying linearly from zero at the stationary plate to the value U at the moving plate. Schlichting (1976) gives an exact solution for the velocity for a steady fully developed Couette flow as follows:

$$u = \frac{U}{h} \left(y + \frac{h}{2} \right) - \frac{1}{2\mu} \frac{dp}{dx} \left(\frac{h^2}{4} - y^2 \right) \quad (2)$$

where y is the distance from the centerline, h is the distance between the plates. In fact Plane-Poiseuille flow is a special case of Couette flow where U value is fixed at zero. Partially reverse flow occurs when there is a positive or adverse pressure gradient because of the fact that the influence of the pressure gradient surpasses the effect of the viscous forces.

In this case too, velocity profiles are computed for different Reynolds numbers and when they were non dimensionalized, they fell on a single curve. The velocity profiles, non dimensionalized with respect to the top plate velocity, for different pressure gradients are shown in Figs 5.3a-5.3c. In all the cases the error in the non dimensional velocity compared with the exact solution was less than 0.0001%.

5.1.3 Flow Down an Inclined Plane

In the above two examples the pressure terms and the viscous terms are the only ones included in the momentum equations. In the following example it will be shown how the program can handle

the body force terms in the momentum equations. This problem relates to a fluid flowing down an inclined plane due to gravity. The atmosphere above the fluid has no effect on the fluid and hence there is no pressure variation. The equilibrium conditions are preserved because of the fact that the gravitational or body forces balance the viscous or frictional forces.

Under certain simplified assumptions such as constant film thickness White (1979) gives an exact solution for this problem for the velocity, given by

$$u = \frac{\rho g \sin \theta}{2\mu} y(2h - y) \quad (3)$$

The orientation of the coordinate system is shown in Fig.5.4. $\rho g \sin \theta$ is the body force component acting in the direction of motion of the fluid and θ is the angle of inclination of the plane. Different velocity profiles are computed by varying the magnitude of the body force acting on the fluid. Again all these profiles gave a single non dimensionalized velocity profile as shown in Fig.5.5.

These were some of the problems tested to demonstrate the program's capability to handle the viscous, pressure and body force terms in the momentum equations coupled with the continuity equation. The error found in the FEM solution for these problems was less than 0.0001%.

5.1.4 Flow Between Porous Plates

The previous problems demonstrate the program's ability to handle the linear momentum equations in general. The following example deals with the more critical convective terms, e.g. $u \frac{\partial u}{\partial x}$ etc.

This problem represents the flow between two parallel porous plates, one of them being under suction and the other under injection, both the flow rates at the walls being equal. The main flow is being generated by a constant pressure gradient. The porous walls are such that a uniform crossflow is generated normal to the wall; that is $w = \text{const}$. This results in a momentum equation in x direction with only one left hand side convection term $w \partial u / \partial x$, that is

$$w \frac{du}{dx} = \frac{-1}{\rho} \frac{dp}{dx} + \nu \frac{d^2 u}{dy^2} \quad (4)$$

Since w is constant, the equation is still linear. White (1974) gives an exact solution in a non dimensional form in terms of a wall Reynolds number defined as $Re = \frac{wh}{\nu}$, as follows

$$\frac{u}{u_{max}} = \frac{2}{Re} \left(\frac{y}{h} - 1 + \frac{e^{Re} - e^{Re y/h}}{\sinh Re} \right) \quad (5)$$

where y is the distance from the centerline, $2h$ is the distance between the plates and $u_{max} = \frac{h^2}{2\mu} \left(-\frac{dp}{dx} \right)$ is the centerline velocity for flow where $w = 0$ or Poiseuille flow.

For small values of the wall Reynolds numbers, the last term in the parentheses can be expanded as a power series and the Poiseuille solution is obtained. For large Reynolds numbers, the same last term in the parentheses has the approximate value 2.0 except very near the upper wall, so that the velocity profile could be expressed as

$$\frac{u}{u_{max}} = \frac{2(1 + y/h)}{Re} \quad (6)$$

The velocity profiles at different wall Reynolds numbers are computed using the FEM with a two element mesh. The exact solutions were obtained using the equation (5) since the Reynolds numbers considered were small. The non dimensional velocity profile for a wall Reynolds number of 2 with respect to the centerline velocity of the Poiseuille flow is given in Fig.5.6. There was an estimated error of 23% in the non dimensional velocity compared with that of the exact solution at the centerline. The reason was that there were steep velocity gradients near the top wall and two elements were not sufficient to resolve these gradients. By refining this mesh into a three element mesh as shown in Fig.5.7, the maximum error was reduced to 0.15% compared with the exact solution. The velocity profiles for different wall Reynolds numbers using this mesh are given in the Figs.5.8a-5.8c. The convergence was monotonic for this problem and a weighting factor of 1.0 gave the fastest convergence in 3 iterations. We also observed that as the crossflow increases, the velocity gradients become steeper at the walls and more elements are needed for better approximation.

5.2 Fully developed Laminar and Turbulent Flow

Quite frequently the laminar flow through a circular duct is termed Hagen-Poiseuille flow in the literature. For a fully developed laminar flow through a circular pipe, with no swirl, the

momentum equation in the axial direction could be written in polar coordinates as follows

$$\mu \left(\frac{d^2 u}{dr^2} + \frac{1}{r} \frac{du}{dr} \right) = \frac{dp}{dx} \quad (7)$$

where r is the radial coordinate. Under fully developed flow conditions an exact solution exists for this problem which is given below

$$u(r) = - \frac{1}{4\mu} \frac{dp}{dz} \left(\frac{R^2}{4} - r^2 \right) \quad (8)$$

where R is the radius of the circular duct.

Taylor et al (1977) have solved this problem as an axis-symmetric flow case using five elements spaced between the wall and the centerline. In the present work this test case is solved as a three dimensional problem with v and w components being zero. One quarter of the pipe is considered because of the symmetry of the problem and three elements constituted the mesh as shown in Fig.5.9. The velocity profiles were non dimensionalized with respect to the centerline velocity and the non dimensional velocity profiles for different Reynolds numbers fell on a single curve as shown in Fig. 5.10. Even with a coarse mesh of three elements the exact solution was obtained with an estimated error of less than 0.2% at the nodes near the axis and about 1% at the nodes very close to the wall. A higher percentage error near the solid wall could be because of the curvature effect of the wall, that is with fewer elements the circular boundary of the pipe wall would not have been correctly interpolated in a FEM domain. Though this error is not significant, a finer mesh would reduce this error to a negligible minimum. By changing the coordinates of the nodes, one of them was brought closer to the wall. The error in the non dimensional velocity obtained using the FEM at this node compared to the exact solution increased. This observation confirms the fact that the curvature effect is felt, stronger as we go nearer to the wall. In general to reduce this error at the global level, a FEM mesh is to be refined in such a way that there are more elements near the wall. Since the errors obtained for the laminar flow are not significant (maximum being 1.05 %) and in view of the fact that the computational cost would increase with the number of the elements, this work was not undertaken in this project.

Unlike the case of laminar pipe flow, there does not exist an exact solution for the turbulent pipe flow, but Laufer (1953) performed careful and well-documented measurements for it. In the

current project the turbulence model is based on the eddy-viscosity hypothesis, and information about the eddy-viscosity at each of the nodal points in the flow domain is necessary to carry-out the FEM calculations for this flow. Since a generalized turbulence model which generates this information on its own under specified flow conditions is not built in to the FEM package at this stage, some other source was required for providing this information regarding the eddy-viscosity. Hinze (1956) used Laufer's (1953) data and presented eddy-viscosity distribution in a form as a function of non dimensional radius as given below

$$\frac{\nu_t}{U_{cl}D} = F\left(\frac{r}{R}\right) \quad (9)$$

where ν_t is the eddy-viscosity, U_{cl} is the centerline velocity and D is the diameter of the pipe. This information which is based on the centerline velocity and the diameter of the pipe was utilized in our project to generate an eddy-viscosity distribution in the flow.

Unfortunately, for reasons which are not clear, we were unable to obtain a converged solution for this flow. Including the effects of turbulence results in a significant increase in complexity since the eddy-viscosity is a function of both the distance from the wall and the conditions in the flow. Within the time available, we were unable to resolve a number of questions concerning the finite-element discretization of the eddy-viscosity terms. The incorporation of a turbulence model is therefore left as a future development.

5.3 Further Observations

The algorithm used in this work is basically developed to solve three dimensional flow problems. There does not seem to exist a simple three dimensional problem in the literature which satisfies all our test problem requirements. In our search for a test problem to demonstrate the program's capability to handle the three dimensional effects in the flow, a simple secondary flow through a curved square duct was considered.

When an inviscid, sheared duct flow passes through a bend, it emerges with a streamwise component of vorticity. The corresponding velocity components in the normal and binormal directions are known as 'secondary velocities'. Using the 'secondary flow theory' as summarized

by Hawthorne (1951) it is possible to obtain reasonable, approximate values of these secondary velocities, provided the turning angle is not too large. This flow can therefore be used as a simple test case for demonstrating the three-dimensional capabilities of other numerical flow-prediction techniques, such as the FEM.

The secondary velocity at the outlet plane of a square duct with a 30° bend were obtained using a finite difference scheme and the method of successive over-relaxation (Sjolander, 1984). The boundary condition at the outlet is $\frac{\partial u}{\partial x} = 0$; that is, the streamwise component of velocity is invariant downstream of the bend. To apply this boundary condition to the FEM solution, a special element having this property was developed.

It was originally intended to solve this case as an Euler problem, since this would lead to non-zero velocities at the walls. This in turn would allow the use of elements having nodes at the walls only and would thus minimize the total number of elements, and hence the computational costs. However, during the course of this work it was realized that this approach complicates the specification of the boundary conditions at the solid walls. In the case of a Navier-Stokes solution, the no-slip boundary condition also ensures that there is no flow normal to the wall. By contrast in the Euler solution this condition must be satisfied by requiring the wall velocity to be parallel to the wall. Engelman et al (1982) describe two methods of achieving this: for example, by transforming the stiffness matrix at the element level into a matrix in terms of tangential and normal components of velocity at the nodal points. However, this would involve considerable programming effort. The simplest approach is to solve the case as a Navier-Stokes problem. Unfortunately, this requires an increase in the number of elements (from 4 to a minimum of 16) in order to have nodes on other than the walls or the centerline where the secondary velocities are zero. Because of the considerable computational cost involved with such a mesh this problem was temporarily abandoned. It is suggested that this be the first case to be attempted when the attached processor becomes available in the near future. Up to this point we have thus not demonstrated the three-dimensional capabilities of our program package. However, provided the flow is laminar, we believe that such flows can be predicted without further difficulties.

Chapter 6

CONCLUSIONS

An engineering tool based on the finite element method is developed to solve the Navier-Stokes equations which govern steady, incompressible viscous flow. At this point it is capable of handling laminar flows only. The major observations made during this research are outlined below.

- i) The mixed interpolation scheme, as suggested by the other researchers is found to be successful in obtaining stable and converged solutions. The hierarchical element developed with a quadratic interpolation for the velocities and a linear interpolation for the pressure is successfully implemented.
- ii) The successive approximation method used for updating the variables in between iterations was found to be successful in obtaining convergence in comparison with an exact or an approximate solution. Still, some of the questions on the criterion of convergence have yet to be answered: for example, the basis for choosing the weighting factor which gives the fastest convergence has not been established.
- iii) Few conclusions were drawn concerning the specification of the boundary conditions. Though it is easier to specify the essential boundary conditions, it becomes complicated sometimes to prescribe the natural boundary conditions, for example, the specification of $\frac{\partial u}{\partial z} = 0$ on a boundary. A special technique to modify the element has to be employed to achieve this effect in such a way that 'u' velocity remains the same on the planes normal to the x-axis. Though Euler's equations are a subset of the Navier-Stokes equations, the specification of boundary conditions becomes more complicated. This difficulty essentially arises because of the fact that the tangential component of velocity does not vanish at the solid boundary as in the case of a Navier-Stokes problem. The program could be modified to specify this boundary condition

in order to solve the Euler's equations. Specification of traction boundary conditions, that is specifying the shear force on a boundary is not incorporated in the program since this is not a general case.

- iv) The influence of the initial conditions is insignificant for a linear problem, though a good initial guess gives faster convergence for a non-linear problem.
- v) The program is found to handle problems with curved boundaries effectively. For a more accurate solution, more elements are required very near the curved boundary. With fewer elements in the mesh, the error in the FEM solution increases as the wall is approached. Similarly, it was also observed that more elements are required in the regions where steep velocity gradients exist.
- vi) A higher order integration scheme is recommended for evaluating the velocity terms in the momentum equations than the one to be used for the pressure terms. This observation was made based on the order of interpolations used for different terms at the element level.
- vii) The tolerance limit for convergence of 0.1 which was used in this project was arbitrary. Since most of the test cases are linear in nature, this tolerance limit did not have much of an impact. But when a practical non-linear problem is tackled, this limit can be changed depending on the accuracy requirements and computer cost limitations. A basis for choosing a tolerance limit which gives the required accuracy without unnecessary iterations has to be established.
- viii) The program provides the basic capability needed to analyze viscous flows. However at this stage only laminar flows can be handled by this program and its three-dimensional capability has yet to be demonstrated.

Some suggestions for the future work are discussed in the next section.

6.1 Future Work

In this section some suggestions, some of which can be implemented in the near future and others which can be undertaken in the longer term, are presented.

The suggestions for short term application are as follows:

- i) The program has to be tested to demonstrate its three-dimensional capability. As suggested in the chapter 5, the flow through a curved duct would be a suitable for the first test case.
- ii) The ability of the program to handle turbulent flows needs to be demonstrated. Turbulent fully-developed pipe flow would be a suitable test case since the required eddy-viscosity distribution is available from experimental and can be entered as a table. The program would subsequently be modified to generate its own eddy-viscosity distribution from one of the available, simple models.

Suggestions for longer term application are as follows:

- i) At this point the Navier-Stokes equations are handled in a dimensional form throughout the program. It is desirable to handle the governing equations in a non-dimensional form. A characteristic velocity can be defined based on a datum value and all the terms in the Navier-Stokes equations can be non-dimensionalized with respect to this velocity and some characteristic length.
- ii) It is also desirable to have the equations expressed in a generalized coordinate system. In this case the specification of boundary conditions becomes easier for the Euler problem since one of the co-ordinate directions can be chosen to be normal to the wall.
- iii) Though the eddy-viscosity hypothesis is a simple turbulence model to implement and is very successful in thin shear flows, some questions about its validity arise in more complex flows such as those with separated regions or flow reversal. Therefore, in the longer term consideration should be given to implementing one of the higher-order turbulence models, such as the $k - \epsilon$ model (eg. see Rodi (1982)). Baker (1982) successfully implemented this model in his three-dimensional algorithm.
- iv) Some others suggest that the Newton-Raphson method produces more rapid convergence than the method of successive approximations used in the present work. Since having an efficient iteration procedure will become increasingly important as larger problems are tackled, the possibility of using Newton-Raphson should be examined.

REFERENCES

Baker A.J.	1982	On current aspects of finite element computational fluid mechanics for turbulent flows, <i>Computer Methods In Applied Mechanics And Engineering</i> , Vol.32, p261-282.
Cebeci T.,
Smith A.M.O.	1974	Analysis of turbulent boundary layers, <i>Applied Mathematics and Mechanics</i> , Vol.15, Academic Press, New York.
Chung T.J.	1981	Introduction to finite element analysis in fluids and heat transfer, Vol.1, <i>Proceedings of the short course on Finite Element Analysis in Fluids and Heat Transfer</i> , Published by University of Alabama in Huntsville, Alabama, U.S.A.
Donea J.,
Giuliani S.,
Laval H. and
Quartapelle L.	1982	Finite element solution of the unsteady Navier-Stokes equations by a fractional step method, <i>Computer Methods in Applied Mechanics and Engineering</i> , Vol.30, p53-73.
Engelman M.S.,
Sani R.L. and
Gresho P.M.	1982	The implementation of normal and/or tangential boundary conditions in finite element codes for incompressible fluid flow, <i>International Journal for Numerical Methods in Fluids</i> , Vol.2, p225-238.
Engelman M.S.,
Sani R.L.,
Gresho P.M. and
Bercovier M.	1982	Consistent vs. reduced integration penalty methods for incompressible media using several old and new elements, <i>International Journal for Numerical Methods in Engineering</i> , Vol.2, p25-42.
Goldak J.A.	1983	Private communication, Carleton University, Ottawa, CANADA.

- Galbraith R.A.M., 1976 Mixing length in the wall region of turbulent boundary layers,
Sjolander S. and *Aeronautical Quarterly*, Vol.XXVII, p97-110.
Head M.R.
- Hood P. 1976 Frontal solution program for unsymmetric matrices, *International
Journal For Numerical Methods in Engineering*, Vol.10, p379-399.
- Hood P. 1977 Note on frontal solution program for unsymmetric matrices, *In-
ternational Journal for Numerical Methods in Engineering*, Vol 10,
p1055.
- Hinze J.O. 1959 'Wall' turbulent shear flows, *Turbulence*, p586, second edition,
McGraw-Hill Inc., U.S.A.
- Hawthorne W.R. 1951 Secondary circulation in fluid flow, *Proceedings of the Royal So-
ciety*, Series A, Vol.206, No.A1086, p374-387.
- King I.P., 1975 A finite element solution for two-dimensional stratified flow prob-
Norton W.R. and lems, *Finite Elements in Fluids*, Vol.1, Ed. by R.H. Gallagher,
Iceman K.R. J.T. Oden, C. Taylor and O.C. Zienkiewicz, John Wiley and Sons,
p133-156.
- Kawahara M., 1976 Steady and unsteady finite element analysis of incompressible vis-
Yoshimura N., cous fluids, *International Journal for Numerical Methods in En-
Nakagawa K., gineering*, Vol.10, p437-456.
and Ohsaka H.

- Laufer J. 1953 The structure of turbulence in fully developed pipe flow, NACA Report No.1174.
- Moore J., 1979 A calculation procedure for three dimensional, viscous compressible duct flow: Part1-Inviscid flow considerations; Part2-Stagnation pressure losses in a rectangular elbow, *Journal of Fluids Engineering*, Vol.101, p415-428.
- Moore J.G.
- Prandtl L. 1925 Über die ausgebildete Turbulenz, ZAMM, Vol.5, p136.
- Rodi W. 1982 Examples of turbulence models for incompressible flows, *AIAA Journal*, Vol.20, p872-879.
- Sani R.L., 1981 The cause and cure of the spurious pressures generated by certain FEM solutions of the incompressible Navier-Stokes equations:Part1, *International Journal for Numerical Methods in Fluids*, Vol.1, p17-43.
- Gresho P.M.,
- Lee R.L. and
- Griffiths D.F.
- Schlichting H. 1976 Exact solutions of the Navier-Stokes equations, P83, *Boundary Layer Theory*, McGraw-Hill Inc., U.S.A.
- Sjolander S.A. 1984 Private communication, Carleton University, Ottawa, CANADA.
- Taylor C., 1977 A numerical analysis of turbulent flow in pipes, *Computers and Fluids*, Vol.5, p191-204.
- Hughes T.G.and
- Morgan K.

- Taylor C., 1981 *Finite Element Programming of the Navier-Stokes Equations*, Pineridge Press Ltd., Swansea, U.K.
and Hughes T.G.
- Taylor C. 1973 A numerical solution of the Navier-Stokes equations using the
and Hood P. finite element techniques, *Computers and Fluids*, Vol.1, No.1, p73-100.
- Tuann S.Y., 1976 A study of various finite element solution methods for the Navier-Stokes equations, Report No.14, Structural Research Series, University of British Columbia, Vancouver, B.C., CANADA.
Olson M.D.
- White F.M. 1974 Solutions of the newtonian viscous flow equations, McGraw-Hill Inc., New York, U.S.A.
- Yamada Y., 1975 Finite element analysis of steady fluid and metal flow, *Finite Elements in Fluids*, Vol.1, Ed. by R.H.Gallagher, J.T.Oden, C.Taylor and O.C.Zienkiewicz, John Wiley and sons, p73-94.
Ito K., Yokouchi Y.,
Tamano T. and
Ohteubo T.
- Zienkiewicz O.C., 1981 Hierarchical finite element approaches, error estimates and adaptive finite element approaches, error estimates and adaptive refinement, *Proceedings of the Third Conference on Mathematics of Finite Elements and Applications*, Brunel University, U.K.
Kelly D.W.,
Gago J. and
Babuska I.
- Zienkiewicz O.C., 1983 Higher order finite element approximation, *Finite Elements and Approximations*, p161, John Wiley and sons, U.S.A.
Morgan K.

Fig.2.1 One-dimensional elements and associated hierarchical shape functions and variables of (a) linear, (b) quadratic, (c) cubic form. Ref. to Zienkiewicz and Morgan(1983)

Fig.2.2 Unit hierarchical brick element

Fig.2.3 Flow diagram for the ASGARD package for fluid mechanics

Fig.3.1 Initial and boundary conditions for the problem of flow over a circular cylinder

PLANE-POISEUILLE FLOW

Fig.5.1 Velocity profile for the flow between parallel plates with one element in the FEM mesh

PLANE-POISEUILLE FLOW

Fig.5.2 Non dimensional velocity profile for the flow between parallel plates with two elements in the FEM mesh

Fig.5.3a Non dimensional velocity profile for Couette flow with negative pressure gradient

Fig. 5.3b Non dimensional velocity profile for Couette flow with zero pressure gradient

Fig. 5.3c Non dimensional velocity profile for Couette flow with positive pressure gradient

Fig.5.4 Geometrical representation of the problem of flow down an inclined plane

Fig.5.5 Non dimensional velocity profile for the flow down an inclined plane

Fig.5.6 Non dimensional velocity profile for the flow between porous plates, with two elements in the FEM mesh, at a wall Reynolds number of 2

Fig.5.7 A three element mesh for the flow between porous plates problem

Fig. 5.8a Non dimensional velocity profile for the flow between porous plates, at wall $Re=2$, with a three element mesh

FLOW BETWEEN POROUS PLATES

Fig. 5.8b Non dimensional velocity profile for the flow between porous plates, at wall $Re=3$, with a three element mesh

FLOW BETWEEN POROUS PLATES

Fig. 5.8c Non dimensional velocity profile for the flow between porous plates, $Re = 5$, with a three element mesh

Fig.5.9 A three element mesh for the flow through a circular pipe

Fig. 5.10 Non dimensional velocity profile for the laminar pipe flow

TABLES

Table 2.1

Previous Work With V-P Formulation

Researchers	Type of the element used	Type of problem solved
Taylor and Hood(1974)	Eight noded quadrilaterals (quad:velocity;linear:pressure)	Plane Poiseuille flow flow over a circular cylinder flow in a backward facing step.
Nickell, Tanner and Caswell (1974)	Eight noded quadrilaterals	Plane Poiseuille flow
King, Norton and Iceman (1975)	Six noded triangles (quad:velocity;linear:pressure)	Two dimensional free surface flow over a weir and flow through a constriction
Yamada, Ito, Yokouchi, Tamano, Ohteubo (1975)	Six noded triangles (quad:velocity;linear:pressure)	Plane Poiseuille flow and Couette flow
Kawahara, Yoshimura, Nakagawa, Ohsaka (1976)	Six noded triangles (quad:velocity;linear:pressure)	Plane Poiseuille flow
Engelman M.S., Sani R.L., Gresho P.M., Bercovier M., (1982)	9 node quadrilaterals (Biquadratic velocity;linear pressure)	colliding flow and Gravity flow problems
Donea J., Giuliani S., Laval H., Quartapelle L.(1982)	9 node quadrilaterals (Biquadratic velocity;bilinear pressure)	Unsteady driven cavity flow, flow in a converging channel

Table 2.2

Shape Functions For The Hierarchical Brick Element

Node No.	Shape Function	Node No.	Shape Function
1	$\frac{1}{8} (1 - \eta) (1 - \xi) (1 - \zeta)$	11	$\frac{1}{4} (1 - \eta^2) (1 + \xi) (1 - \zeta)$
2	$\frac{1}{8} (1 + \eta) (1 - \xi) (1 - \zeta)$	12	$\frac{1}{4} (1 - \eta) (1 - \xi^2) (1 - \zeta)$
3	$\frac{1}{8} (1 + \eta) (1 + \xi) (1 - \zeta)$	13	$\frac{1}{4} (1 - \eta) (1 - \xi) (1 - \zeta^2)$
4	$\frac{1}{8} (1 - \eta) (1 + \xi) (1 - \zeta)$	14	$\frac{1}{4} (1 + \eta) (1 - \xi) (1 - \zeta^2)$
5	$\frac{1}{8} (1 - \eta) (1 + \xi) (1 + \zeta)$	15	$\frac{1}{4} (1 + \eta) (1 + \xi) (1 - \zeta^2)$
6	$\frac{1}{8} (1 + \eta) (1 - \xi) (1 + \zeta)$	16	$\frac{1}{4} (1 - \eta) (1 + \xi) (1 - \zeta^2)$
7	$\frac{1}{8} (1 + \eta) (1 + \xi) (1 + \zeta)$	17	$\frac{1}{4} (1 - \eta^2) (1 - \xi) (1 + \zeta)$
8	$\frac{1}{8} (1 - \eta) (1 + \xi) (1 + \zeta)$	18	$\frac{1}{4} (1 + \eta) (1 - \xi^2) (1 + \zeta)$
9	$\frac{1}{4} (1 - \eta^2) (1 - \xi) (1 - \zeta)$	19	$\frac{1}{4} (1 - \eta^2) (1 + \xi) (1 + \zeta)$
10	$\frac{1}{4} (1 + \eta) (1 - \xi^2) (1 - \zeta)$	20	$\frac{1}{4} (1 - \eta) (1 - \xi^2) (1 + \zeta)$

Table 5.1
Summary of The Test Cases

S No	Type of the flow	Geometry	Boundary conditions	Remarks
1	Plane-Poiseuille flow		$u = v = w = 0$ $v = w = 0$ $p = p_1$	No flow is taking place
	a Zero pressure gradient no slip at solid walls		$u = v = w = 0$ $v = w = 0$ $p = p_1$	
	b Positive pressure gradient no slip at solid walls		$u = v = w = 0$ $v = w = 0$ $p = p_1$	Fully developed reverse flow takes place
	c Negative pressure gradient no slip at solid walls		$u = v = w = 0$ $v = w = 0$ $p = p_1$	Fully developed parabolic velocity profile is to be observed between the plate
	d Inlet flow (fully developed) and inlet pressure distribution prescribed, no slip at the solid walls		$u = v = w = 0$ $v = w = 0$ $u = u(y)$ $p = p_1$	Outlet velocity distribution and outlet pressure distribution come out of the solution
				Continued.....

S No	Type of the flow	Geometry	Boundary conditions	Remarks
2	e. Outlet flow (fully developed) and inlet pressure and inlet pressure distribution prescribed. No slip at the walls		$u = v = w = 0$ $p = p_2$ $v = w = 0$ $u = u(y)$	Fully developed velocity profile is to be observed at the inlet and elsewhere in the flow field
	f. Inlet flow distribution (arbitrary) and pressure distribution prescribed. no slip at the solid walls		$u = v = w = 0$ $v = w = 0$ $p = p_1$ $u = u$	velocity distribution for a developing flow and pressure distribution comes out of the solution
	g. Only half of the domain upto the line of symmetry is considered no-slip at bottom plate Maximum velocity at the center line		$u = v = w = 0$ $v = w = 0$ $p = p_1$ $u = u$	One half of the symmetric parabolic profile is to be observed at inlet and outlet of the domain
	h. Couette flow a. Top plate moving with a velocity U. Pressure is constant everywhere no-slip at walls		$u = U; v = w = 0$ $v = w = 0$ $p = c$ $u = u$	A linear velocity profile is observed everywhere in the flow field Continued.....

S No	Type of the flow	Geometry	Boundary conditions	Remarks
3	<p>b Top plate moving with a velocity U Negative pressure gradient No-slip at the walls</p>		$u = U, v = w = 0$ $v = w = 0$ $p = p_1$ $u = v = w = 0$	<p>Parabolic profiles unsymmetric about the centerline to be found everywhere in the flow field</p>
	<p>c Top plate moving with a velocity U Positive pressure gradient No-slip at the walls</p>		$u = U, v = w = 0$ $v = w = 0$ $p = p_1$ $u = v = w = 0$	<p>Partially reverse flow is to be seen everywhere in the flow domain</p>
	<p>a Flow down an inclined plane a body force component in the X direction specified Pressure is constant in the flow</p>		$v = w = 0$ $p = p_1$ $u = v = w = 0$ $v = w = 0$ $p = p_1$	<p>Fully developed parabolic velocity profile is to be observed everywhere</p>

Continued

S No	Type of the flow	Geometry	Boundary conditions	Remarks
4	<p>Flow between porous plates</p> <p>Uniform suction at the bottom plate and uniform injection at the top plate No-slip at the walls Constant pressure gradient</p>		$u=v=w=0 \quad p=p_1$ $u=v=w=0 \quad p=p_2$	<p>Reduced mass flow as compared with Poiseuille flow under the same pressure gradient and geometry is to be observed in the flow domain</p>
5	<p>Hagen-Poiseuille flow</p> <p>No-slip at the curved boundary One-dimensional flow Negative(favourable) pressure gradient in the flow</p>		$u=v=w=0 \quad p=p_1$ $u=v=w=0 \quad p=p_2$ $u=v=w=0 \quad p=p_3$	<p>Parabolic velocity profile with a maximum at the centerline is to be formed all over the flow field</p>

APPENDICES

• APPENDIX I •

Program Documentation

This appendix serves as a user's manual for using the finite element program ASGARD, developed at Carleton University, for solving fluid mechanics problems. A Fortran subroutine listing and sample input and sample output files are given in the other appendices.

The basic algorithm is meant to solve the full form of the Navier-Stokes equations. Though the algorithm could be used to solve the Euler's equations, the program is yet to be modified to incorporate the appropriate boundary conditions. The calculations for the turbulent flow are based on the isotropic eddy-viscosity hypothesis but successful calculation of turbulent flows also has to be demonstrated. An iterative method of successive approximation is used for solving the non linear problems.

This appendix explains how to prepare the input data for a FEM mesh using twenty noded hierarchical brick elements which are specially developed for solving fluid flow problems. The user has to give the global level data as well as the element level data. The user can prescribe different sets of properties to different elements. A special flag in the input, NSPL allows the user to prescribe the acceleration in the X direction in any particular element to be zero. A conversational input program with extensive error checking and diagnostics assists the user in preparing and editing input data files. In the end a graphics package can draw the mesh, number the nodes and elements, contour the velocities.

The code can be used for a wide variety of problems in fluid mechanics at a wide range of Reynolds numbers. At present the program can not generate an eddy-viscosity distribution on its own. The program also does not deal with prescribed gradient type boundary conditions at this stage.

I.1 RUNNING THE PROGRAM

There are three main steps to performing FEM analysis for a fluid mechanics problem.

- (1) Preprocessing: creation of an input file. It often includes plotting a mesh with node numbers and element numbers.
- (2) Processing: generation of an output file using the input file and the ASGARD finite element analysis program.
- (3) Postprocessing: the printing, plotting and interpretation of the results for presentation.

I.2 PREPROCESSING

The user has an option of preparing the input file using the EDIT processor with the information given below or using a conversational input program by typing

```
!XEQ ASGARD_TALK3_XEQ.WASIRON
```

I.3 PREPARING AN INPUT DATA FILE

The input data file for a fluid flow problem contains two distinct sections. The first section of the input file deals with global parameters and is independent of the element type and number or dimensions in the problem. Element dependent information is found in the second section of the input file.

I.3.1 GLOBAL LEVEL SECTION

FIRST LINE Global parameters

NELZ, NFIX, NLOAD, NEXTIF, NROTATE, NSECSM, NEWRHS, MAXRHS, ITERAT,
TITLELEN, TEMPLEN, NCORD, NALGOR, MINCRE, MITERA, THETA, DTIME,
TOLERA, FIXSTEP

NELZ the total number of elements in this problem

NFIX the total number of nodes with prescribed velocities and pressures

NLOAD the total number of nodes with prescribed shear stresses for the present problems this is prescribed zero

NEXTIF the total number of nodes with prescribed conductances applicable for heat transfer problems, for fluid flow problems this is set to zero.

NROTATE the number of nodes with rotated or inclined boundary conditions

NSECSM number of nodal pairs with sectorial symmetry. See sectorial symmetry.

NEWRHS number of right hand sides in this problem (default is 1). See Jennings's 'Matrix Computations...' page 101

MAXRHS the maximum number of right hand sides expected in an future resolution of the problem (default = 1).

ITERAT a flag = 1 for iterative or non-linear problems.

TITLELEN the number of lines occupied by title and abstract in the input file. Users are advised to include full details, including the reasons for doing the problem. There is no limit on the number of lines for title and description. For example, if 6 lines of the input file are used for the title and description, then TITLELEN = 6.

TEMPLN set to 10,000. It is used for dynamic memory management. Most users may safely set this parameter to 10,000 and forget it.

NCORD the number of parameters per nodal point. The coordinates x, y, z, the three velocities u, v and w, and the pressure. The user could specify the fluid properties at virtually every node with minor changes in coding. Default is the no. of dimensions in the problem.

NALGOR integer to define the algorithm; 144 for incompressible steady state analysis of fluids.

MINCRE the maximum no of increments in this problem (default=1).

MITERA maximum number of iterations in any time step (default=1)

THETA a real number ranging from 0 to 1 that defines a weighting factor for updating the variables in between iterations, a common example being 0.5

DTIME magnitude of the first time increment in seconds for steady state problems it is fixed at any arbitrary number

TOLERA a convergence parameter. It defines the tolerance limit for the non-linear problems, not applicable for the first iteration.

FIXSTEP the number of time increments with fixed DTIME for steady state problems this is fixed at zero

NEXT TITLELEN LINES are title and description of problem

NOTE: The title and description must contain exactly TITLELEN lines.

NEXT LINE Element topology—repeat one line for each element

NEL, LTYPE, LPOP, LNODS1

NEL the element numbers in sequence with no missing elements

LTYPE an integer specifying the element type, eg. 144 for 20 node hierarchical brick for fluid flow analysis.

LPOP the property list number for this element

LNODS the element nodes listed in the correct sequence, counter clockwise for 2D elements. See element library for 3D elements in DOC1.WASIRON

NEXT LINE Nodal Coordinates

NODE, X-coord, Y-coord, Z-coord

NODE the global node number in any order except the largest number must be last. Midedge nodes may be omitted if the edge is straight.

X-coord x coordinate of this node
 Y-coord y coordinate of this node if y coordinate exists
 Z-coord z coordinate of this node if z coordinate exists

* NEXT LINE Prescribed velocities and pressures NFIX lines

SEQUENCE_No. NODE_No. FIXITY NODAL_VARIABLES

Nodes may be entered in any order. The FIXITY value determines the variables that are fixed at this node. A 1 corresponds to a fixed variable and a 0 corresponds to a free variable. Hence for a corner node FIXITY is four digit number and for a mid-edge node it is a three digit number. For example, 1111 at a corner node means the three velocities as well as the pressure are fixed at this node.

NEXT LINE Prescribed tractions NLOAD lines

SEQUENCE_No. NODE_No. NODAL_STRESS in Newtons/meter square

Prescribed tractions can be entered as element level nodal stresses which is often more convenient in some field problems. This prescription is not yet implemented in the program.

1.3.2 ELEMENT LEVEL SECTION

This section deals with property lists of each of the elements in the mesh. The sequence number at the beginning of each of the property list identifies a particular property list.

NEXT LINE Property List

- one line for each property table

SEQUENCE_No. DYNVIS RHOIN BFORX BFORY BFORZ UINIT VINIT WINIT

LCOUNT VSCHEM PSHEM NSPL

DYNVIS coefficient dynamic viscosity of the fluid in Kg/m-sec

RHOIN mass density of the fluid in Kg/m³

BFORX	body force component in the X direction per unit volume
BFORY	body force component in the Y direction per unit volume
BFORZ	body force component in the Z direction per unit volume
UINIT	uniform U velocity component given as an initial condition
VINIT	uniform V velocity component given as an initial condition
WINIT	uniform W velocity component given as an initial condition
LCOUNT	flag for determining the type of the flow 0 for an inviscid flow 1 for the laminar flow 2 for the turbulent flow
VSCHEM	an integration scheme for summing up the velocity terms in the momentum equation
PSCHEM	an integration scheme for summing up the pressure terms in the momentum equation
NSPL	a flag given to a particular elements to incorporate special properties such as zero acceleration in the X direction

I.4 PROCESSING

The computer must know the name of the input file (for purposes of illustration we will call it `fid_IN`), the name user wishes to have attached to the output file, eg `fid_OUT`, and the name of the program to execute, `A_FLOW3D` in this case. The above steps are most conveniently done by building yet another file that could be called `fid_XEQ` that might read as follows:

```
!SET 5 fid_IN,FUN = IN
```

```
!DELETE fid_OUT
```

```
!SET 6 fid_OUT,CTG = YES,EXIST = NEWFILE
```

```
*A_FLOW3D.
```


FUN=IN confirms that it is an input file. Deleting the output file before starting avoids appending the results to the end of the file if it exists. This is the user's option. CTG=YES will preserve any part of the output file that is created even if the job aborts because of errors - a useful feature.

To process or execute the job, the following has to be typed

```
!XEQ fid_XEQ
```

1.5 POSTPROCESSING

It is usual to begin with the following command to determine how many lines are in the output file

```
!L fid_OUT
```

Often one goes into EDIT to check the last 20 lines of the file for errors and to see that the job completed successfully before printing a hard copy listing

PRINTING a hard copy listing of fid_OUT.

```
!C fid_OUT TO LP ME351
```

For long listings the user can save money by having the listings printed after 11 pm. To do this the following has to be typed

```
!XEQ PRINT.WASIRON
```

1.6 PLOTTING RESULTS

The user may plot the mesh, velocity contours, or a surface representation of the velocity using GNU. See DOC_GRAPHICS.WASIRON

APPENDIX II

Sample Input File

```
3 39 0 0 0 0 0 0 1 5 60000 7 43 0 20 0.5 0 .1 0
THIS REPRESENTS THE PROBLEM OF LAMINAR FLOW IN A CIRCULAR CYLINDER
NO BODY FORCES ARE INCLUDED (-) PRESSURE GRADIENT =1
THE VISCOSITY IS FIXED AT 1. ONLY X VEL CONSIDERED
VSCHM IS 3 AND PSCHM IS 2. THREE ELEMENTS ARE CONSIDERED
SECTORIAL SYMMETRY OF THE GEOMETRY IS CONSIDERED.
1 144 1 1 2 3 4 5 6 7 8 16 16 17 18 19 20 21 22 23 24 25 26
2 144 1 4 3 9 12 8 7 10 11 17 27 33 30 22 21 28 31 25 29 34 32
3 144 1 5 6 7 8 14 13 10 11 23 24 25 26 38 36 29 32 39 35 34 37
2 2 0 0
16 2 1 0
3 2 2 0
27 2 3 0
9 2 4 0
20 2 0 1
21 2 2 1
28 2 3.6955182 1.5307337
6 2 0 2
24 2 1 2
7 2 2 2
29 2 2.4142136 2.4142136
10 2 2.8284271 2.8284271
36 2 0 3
35 2 1.5307337 3.6955182
13 2 0 4
1 0 0 0
18 0 1 0
4 0 2 0
30 0 3 0
12 0 4 0
19 0 0 1
22 0 2 1
31 0 3.6955182 1.5307337
5 0 0 2
26 0 1 2
8 0 2 2
32 0 2.4142136 2.4142136
11 0 2.8284271 2.8284271
38 0 0 3
37 0 1.5307337 3.6955182
14 0 0 4
15 1 0 0
17 1 2 0
33 1 4 0
23 1 0 2
```

25 1 2 2
34 1 2.8284271 2 8284271
39 1 0 4
1 1 0111 0 0 0 2.0
2 2 0111 0 0 0 0
3 3 0111 0 0 0 0
4 4 0111 0 0 0 2 0
5 5 0111 0 0 0 2 0
6 6 0111 0 0 0 0
7 7 0111 0 0 0 0
8 8 0111 0 0 0 2.0
9 9 1111 0 0 0 0
10 10 1111 0 0 0 0
11 11 1111 0 0 0 2 0
12 12 1111 0 0 0 2.0
13 13 1111 0 0 0 0
14 14 1111 0 0 0 2 0
15 15 111 0 0 0
16 16 011 0 0 0
17 17 111 0 0 0
18 18 011 0 0 0
19 19 011 0 0 0
20 20 011 0 0 0
21 21 011 0 0 0
22 22 011 0 0 0
23 23 111 0 0 0
24 24 011 0 0 0
25 25 111 0 0 0
26 26 011 0 0 0
27 27 011 0 0 0
28 28 111 0 0 0
29 29 011 0 0 0
30 30 011 0 0 0
31 31 111 0 0 0
32 32 011 0 0 0
33 33 111 0 0 0
34 34 111 0 0 0
35 35 111 0 0 0
36 36 011 0 0 0
37 37 111 0 0 0
38 38 011 0 0 0
39 39 111 0 0 0
1 1 1 0 0 0 0.3 0 0 1 3 2
1 0 0 0 0
2 0 0 0 0
3 0 0 0 0

APPENDIX III

Sample Output File

THIS REPRESENTS THE PROBLEM OF LAMINAR FLOW IN A CIRCULAR CYLINDER
 NO BODY FORCES ARE INCLUDED. (-) PRESSURE GRADIENT =1
 THE VISCOSITY IS FIXED AT 1. ONLY X VEL CONSIDERED
 VSCHM IS 3 AND PSCHM IS 2. THREE ELEMENTS ARE CONSIDERED
 SECTIONIAL SYMMETRY OF THE GEOMETRY IS CONSIDERED.

```

Time      Date
22:14 52   07/19/84
This is version E01 of ASGAR
Machine run on is Honeywell CP6
Run unit used is A_FLOW3D.F88G0L04
Input file : GURU.DAT033
Output file : *OUT
    
```

LENGTH OF VECTOR OF FLOATING WORDS = LENVEC =80000
 NUMBER OF INTEGER WORDS PER FLOATING WORD = INTEG = 2

*** WE HAVE A NEW JOB, SO NEWJOB = 1

IT IS NOT A RE-SOLUTION, SO IFCOEF = 1

```

NUMBER OF ELEMENTS = NELZ = 3
NUMBER OF NODES WITH SOME FIXED VALUES = NFIX = 30
NUMBER OF NODES WITH ADDITIONAL LOADS = NLOAD = 0
NUMBER OF NODES WITH ADDITIONAL STIFFNESSES = NEXTRF = 0
THE NUMBER OF NODES WITH ROTATED DOF = 0
NUMBER OF NODES RELABELLED FOR SECTIONIAL SYMMETRY=NSECSM= 0
NUMBER OF RIGHT HAND SIDES = NEWRHS = 1
MAXIMUM N.H.S. ENVISAGED IN RE-SOLUTIONS = MAXRHS = 1
IS IT AN ITERATIVE PROBLEM. ITERAT = 1
NUMBER OF INCREMENTS WITH A TIME STEP FIXED, FXSTEP = 0
THE NUMBER OF PARAMETERS PER NODE = 7
THE NUMBER OF INCREMENTS WILL NOT EXCEED = 1
THE NUMBER OF ITERATIONS PER INCREMENT IS LIMITED TO 20
    
```

NALGOR=43; 3D Navier - Stokes velocity pressure formulation

NUMBER OF ELEMENT TYPES NOW IMPLEMENTED = MAXTYP =144

ELEMENT NUMBER	ELEMENT TYPE	PROPERTY = LPROP	NODE NUMBERS = LNODS
-------------------	-----------------	---------------------	----------------------

1	144	1	1	2	3	4	5	6	7	8	15	16	17	18	19	20
21	22	23	24	25	26											
2	144	1	4	3	9	12	8	7	10	11	17	27	33	30	22	21
28	31	25	29	34	32											
3	144	1	5	6	7	8	14	13	10	11	23	24	25	26	38	36
29	32	39	35	34	37											

MAXIMUM NODE NUMBER = NODMAX = 39

MAXIMUM NODES PER ELEMENT = LEOMAX = 20

MAXIMUM DEGREES OF FREEDOM PER NODE = NDFMAX = 4

NUMBER OF DIMENSIONS, 2 OR 3, = MDIM = 3

THE NUMBER OF PARAMETERS PER NODE = NCORD = 7

NUMBER OF PROPERTIES, E.G. THICKNESS, DENSITY = IPROP = 12

NUMBER OF SETS OF PROPERTIES AVAILABLE = JPROP = 1

MAXIMUM NUMBER OF STRESSES AT A POINT = MAXTRS = 1

NODE	X	Y	Z
2	2.0000	.0000	.0000
18	2.0000	1.0000	.0000
3	2.0000	2.0000	.0000
27	2.0000	3.0000	.0000
9	2.0000	4.0000	.0000
20	2.0000	.0000	1.0000
21	2.0000	2.0000	1.0000
28	2.0000	3.8955	1.5307
6	2.0000	.0000	2.0000
24	2.0000	1.0000	2.0000
7	2.0000	2.0000	2.0000
29	2.0000	2.4142	2.4142
10	2.0000	2.8284	2.8284
38	2.0000	.0000	3.0000
35	2.0000	1.5307	3.8955
13	2.0000	.0000	4.0000
1	.0000	.0000	.0000
18	.0000	1.0000	.0000
4	.0000	2.0000	.0000
30	.0000	3.0000	.0000
12	.0000	4.0000	.0000
19	.0000	.0000	1.0000
22	.0000	2.0000	1.0000
31	.0000	3.8955	1.5307
5	.0000	.0000	2.0000
26	.0000	1.0000	2.0000
8	.0000	2.0000	2.0000
32	.0000	2.4142	2.4142
11	.0000	2.8284	2.8284
36	.0000	.0000	3.0000
37	.0000	1.5307	3.8955
14	.0000	.0000	4.0000
15	1.0000	.0000	.0000
17	1.0000	2.0000	.0000
33	1.0000	4.0000	.0000

23	1 0000	0000	2 0000
25	1 0000	2 0000	2 0000
34	1 0000	2 8284	2 8284
39	1 0000	0000	4 0000

RHS	FIXING NUMBER	NODE (NODFIX)	CODE (KODFIX)	FIXED VALUES (VFIX)			
1	1	1	111	.0000000	.0000000	.0000000	2 0000000
1	2	2	111	.0000000	.0000000	.0000000	.0000000
1	3	3	111	.0000000	.0000000	.0000000	.0000000
1	4	4	111	.0000000	.0000000	.0000000	2 0000000
1	5	5	111	.0000000	.0000000	.0000000	2 0000000
1	6	6	111	.0000000	.0000000	.0000000	.0000000
1	7	7	111	.0000000	.0000000	.0000000	.0000000
1	8	8	111	.0000000	.0000000	.0000000	2 0000000
1	9	9	1111	.0000000	.0000000	.0000000	.0000000
1	10	10	1111	.0000000	.0000000	.0000000	.0000000
1	11	11	1111	.0000000	.0000000	.0000000	2 0000000
1	12	12	1111	.0000000	.0000000	.0000000	2 0000000
1	13	13	1111	.0000000	.0000000	.0000000	.0000000
1	14	14	1111	.0000000	.0000000	.0000000	2 0000000
1	15	15	111	.0000000	.0000000	.0000000	.0000000
1	16	16	11	.0000000	.0000000	.0000000	.0000000
1	17	17	111	.0000000	.0000000	.0000000	.0000000
1	18	18	11	.0000000	.0000000	.0000000	.0000000
1	19	19	11	.0000000	.0000000	.0000000	.0000000
1	20	20	11	.0000000	.0000000	.0000000	.0000000
1	21	21	11	.0000000	.0000000	.0000000	.0000000
1	22	22	11	.0000000	.0000000	.0000000	.0000000
1	23	23	111	.0000000	.0000000	.0000000	.0000000

1	24	24	11	0000000	0000000	0000000	0000000
1	25	25	111	0000000	0000000	0000000	0000000
1	26	26	11	0000000	0000000	0000000	0000000
1	27	27	11	0000000	0000000	0000000	0000000
1	28	28	111	0000000	0000000	0000000	0000000
1	29	29	11	0000000	0000000	0000000	0000000
1	30	30	11	0000000	0000000	0000000	0000000
1	31	31	111	0000000	0000000	0000000	0000000
1	32	32	11	0000000	0000000	0000000	0000000
1	33	33	111	0000000	0000000	0000000	0000000
1	34	34	111	0000000	0000000	0000000	0000000
1	35	35	111	0000000	0000000	0000000	0000000
1	36	36	11	0000000	0000000	0000000	0000000
1	37	37	111	0000000	0000000	0000000	0000000
1	38	38	11	0000000	0000000	0000000	0000000
1	39	39	111	0000000	0000000	0000000	0000000

NUMBER ELEMENT PROPERTIES = VPROP

1	.1000000E+01	.1000000E+01	.0000000E+00	.0000000E+00	.0000000E+00
3000000E+00	.0000000E+00				
	.0000000E+00	.1000000E+01	.3000000E+01	.2000000E+01	.0000000E+00

DEGREES OF FREEDOM AT NODES OF ELEMENT OF TYPE144 = NDF = 4 4 4 4 4 4 4 3 3 3 3
3 3 3 3 3 3 3

CREATE ELEMENT FILE

ELEMENT 1 OF TYPE144, WITH PROPERTY TABLE NUMBER 1 AND NODES -1 -2 3
4 5 6 7 8
-15 -16 17 -18 -19 -20 21 22 23 24 25 26

ELEMENT 2 OF TYPE144, WITH PROPERTY TABLE NUMBER 1 AND NODES -4 -3 -
9 -12 8 7 10 11

-17 -27 -33 -30 -22 -21 -28 -31 25 29 34 32

ELEMENT 3 OF TYPE144, WITH PROPERTY TABLE NUMBER 1 AND NODES -6 -6 -
 7 -8 -14 -13 -10 -11
 -23 -24 -25 -26 -38 -36 -29 -32 -39 -35 -34 -
 37

**** FINAL CONVERGED SOLUTION FOLLOWS****

Node	U_Val	V_Val	W_Val	Pr
1	3999374832E+01	.0000000000E+00	.0000000000E+00	.2000000000E+01
2	3999374832E+01	.0000000000E+00	.0000000000E+00	.0000000000E+00
3	2990276596E+01	.0000000000E+00	.0000000000E+00	.0000000000E+00
4	2990276596E+01	.0000000000E+00	.0000000000E+00	.2000000000E+01
5	2990276596E+01	.0000000000E+00	.0000000000E+00	.2000000000E+01
6	2990276596E+01	.0000000000E+00	.0000000000E+00	.0000000000E+00
7	1992487262E+01	.0000000000E+00	.0000000000E+00	.0000000000E+00
8	1992487262E+01	.0000000000E+00	.0000000000E+00	.2000000000E+01
9	.0000000000E+00	.0000000000E+00	.0000000000E+00	.0000000000E+00
10	.0000000000E+00	.0000000000E+00	.0000000000E+00	.0000000000E+00
11	.0000000000E+00	.0000000000E+00	.0000000000E+00	.2000000000E+01
12	.0000000000E+00	.0000000000E+00	.0000000000E+00	.2000000000E+01
13	.0000000000E+00	.0000000000E+00	.0000000000E+00	.0000000000E+00
14	.0000000000E+00	.0000000000E+00	.0000000000E+00	.2000000000E+01
15	3999374832E+01	.0000000000E+00	.0000000000E+00	
16	3742899047E+01	.0000000000E+00	.0000000000E+00	
17	2990276596E+01	.0000000000E+00	.0000000000E+00	
18	3742899047E+01	.0000000000E+00	.0000000000E+00	
19	3742899047E+01	.0000000000E+00	.0000000000E+00	
20	3742899047E+01	.0000000000E+00	.0000000000E+00	
21	2747448165E+01	.0000000000E+00	.0000000000E+00	
22	2747448165E+01	.0000000000E+00	.0000000000E+00	
23	2990276596E+01	.0000000000E+00	.0000000000E+00	
24	2747448165E+01	.0000000000E+00	.0000000000E+00	
25	1992487262E+01	.0000000000E+00	.0000000000E+00	
26	2747448165E+01	.0000000000E+00	.0000000000E+00	
27	1767708433E+01	.0000000000E+00	.0000000000E+00	
28	.0000000000E+00	.0000000000E+00	.0000000000E+00	
29	1095953329E+01	.0000000000E+00	.0000000000E+00	
30	1767708433E+01	.0000000000E+00	.0000000000E+00	
31	.0000000000E+00	.0000000000E+00	.0000000000E+00	
32	1095953329E+01	.0000000000E+00	.0000000000E+00	
33	.0000000000E+00	.0000000000E+00	.0000000000E+00	
34	.0000000000E+00	.0000000000E+00	.0000000000E+00	
35	.0000000000E+00	.0000000000E+00	.0000000000E+00	
36	1767708433E+01	.0000000000E+00	.0000000000E+00	
37	.0000000000E+00	.0000000000E+00	.0000000000E+00	
38	1767708433E+01	.0000000000E+00	.0000000000E+00	
39	.0000000000E+00	.0000000000E+00	.0000000000E+00	

NEW SOLUTION IS IT A NEW JOB. 0
ARE THERE NEW COEFFICIENTS 0 NOW WE HAVE 1 RIGHT HAND SIDES.

APPENDIX IV

Program Listings

```

SUBROUTINE VFL3D20(COORD,ESTIFM,IPROP,JPROP,LNODS,LNOMAX,
1LPOP,NDIM,NEL,NELZ,LVABZ,LNODZ,LTYPE,NODMAX,NITERA,VPROP)
* NAME: VFL3D20
*
* DESCRIPTION: To compute the effective element stiffness matrix
* and effective load vector for the hierarchical
* formulation using a 20 node and a 8 node brick
* elements for the full form of Navier-Stokes equatio
* ns. Notation favors viscous steady three dimensiona
* l flow problems. Turbulence effects are incorporated.
* Numerical integration is used throughout.
* DATE: 3 rd October 1983
*
*****
***** DICTONARY OF VARIABLES *****
*****
*BFORX : BODY FORCE IN X DIRECTION
*BFORY : BODY FORCE IN Y DIRECTION
*BFORZ : BODY FORCE IN Z DIRECTION
*CONVTN : CONVECTION TERMS IN THE MOMENTUM EQUATION
*DERG : GLOBAL DERIVATIVE OF THE SHAPE FUNCTION
*DERL : LOCAL DERIVATIVE OF THE SHAPE FUNCTION
*DERSHX : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. X
*DERSHY : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. Y
*DERSHZ : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. Z
*DERSXI : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. X FOR
* THE 'I'TH ROW
*DERSXJ : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. X FOR
* THE 'J'TH COLUMN
*DERSYI : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. Y FOR
* THE 'I'TH ROW
*DERSYJ : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. Y FOR
* THE 'J'TH COLUMN
*DERSZI : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. Z FOR
* THE 'I'TH ROW
*DERSZJ : DERIVATIVE OF THE SHAPE FUNCTION W.R.T. Z FOR
* THE 'J'TH COLUMN
*DIFSON : DIFFUSION TERM IN THE MOMENTUM EQUATION
*DVOLUM : RATIO OF THE VOLUME OF AN ELEMENT TO THAT OF AN UNIT BRICK
*DYNVIS : MOLECULAR DYNAMIC VISCOSITY OF THE FLUID
*ELXYZT : GLOBAL COORDINATES OF THE CURRENT ELEMENT
*ERESU : RIGHT HAND SIDE TERM FOR U MOMENTUM EQUATION
*ERHSV : RIGHT HAND SIDE TERM FOR V MOMENTUM EQUATION
*ERESW : RIGHT HAND SIDE TERM FOR W MOMENTUM EQUATION
*IGAUS : COUNTER FOR THE GAUSS POINT

```

```

ESTIFM(IROWV,JCOLP)=ESTIFM(IROWV,JCOLP)+SHAPE*DESYJ*DVOLUM/RHOIN
ESTIFM(IROWV,JCOLP)=ESTIFM(IROWV,JCOLP)+SHAPE*DESZJ*DVOLUM/RHOIN
ENDIF
*****
***      Form the continuity equation      *
*****
      IF(IPCON.LE.8) THEN
ESTIFM(IROWP,JCOLU)=ESTIFM(IROWP,JCOLU)+SHAPE*DVOLUM*DERSXJ
ESTIFM(IROWP,JCOLV)=ESTIFM(IROWP,JCOLV)+SHAPE*DVOLUM*DESYJ
ESTIFM(IROWP,JCOLW)=ESTIFM(IROWP,JCOLW)+SHAPE*DVOLUM*DESZJ
ENDIF
*      End the loop for the 'j'th column
*
170  CONTINUE
*
***      End the loop for the 'i'th row
*
180  CONTINUE
*****
***      End the loop for the gauss point integration      *
*****
140  CONTINUE
*
      IF(NSPL.EQ.1) THEN
CALL NOXAXL(ESTIFM,LVABZ)
ELSEIF (NSPL.EQ.2) THEN
CALL RAGHAVENDER(ESTIFM,NEL,LVABZ)
ENDIF
WRITE(UNIT=1,REC=NEL)LVABZ,ESTIFM
WRITE(UNIT=3,REC=NEL+NELZ) LMODZ,ERHSU,ERHSV,ERNHV
RETURN
C      STOP
END
*
*****
*
*      SUBROUTINE BRICKH(LMODZ,LVABZ,LNOMAX,NEL,NELZ,XITA,
1          SHAPE,ELXYZ,DERIV,VOL)
*
*      THIS SUBROUTINE BRIC20 COMPUTES THE JACOBIAN MATRIX OF THE
*      TRANSFORMATION, ITS DETERMINANT WHICH IS THE VOLUME OF THE
*      TRANSFORMATION, THE INVERSE OF THIS MATRIX, AND FINALLY THE
*      SHAPE FUNCTIONS FOR THIS ELEMENT (A 20-NODE BRIC) IN BOTH
*      LOCAL AND GLOBAL COORDINATES.
*
*      IMPLICIT LOGICAL (A-Z)
*
*      Declarations for the parameters
INTEGER
1  LMODZ, LNOMAX, LVABZ, NEL, NELZ, OERR
REAL
1  XITA(3)
*
*      Declarations for the local variables or those in COMMON blocks
*

```

```

*IPCON : COUNTER FOR THE ROW IN THE PRESSURE LOOP
*IPROP : NUMBER OF THE ITEM IN THE PROPERTY LIST
*IROWP : COUNTER FOR THE ROW OF THE CONTINUITY EQUATION
*IROWU : COUNTER FOR THE ROW OF U COMPONENT OF VELOCITY
*IROWV : COUNTER FOR THE ROW OF V COMPONENT OF VELOCITY
*IROWW : COUNTER FOR THE ROW OF W COMPONENT OF VELOCITY
*JCOLP : COUNTER FOR THE COLUMN OF THE PRESSURE TERMS
*      : IN THE MOMENTUM EQUATIONS
*JCOLU : COUNTER FOR THE COLUMN OF U COMPONENT OF VELOCITY
*JCOLV : COUNTER FOR THE COLUMN OF V COMPONENT OF VELOCITY
*JCOLW : COUNTER FOR THE COLUMN OF W COMPONENT OF VELOCITY
*JPCON : COUNTER FOR THE COLUMN IN COMPUTING THE PRESSURE TERMS
*JPROP : NUMBER OF THE PROPERTY LIST
*KPOIN : FLAG FOR THE GLOBAL NODE NUMBERING
*LCOUNT : COUNTER FOR THE TYPE OF THE FLOW, 1. 0 FOR INVISCID FLOW
*      : 1 FOR LAMINAR FLOW 2 FOR VISCOUS TURBULENT FLOW
*LNODS : ARRAY RELATING LOCAL TO GLOBAL NODE NUMBERING
*LVABZ : TOTAL NUMBER OF DEGREES OF FREEDOM IN AN ELEMENT
*MIDS : ARRAY OF THE MID-SIDE NODES IN THE ELEMENT
*ND : A COUNTER FOR THE NODE
*NDIM : NUMBER OF DIMENSIONS OF THE PROBLEM
*NEL : NUMBER OF THE CURRENT ELEMENT BEING ASSEMBLED
*NELZ : TOTAL NUMBER OF ELEMENTS IN THE MESH
*NGAUS : NUMBER OF GAUSS POINTS
*NITERA : COUNTER FOR THE NUMBER OF ITERATION
*NOPP : ARRAY OF THE CODED VALUES OF THE FIRST DEGREE OF FREEDOM
*PRCON : COUNTER TO LOCATE THE CONTINUITY EQUATION IN THE E.S.M.
*PSCHEM : FLAG TO DETERMINE THE INTEGRATION SCHEME FOR THE PRESSURE
*      : TERMS
*RHOIN : DENSITY OF THE FLUID
*RNST(1-9) : ADDITIONAL TERMS IN THE MOMENTUM EQUATIONS
*      : BECAUSE OF THE REYNOLDS STRESSES
*SHAFUN : SHAPE FUNCTION VALUE
*TURVDX : DERIVATIVE OF THE TURBULENT VISCOSITY W.R.T. X
*TURVDY : DERIVATIVE OF THE TURBULENT VISCOSITY W.R.T. Y
*TURVDZ : DERIVATIVE OF THE TURBULENT VISCOSITY W.R.T. Z
*TURVIS : TURBULENT VISCOSITY
*TVISC : VALUE OF THE TURBULENT VISCOSITY AT A NODAL POINT
*UMAX : MAXIMUM VELOCITY IN THE FLOW DOMAIN
*UVELY : U COMPONENT OF VELOCITY
*VISLAM : MOLECULAR KINEMATIC VISCOSITY OF THE FLUID
*VSCHM : FLAG TO DETERMINE THE INTEGRATION SCHEME FOR THE VELOCITY
*      : TERMS
*VVELY : V COMPONENT OF VELOCITY
*WVELY : W COMPONENT OF VELOCITY
*XGAUS : THE LOCAL X-COORDINATE AT THE GAUS POINT
*XITA : THE LOCAL COORDINATES OF THE GAUS POINTS
*YGAUS : THE LOCAL Y-COORDINATE AT THE GAUS POINT
*ZGAUS : THE LOCAL Z-COORDINATE AT THE GAUS POINT

```

```

*****

```

IMPLICIT LOGICAL (A-Z)

INTEGER

```

& IPROP, JPROP, LCOUNT, LNOMAX, LPOP, NDIM, NEL, NELZ, LVABZ, LNODZ, VSCHM,
& NOD, ND, NODE, NODMAX, NITERA, INODZ, IEVAB, JEVAB, IGAUS, LTYPE, NGAUS, I,
& KPOIN, ICON, IROWU, IROWV, IROWW, IROWP, JCOLU, JCOLV, JCOLW, JCOLP, NSPL

```

```

* ,PSCHEM,JCON,IPCON,JPCOM,LNODS(LNODZ,NELZ)
REAL
* VISLAM,RHOIN,BFORX,BFORZ,DVOLUM,UVELY,VVELY,WVELY,
* RNST1,RNST2,RNST3,RNST4,RNST5,RNST6,RNST7,RNST8,RNST9,DIFSON,
* CONVTM,TURVDX,TURVDY,TURVDZ,TURVIS,VOLUME,UMAX,
* COORD(NODMAX,7),ESTIFM(LVABZ,LVABZ),XITA(3),ERHSU(20),
* ERHSV(20),ERHSV(20),VPROP(IPROP,JPROP),SHAFUN(20),WEIGP(30),
* XGAUS(20),YGAUS(20),ZGAUS(20),ELXYZT(20,3),P(20),
* DERG(20,3),DERSHX(20),DERSHY(20),DERSHZ(20),
* TVISC(400),DERSXI,DERSYI,DERSZI,DYNVIS,
* DERSXJ,DERSYJ,DERSZJ,SHAPI,RADIUS(20),DTRVC(20)

```

```

*****
*** Define the element properties required for the ESM *
*****

```

```

DYNVIS = VPROP(1,LPOP)
RHOIN = VPROP(2,LPOP)
BFORX = VPROP(3,LPOP)
BFORY = VPROP(4,LPOP)
BFORZ = VPROP(5,LPOP)
LCOUNT = VPROP(9,LPOP)
VSCHM = VPROP(10,LPOP)
PSCHM = VPROP(11,LPOP)
VISLAM = DYNVIS/RHOIN
NSPL = VPROP(12,LPOP)

```

```

*
*****
*** Assign the nodes and their coordinates for the current *
*** element. Modify the coordinates for the midside nodes. *
*** specified.

```

```

*****
*
* IF (NITERA.EQ.1) THEN
* CALL MIDSIDE(COORD,LNOMAX,LNODS,LNODZ,LTYPE,NEL,NELZ,
1 ELXYZT,NDIM,NODMAX,12)

```

```

*
* ENDIF
*

```

```

*****
*
* Initialization of the right hand side vector and *
* the element stiffness matrix *
*

```

```

*****
*
*** Initialize the right hand side vector to zero.
*
      DO 10 INODZ = 1,LNODZ
        ERHSU(INODZ)= 0.0
        ERHSV(INODZ)= 0.0
        ERHSV(INODZ)=0.0
      10  CONTINUE
*
*** Initialize the coefficients of element stiffness matrix to zero
*
      DO 20 IEVAB = 1,LVABZ
        DO 20 JEVAB = 1,LVABZ
          ESTIFM(IEVAB,JEVAB) = 0.0
        20  CONTINUE
*
*****
*
*   Enter the loop for the volume integration using
*   Gaussian scheme
*
*****
*
*** Evaluate the weights and distances for the Gaussian integration
*
      CALL GAUS3D(VBSCHM,NGAUS,XGAUS,YGAUS,ZGAUS,WEIGP)
      DO 100 IGAUS = 1,NGAUS
*
*** Assign the values of the local coordinates of the current
*   Gauss point
*
      XITA (1) = XGAUS(IGAUS)
      XITA (2) = YGAUS(IGAUS)
      XITA (3) = ZGAUS(IGAUS)
*
*** Evaluate the shape functions , their derivatives, the determinant
*** of the Jacobian etc.
*****
*** Calculate the shape functions and their derivatives for :
*** a 20-noded brick using linear interpolation for the
*** corner nodes and parabolic interpolation for the
*** mid-side nodes using subroutine HISEAP, which involves
*** hierarchical formulation.
*****
      CALL BRICKH(LNODZ,LVABZ,LNOMAX,WEL,WELZ,XITA,P,
1      ELXYZI,DERG,VOLUME)
      DVOLUM = VOLUME*WEIGP(IGAUS)
*
*** Assign the values of the shape functions and its derivatives*
*** to the shape functions and their derivatives at the current*
*** Gauss point.
*
      DO 30 INODZ = 1,LNODZ

```

```

      SHAFUN(INODZ) = P (INODZ)
      DERSHX(INODZ) = DERG (INODZ,1)
      DERSHY(INODZ) = DERG (INODZ,2)
      DERSHZ(INODZ) = DERG (INODZ,3)
30      CONTINUE
*****
*      Impose the initial conditions for the problem      *
*****
*
*** Check for the type of the flow. In case of the laminar and
*** inviscid flows eliminate the turbulent terms.
*
      IF(LCOUNT.LT.2) THEN
*
*** Initialize the three components of velocity i.e., U,V and W,
*** for the laminar and inviscid flows.
*
      UVELY = 0.0
      VVELY = 0.0
      WVELY = 0.0
*****
*** Evaluate the previous velocities for the laminar and the *
*** inviscid flows.
*****
      DO 35 INODZ = 1, LNODZ
      KPOIN = IABS(LNODS (INODZ,WEL))
*      ITOTU = NOPP(KPOIN)

      UVELY = UVELY+(COORD(KPOIN,5)*SHAFUN(INODZ))
      VVELY = VVELY+(COORD(KPOIN,6)*SHAFUN(INODZ))
      WVELY = WVELY+(COORD(KPOIN,7)*SHAFUN(INODZ))
35      CONTINUE
*
      ELSE
*      Call a subroutine to compute the turbulent viscosity values at
*      the nodal points of this element ex; TURBPIPE
      IF(NITERA.EQ.1) THEN
      UMAX = 0.9387
      ELSE
      UMAX = COORD(1,5)
      ENDIF
      CALL TURBPIPE(COORD,LNODS,LNODZ,WEL,TVISC,VISLAM,WELZ,NOD
1MAX,NDIM,UMAX,RADIUS,DRTVC)
*
*** Initialize the three components of velocity i.e., U,V and W,
*** the coefficient of turbulent viscosity and their derivatives.
*** for the turbulent flow.
*
      UVELY = 0.0
      VVELY = 0.0
      WVELY = 0.0
      TURVIS = 0.0
      TURVDX = 0.0
      TURVDY = 0.0
      TURVDZ = 0.0

```

```
*****
*** Evaluate the previous velocities, the turbulent viscosity, *
*** and its derivatives at the current Gauss point for the *
*** turbulent flow.
```

```
*****
```

```
DO 40 INODZ = 1, LNODZ
  KPOIN = IABS(LNODS (INODZ, NEL))
  ITOTU = NOPP(KPOIN)
  UVELY = UVELY + (COORD(KPOIN, 5) * SHAFUN(INODZ))
  VVELY = VVELY + (COORD(KPOIN, 6) * SHAFUN(INODZ))
  WVELY = WVELY + (COORD(KPOIN, 7) * SHAFUN(INODZ))
  TURVIS = TURVIS + (TVISC(INODZ) * SHAFUN(INODZ))
  TURVDX = TURVDX + (TVISC(INODZ) * DERSHX(INODZ))
  TURVDY = TURVDY + (TVISC(INODZ) * DERSHY(INODZ))
  TURVDZ = TURVDZ + (TVISC(INODZ) * DERSHZ(INODZ))
40 CONTINUE
  ENDIF
```

```
*
*** Include the body forces on the right hand side
```

```
DO 55 INODZ = 1, LNODZ
  ERHSU(INODZ) = ERHSU(INODZ) + SHAFUN(INODZ) * DVOLUM * BFORX
  ERHSV(INODZ) = ERHSV(INODZ) + SHAFUN(INODZ) * DVOLUM * BFORY
  ERHSW(INODZ) = ERHSW(INODZ) + SHAFUN(INODZ) * DVOLUM * BFORZ
55 CONTINUE
  WRITE(UNIT=3, REC=NEL+1, NELZ) LNODZ, ERHSU, ERHSV, ERHSW
```

```
*****
*** Start the loop over the element for the left hand side terms*
*****
```

```
DO 70 ICON = 1, 20
```

```
*
*** Counter for the number of row in the matrix has started. There*
*** are four rows for each of the corner nodes and three for the *
*** midside nodes.
```

```
*
  IF (ICON .LE. 8) THEN
    IROWU = (4 * ICON) - 3
    IROWV = IROWU + 1
    IROWW = IROWU + 2
    IROWP = IROWU + 3
  ELSE
    IROWU = (3 * ICON) + 6
    IROWV = IROWU + 1
    IROWW = IROWU + 2
  ENDIF
```



```

*** Evaluate the 'i'th shape function and its derivatives for the *
*** 'i'th row.
*
      SHAPI = SHAFUN(ICON)
      DERSXI = DERSHX(ICON)
      DERSYI = DERSHY(ICON)
      DERSZI = DERSHZ(ICON)
*
*** Counter for the number of column in the matrix has started.*
*** There are four columns for the corner nodes and three for *
*** the midside nodes.
*
      DO 80 JCON = 1,20
*
      IF(JCON.LE.8) THEN
        JCOLU = (4*JCON)-3
        JCOLV = JCOLU+1
        JCOLW = JCOLU+2
        JCOLP = JCOLU+3
      ELSE
        JCOLU = (JCON*3)+6
        JCOLV = JCOLU+1
        JCOLW = JCOLU+2
      ENDIF
*
*** Evaluate the 'j'th shape function and its derivatives for the*
*** 'j'th column.
*
      DERSXJ = DERSHX(JCON)
      DERSYJ = DERSHY(JCON)
      DERSZJ = DERSHZ(JCON)
*****
*** Check for the type of the flow. In case of laminar flow exclude *
*** the turbulent terms. In the case of the inviscid flow exclude *
*** the turbulent and the diffusion terms in the momentum equations.*
*****in170.051
*
      IF(LCOUNT.EQ.0) THEN
        DIFSON = 0.0
      ELSEIF(LCOUNT.EQ.1) THEN
        DIFSON=(DERSXI*DERSXJ+DERSYI*DERSYJ+DERSZI*DERSZJ)*VISLAM*DVOLUM
      ENDIF
      IF(LCOUNT.LT.2) THEN
        CONVTH = (UVELY*DERSXJ+VVELY*DERSYJ+WVELY*DERSZJ)*SHAPI*DVOLUM
*
*** Include the diffusion and convection terms in the momentum *
*** equation.
*
      ESTIFM(IROWU,JCOLU) = ESTIFM(IROWU,JCOLU)+DIFSON+CONVTH
      ESTIFM(IROWV,JCOLV) = ESTIFM(IROWV,JCOLV)+DIFSON+CONVTH
      ESTIFM(IROWW,JCOLW) = ESTIFM(IROWW,JCOLW)+DIFSON+CONVTH
*
      ELSE

```

```

*****
*** Evaluate the additional terms in the momentum equations *
*** because of the Reynolds stress terms for the turbulent flow. *
*****
      RNST1 = SHAPI*DVOLUM*((TURVDX*2.*DERSXJ)+(TURVDY*DERSY
1J*1.)+(TURVDZ*DERSZJ*1.))
      RNST2 = SHAPI*DERSXJ*TURVDY*DVOLUM
      RNST3 = SHAPI*DERSXJ*TURVDZ*DVOLUM
      RNST4 = SHAPI*DERSYJ*TURVDX*DVOLUM
      RNST5 = SHAPI*DVOLUM*((TURVDX*DERSXJ*1.)+(TURVDY*2.*DE
1RSYJ)+(TURVDZ*DERSZJ*1.))
      RNST6 = SHAPI*DERSYJ*TURVDZ*DVOLUM
      RNST7 = SHAPI*DERSZJ*TURVDX*DVOLUM
      RNST8 = SHAPI*DERSZJ*TURVDY*DVOLUM
      RNST9 = SHAPI*DVOLUM*((TURVDX*DERSXJ*1.)+(TURVDY*DERSY
1J*1.)+(TURVDZ*DERSZJ*2.))
*
*** Include these additional terms in the element stiffness matrix*
*
      ESTIFM(IROWU,JCOLU) = ESTIFM(IROWU,JCOLU)-RNST1
      ESTIFM(IROWU,JCOLV) = ESTIFM(IROWU,JCOLV)-RNST2
      ESTIFM(IROWU,JCOLW) = ESTIFM(IROWU,JCOLW)-RNST3
      ESTIFM(IROWV,JCOLU) = ESTIFM(IROWV,JCOLU)-RNST4
      ESTIFM(IROWV,JCOLV) = ESTIFM(IROWV,JCOLV)-RNST5
      ESTIFM(IROWV,JCOLW) = ESTIFM(IROWV,JCOLW)-RNST6
      ESTIFM(IROWW,JCOLU) = ESTIFM(IROWW,JCOLU)-RNST7
      ESTIFM(IROWW,JCOLV) = ESTIFM(IROWW,JCOLV)-RNST8
      ESTIFM(IROWW,JCOLW) = ESTIFM(IROWW,JCOLW)-RNST9
*****
*** Evaluate the diffusion and convection terms in the momentum*
*** equations for the turbulent flow.
*****
      DIFSON=(DERSXI*DERSXJ+DERSYI*DERSYJ+DERSZI*DERSZJ)*(TURVIS)
1*DVOLUM
      CONVTN = (UVELY*DERSXJ+VVELY*DERSYJ+WVELY*DERSZJ)*SHAPI*DVOLUM
*
      ENDIF
*
*** Include the diffusion and convection terms in the momentum *
*** equations for all types of flow.
*
      ESTIFM(IROWU,JCOLU) = ESTIFM(IROWU,JCOLU)+DIFSON+CONVTN
      ESTIFM(IROWV,JCOLV) = ESTIFM(IROWV,JCOLV)+DIFSON+CONVTN
      ESTIFM(IROWW,JCOLW) = ESTIFM(IROWW,JCOLW)+DIFSON+CONVTN
*
*** Include the pressure terms in the momentum equation *
      ENDIF
* End the loop for the 'j'th column
*
80 CONTINUE
*
*** Call a separate subroutine to make accelerations in the X dire
*** ction zero
*
*** End the loop for the 'i'th row

```

```

*
70  CONTINUE
*****
***  End the loop for the gauss point integration  *
*****
100  CONTINUE
*
***  Different integration scheme is to be used for the
***  pressure terms in the momentum equations.
*
    CALL GAUS3D(PSCHEM,NGAUS,XGAUS,YGAUS,ZGAUS,WEIGP)
    DO 140   IGAUS = 1,NGAUS
*
***  Assign the values of the local coordinates of the current
*  Gauss point
*
    XITA (1) = XGAUS(IGAUS)
    XITA (2) = YGAUS(IGAUS)
    XITA (3) = ZGAUS(IGAUS)
*
    CALL BRICKH(LNODZ,LVABZ,LNOMAX,WEL,WELZ,XITA,P,
1      ELXZYI,DERG,VOLUME)
      DVOLUM = VOLUME*WEIGP(IGAUS)
      ENDIF
*
***  Assign the values of the shape functions and its derivatives
***  to the shape functions and their derivatives at the current
***  Gauss point.
*
    DO 150 INODZ = 1,LNODZ
      SHAFUN(INODZ) = P (INODZ)
      DERSHX(INODZ) = DERG (INODZ,1)
      DERSHY(INODZ) = DERG (INODZ,2)
      DERSHZ(INODZ) = DERG (INODZ,3)
150    CONTINUE
*

```

```

***      Start the looping over rows and columns for the pressure terms
***      in the momentum equations. Assign the shape functions and
***      their derivatives for the pressure terms.
*
      DO 160 IPCON = 1,20
*
***      Counter for the number of row in the matrix has started. There*
***      are four rows for each of the corner nodes and three for the *
***      midside nodes.
*
      IF(IPCON.LE.8) THEN
        IROWU = (4*IPCON)-3
        IROWV = IROWU+1
        IROWW = IROWU+2
        IROWP = IROWU+3
      ELSE
        IROWU = (3*IPCON)+6
        IROWV = IROWU+1
        IROWW = IROWU+2
      ENDIF
*
***      Evaluate the 'i'th shape function for the *
***      'i'th row.
*
      SHAPI = SHAFUN(IPCON)
*
***      Counter for the number of column in the matrix has started.*
***      There are four columns for the corner nodes and three for *
***      the midside nodes.
*
      DO 170 JPCON = 1,20
*
      IF(JPCON.LE.8) THEN
        JCOLU = (4*JPCON)-3
        JCOLV = JCOLU+1
        JCOLW = JCOLU+2
        JCOLP = JCOLU+3
      ELSE
        JCOLU = (JPCON+3)+6
        JCOLV = JCOLU+1
        JCOLW = JCOLU+2
      ENDIF
*
***      Evaluate the 'j'th shape function and its derivatives for the*
***      'j'th column.
*
      DERSXJ = DERSHX(JPCON)
      DERSYJ = DERSHY(JPCON)
      DERSZJ = DERSHZ(JPCON)
*
***      Include the pressure terms in the momentum equation
*
      IF(JPCON.LE.8) THEN
        ESTIFM(IROWU,JCOLP)=ESTIFM(IROWU,JCOLP)+SHAPI*DERSXJ*DVOLUM/RHOIN

```

```

ESTIFM(IROWV,JCOLP)=ESTIFM(IROWV,JCOLP)+SHAPI*DERSYJ*DVOLUM/RHOIN
ESTIFM(IROWV,JCOLP)=ESTIFM(IROWV,JCOLP)+SHAPI*DERSZJ*DVOLUM/RHOIN
ENDIF
*****
***      Form the continuity equation      *
*****
      IF(IPCON.LE.8) THEN
        ESTIFM(IROWP,JCOLU)=ESTIFM(IROWP,JCOLU)+SHAPI*DVOLUM*DERSXJ
        ESTIFM(IROWP,JCOLV)=ESTIFM(IROWP,JCOLV)+SHAPI*DVOLUM*DERSYJ
        ESTIFM(IROWP,JCOLW)=ESTIFM(IROWP,JCOLW)+SHAPI*DVOLUM*DERSZJ
      ENDIF
*      End the loop for the 'j'th column
*
170  CONTINUE
*
***      End the loop for the 'i'th row
*
180  CONTINUE
*****
***      End the loop for the gauss point integration      *
*****
140  CONTINUE
*
      IF(NSPL.EQ.1) THEN
        CALL NOXAXL(ESTIFM,LVABZ)
      ELSEIF (NSPL.EQ.2) THEN
        CALL RAGHAVENDER(ESTIFM,WEL,LVABZ)
      ENDIF
      WRITE(UNIT=1,REC=WEL)LVABZ,ESTIFM
      WRITE(UNIT=3,REC=WEL+WELZ) LNODEZ,ERHSU,ERHSV,ERHSV
      RETURN
C    STOP
    END
*
*****
*
*
      SUBROUTINE BRICKH(LNODEZ,LVABZ,LNOMAX,WEL,WELZ,XITA,
1        SHAPE,ELXYZ,DERIV,VOL)
*
*  THIS SUBROUTINE BRIC20 COMPUTES THE JACOBIAN MATRIX OF THE
*  TRANSFORMATION, ITS DETERMINANT WHICH IS THE VOLUME OF THE
*  TRANSFORMATION, THE INVERSE OF THIS MATRIX, AND FINALLY THE
*  SHAPE FUNCTIONS FOR THIS ELEMENT (A 20-NODE BRIC) IN BOTH
*  LOCAL AND GLOBAL COORDINATES.
*
*  IMPLICIT LOGICAL (A-Z)
*
*  Declarations for the parameters
  INTEGER
1  LNODEZ, LNOMAX, LVABZ, WEL, WELZ, OERR
  REAL
1  XITA(3)
*
*  Declarations for the local variables or those in COMMON blocks

```

```

REAL
1  DERIV(3,LNODZ), ELXYZ(LNODZ,3),J(3,3), JINV(3,3),
2  POIN(3), SHAPE(LNODZ), VOL
*
*   Initialize error flag
*   OERR=0
*
*   IF (LNODZ.EQ.20) THEN
*
*       CALL SFRH(XITA,SHAPE,DERIV)
*   ELSE
*
*       CALL SFBR8(XITA,SHAPE,DERIV)
*
*   ENDIF
*
*   THE SUBROUTINE SFBR20 WILL GET THE MATRIX W20, WHOSE FIRST ROW
*   CONTAINS THE SHAPE FUNCTIONS, AND THE SECOND THROUGH FOURTH THE
*   DERIVATIVES, AT THE POINT XITA IN LOCAL COORDINATES.
*
*   CALL JACOB(DERIV,ELXYZ,J,LNODZ)
*
*   NOW THE JACOBIAN, WHICH IS THE PRODUCT OF ELXYZ AND THE LAST THREE
*   ROWS OF W20, AND WHICH IS USED TO TRANSFORM FROM LOCAL TO GLOBAL
*   COORDINATES, IS FOUND.
*
*   CALL INVJ(J,JINV,VOL,OERR)
*   IF (OERR.EQ.1) THEN
*       WRITE (6,9020)
*       RETURN
*   ENDIF
*
*   If error flag is equal to 1 we have a neg or zero vol from INVJ
*
*   THE JACOBIAN IS INVERTED, AND ITS DETERMINANT, WHICH IS THE VOLUME
*   OF THE TRANSFORMATION, IS FOUND.
*
*   CALL CARTE (JINV, DERIV,LNODZ)
*
*   NEXT, THE CARTESIAN DERIVATIVES ARE FOUND.
*
*   CALL POINTS(POIN,SHAPE,ELXYZ,LNODZ)

```

```

* LASTLY, THE ACTUAL GLOBAL COORDINATES OF THE GIVEN POINT XITA
* ARE FOUND AS THE MATRIX PRODUCT OF ELXYZ AND THE FIRST ROW OF
* WB20.
*
  IF ((1.2.LT.ABS(XITA(1))) .OR. (1.2.LT.ABS(XITA(2)))
1    .OR. (1.2.LT.ABS(XITA(3)))) THEN
    WRITE(6,9000)
  ENDIF
*
  RETURN
9000 FORMAT('O','PROGRAM HAS ATTEMPTED TO FIND THE SHAPE FUNCTION',
1         ' OF A POINT OUTSIDE THE BRICK.')
9020 FORMAT('O','THE DETERMINANT OF THE JACOBIAN FOR THIS ',
1         ' ELEMENT IS NEGATIVE OR ZERO')
*
* WE SHOULD ONLY BE EVALUATING THE SHAPE FUNCTION INSIDE THE
* 20-NODE BRICK: IN OTHER WORDS, THE LOCAL COORDINATES OF THE
* GIVEN POINT (WHICH ARE XITA(1),XITA(2),XITA(3)) MUST ALL BE BETWEEN
* -1 AND +1.
*
  END
  SUBROUTINE SFRH(XITA,SHAPE,DERIV)
* NAME:          SFRH
*
* PURPOSE:       To compute the shape functions and their derivatives
*                for a 20 noded hierarchical brick element.
* DATE:          3 rd October 1983
  DIMENSION XITA(3),SHAPE(20),DERIV(20,3)
  R=XITA(1)
  S=XITA(2)
  T=XITA(3)
  R1=(1.-R)/2.
*  NN=20
  S1=(1.-S)/2.
  T1=(1.-T)/2.
  R2=1.-R*R
  S2=1.-S*S
  T2=1.-T*T
  R3=(1.+R)/2.
  S3=(1.+S)/2.
  T3=(1.+T)/2.
*
*** Calculate the shape functions for the current element
*
  SHAPE( 1)=R1*S1*T1
  SHAPE( 2)=R3*S1*T1
  SHAPE( 3)=R3*S3*T1
  SHAPE( 4)=R1*S3*T1
  SHAPE( 5)=R1*S1*T3
  SHAPE( 6)=R3*S1*T3
  SHAPE( 7)=R3*S3*T3
  SHAPE( 8)=R1*S3*T3
  SHAPE( 9)=R2*S1*T1
  SHAPE(10)=R3*S2*T1
  SHAPE(11)=R2*S3*T1

```

```

SHAPE(12)=R1*S2*T1
SHAPE(13)=R1*S1*T2
SHAPE(14)=R3*S1*T2
SHAPE(15)=R3*S3*T2
SHAPE(16)=R1*S3*T2
SHAPE(17)=R2*S1*T3
SHAPE(18)=R3*S2*T3
SHAPE(19)=R2*S3*T3
SHAPE(20)=R1*S2*T3

```

```

*
*** Calculate the derivatives of the shape functions w.r.t. R
*

```

```

DERIV( 1,1)=-S1*T1/2.
DERIV( 2,1) = S1*T1/2.
DERIV( 3,1)= S3*T1/2.
DERIV( 4,1)=-S3*T1/2.
DERIV( 5,1)=-S1*T3/2.
DERIV( 6,1)= S1*T3/2.
DERIV( 7,1)= S3*T3/2.
DERIV( 8,1)=-S3*T3/2.
DERIV( 9,1)=-2.*R*S1*T1
DERIV(10,1)= S2*T1/2.
DERIV(11,1)=-2.*R*S3*T1
DERIV(12,1)=-S2*T1/2.
DERIV(13,1)=-S1*T2/2.
DERIV(14,1)= S1*T2/2.
DERIV(15,1)= S3*T2/2.
DERIV(16,1)=-S3*T2/2.
DERIV(17,1)=-2.*R*S1*T3
DERIV(18,1)= S2*T3/2.
DERIV(19,1)=-2.*R*S3*T3
DERIV(20,1)=-S2*T3/2.

```

```

*
*** Calculate the derivatives of the shape functions w.r.t. S
*

```

```

DERIV(1,2)=-R1*T1/2.
DERIV(2,2)=-R3*T1/2.
DERIV(3,2)=R3*T1/2.
DERIV(4,2)= R1*T1/2.
DERIV(5,2)=-R1*T3/2.
DERIV(6,2)=-R3*T3/2.
DERIV(7,2)= R3*T3/2.
DERIV(8,2)= R1*T3/2.
DERIV(9,2)=-R2*T1/2.
DERIV(10,2)=-2.*S*R3*T1
DERIV(11,2)= R2*T1/2.
DERIV(12,2)=-2.*S*R1*T1
DERIV(13,2)=-R1*T2/2.
DERIV(14,2)=-R3*T2/2.
DERIV(15,2)= R3*T2/2.
DERIV(16,2)= R1*T2/2.
DERIV(17,2)=-R2*T3/2.
DERIV(18,2)=-2.*S*R3*T3
DERIV(19,2)= R2*T3/2.
DERIV(20,2)=-2.*S*R1*T3

```


*** Calculate the derivatives of the shape functions w r t T

```

*
DERIV(1,3)=-R1*S1/2.
DERIV(2,3)=-R3*S1/2.
DERIV(3,3)=-R3*S3/2
DERIV(4,3)=-R1*S3/2
DERIV(5,3)= R1*S1/2.
DERIV(6,3)= R3*S1/2.
DERIV(7,3)= R3*S3/2.
DERIV(8,3)= R1*S3/2.
DERIV(9,3)=-R2*S1/2.
DERIV(10,3)=-R3*S2/2.
DERIV(11,3)=-R2*S3/2.
DERIV(12,3)=-R1*S2/2.
DERIV(13,3)=-2.*T*R1*S1
DERIV(14,3)=-2.0*R3*S1*T
DERIV(15,3)=-2.0*T*R3*S3
DERIV(16,3)=-2.*T*R1*S3
DERIV(17,3)= R2*S1/2.
DERIV(18,3)= R3*S2/2.
DERIV(19,3)= R2*S3/2.
DERIV(20,3)= R1*S2/2
* WRITE(6,*) (SHAPE(I),I=1,NN)
* WRITE(6,*) ((DERIV(I,J),J=1,NN),I=1,3)
RETURN
END

```

```

      SUBROUTINE MIDSIDE(COORD, LNOMAX, LNODS, LNODZ, LTYPE, NEL, NELZ,
1      ELXYZ, NDIM, NODMAX, MIDNOD)
* NAME: MIDSIDE
* DATE: 23rd October 1983
* PURPOSE :      To modify the coordinates of the nodes of a brick
*                element for the hierarchical formulation. It modi-
*                fies the coordinates of the midside nodes if given
*                and keeps them to be zero if they are not given
      IMPLICIT LOGICAL (A-Z)
      INTEGER
1      I, J, LNODZ, LNOMAX, LTYPE,
2      MIDNOD, MIDS20(3,12), ND, NDIM, NEL, NELZ, NOD,
3      NODE, NODMAX, LNODS(LNOMAX, NELZ)
      REAL
1      COORD(NODMAX, 7), ELXYZ(LNODZ, NDIM)
*
**** LIST OF CONNECTIVITY DATA FOR MIDSIDE NODES IN THE 20 NODE
* BRICK 8 NODE QUAD AND BEAM ARE A SUBSET.
*
      DATA MIDS20/ 9,1,2, 10,2,3, 11,3,4, 12,4,1,
1      13,1,5, 14,2,6, 15,3,7, 16,4,8,
2      17,5,6, 18,6,7, 19,7,8, 20,8,5/
*
**** WE COLLECT THE NODAL COORDINATES FOR THE ELEMENT IN ELXYZ(LNODZ, NDI
*
      DO 20 NOD=1, LNODZ
        NODE = IABS(LNODS(NOD, NEL))
        DO 10 ND = 1, NDIM
          ELXYZ(NOD, ND) = COORD(NODE, ND)
10      CONTINUE
20      CONTINUE
*
* CALCULATE THE CO-ORDINATE VALUES FOR THE EDGES OF THE
* 8 NODE QUAD AND THE 20 NODE BRICK FOR HIERARCHICAL FORMULATION
* IT MODIFIES THE COORDINATES FOR THE EDGES IF THEY ARE SPECIFIED
* AND MAKES THEM ZERO IF THE COORDINATES ARE NOT SPECIFIED.
*
      DO 40 I = 1, MIDNOD
        NODE = IABS(LNODS(MIDS20(1, I), NEL))
        DO 30 ND = 1, NDIM
          IF (ELXYZ(MIDS20(1, I), ND) .EQ. 5.5E+55) THEN
            ELXYZ(MIDS20(1, I), ND) = 0.0
            COORD(NODE, ND) = (ELXYZ(MIDS20(2, I), ND) + ELXYZ(MIDS20(3, I), ND)) * 0.5
          ELSE
            ELXYZ(MIDS20(1, I), ND) = ELXYZ(MIDS20(1, I), ND) - ((ELXYZ(MIDS20(2,
1      I), ND) + ELXYZ(MIDS20(3, I), ND)) / 2.)
          ENDIF
        COORD(NODE, ND) = ELXYZ(MIDS20(1, I), ND)
30      CONTINUE
40      CONTINUE
*
* CALCULATE THE CENTRE NODE CO-ORDINATES FOR THE 9 OR 17
* NODE QUAD IF REQUIRED

```

```

*
*
IF (LNODZ EQ 9 OR LNODZ EQ 17) THEN
*
DO 5 ND=1,NDIM
*
IF (LNODZ EQ 9 AND (ABS(ELXYZ(9,ND)-5.5E+55) LT 1E-5)) THEN
*
  Calculate the centre node co-ordinates for 9 node quad
*
  ELXYZ(9,ND)=- 25*(ELXYZ(1,ND)+ELXYZ(3,ND)+ELXYZ(5,ND)+ELXYZ(7,ND))
1      +.5*(ELXYZ(2,ND)+ELXYZ(4,ND)+ELXYZ(6,ND)+ELXYZ(8,ND))
*
  NODE=IABS(LNODS(9,NEL))
  COORD(NODE,ND)=ELXYZ(9,ND)
  ELSEIF (LNODZ EQ 17 AND (ABS(ELXYZ(17,ND)-5.5E+55) LT 1E-5)) THEN
*
  Calculate the centre node co-ordinates for 17 node quad
*
  ELXYZ(17,ND)=-.25*(ELXYZ(1,ND)+ELXYZ(5,ND)+ELXYZ(9,ND)+
1      ELXYZ(13,ND))+.5*(ELXYZ(3,ND)+ELXYZ(7,ND)+
2      ELXYZ(11,ND)+ELXYZ(15,ND))
  NODE=IABS(LNODS(17,NEL))
  COORD(NODE,ND)=ELXYZ(17,ND)
  ENDIF
5  CONTINUE
  ENDIF
*
**** ARE ALL NODE COORD. NOW ASSIGNED?
*
DO 140 NOD=1,LNODZ
  NODE = IABS(LNODS(NOD,NEL))
  DO 130 ND = 1,NDIM
*** REPLACE STOP BY CALL TO DOCTOR ON NEXT 2 LINES.
    IF (ELXYZ(NOD,ND) EQ 5.5E+55) THEN
      PRINT*, 'UNASSIGNED COORDINATE IN MIDSID'
      STOP
    ENDIF
130  CONTINUE
140  CONTINUE
  RETURN
*9000 FORMAT(' NODE',6X,'X',12X,'Y',12X,'Z',/)
*9010 FORMAT(1X,I6,3F13.7)
  END
  SUBROUTINE TURBPIPE(COORD,LNODS,LNODZ,NEL,TVISC,VISLAM,NELZ,NOD
1MAX, NDIM,UMAX,RADIUS,DREVC)
*
*
DATE: 3rd MAY 1984
*
*
Purpose: To compute the eddy-viscosity values for a turbulent fl-
*
*         ow through a circular duct. This subroutine calculates
*
*         the nodal values of the eddy-viscosity of an element.
*
*         This information has been derived from the experimental
*
*         results published by Laufer. A tenth degree polynomial
*
*         is fit through the data points. Frictional velocity is
*
*         related to the maximum axial velocity through the empiri

```

```

*      cal relation given by Laufer. This subroutine works for
*      only a Reynolds number of 5XE5. User has to change this
*      for other Reynolds numbers.
*
*      DIMENSION      LNODS(LNODZ,WELZ), COORD(NODMAX,7), TVISC(LNODZ),
0      RADIUS(20), DTRVC(20), MIDS20(3,12)
*      DATA MIDS20/ 9,1,2, 10,2,3, 11,3,4, 12,4,1,
1      13,1,5, 14,2,6, 15,3,7, 16,4,8,
2      17,5,6, 18,6,7, 19,7,8, 20,8,5/
*
*      Calculate the radius of each nodal point using the values of the
*      nodal coordinates.
*
      DO 100 I=1,LNODZ
        KPOIN = IABS(LNODS(I,WEL))
        RADIUS(I) = SQRT((COORD(KPOIN,2)**2)+(COORD(KPOIN,3)**2))
100    CONTINUE
*
      DIAM = 8.0
*
*      Evaluate the constants for the polynomial approximation of Lauf
*      er's data.
*
      A0 = -2.360865526E-5
      A1 = 0.3257851023
      A2 = -0.2251211365
      A3 = 4.414638125
      A4 = -22.15354353
      A5 = -13.85791086
      A6 = 270.8817342
      A7 = -673.9890752
      A8 = 773.8285017
      A9 = -438.3667828
      A10 = 99.21432054
*
*      Compute the actual values of the eddy-viscosity at the nodal poin
*      ts.
*
      DO 200 J=1,LNODZ
        X = 1. - (2.*RADIUS(J)/DIAM)
        Y = A0 + (A1*(X**1)) + (A2*(X**2)) + (A3*(X**3)) + (A4*(X**4)) + (A5*
1      1(X**5)) + (A6*(X**6)) + (A7*(X**7)) + (A8*(X**8)) + (A9*(X**9)) + (A10*(X**
2      10))
        Y1 = (A1) + (A2*2.*(X**1)) + (A3*3.*(X**2)) + (A4*4.*(X**3)) + (A5*5.*(X**4)
1      1) + (A6*6.*(X**5)) + (A7*7.*(X**6)) + (A8*8.*(X**7)) + (A9*9.*(X**8)) + (A10
2      2*10.*(X**9))
        TVISC(J) = VISLAM + ((Y*0.035*DIAM*UMAX)/(2.0))
        DTRVC(J) = -(Y1*0.035*UMAX)
200    CONTINUE
      DO 300 I=1,12
        TVISC(MIDS20(1,I)) = TVISC(MIDS20(1,1)) - (((TVISC(MIDS20(2,I))) + (T
1      1VISC(MIDS20(3,I))))*0.5)
300    CONTINUE
      RETURN
      END

```

```
SUBROUTINE NOXAXL(ESTIFM,LVABZ)
DIMENSION ESTIFM(68,68)
```

```
*
*** This subroutine essentially modifies the element stiffness matrix
*** to accomodate zero acceleration in X direction within this
*** particular element.
*
DO 100 IROWU = 1,LVABZ
ESTIFM(IROWU,1) = ESTIFM(IROWU,1)+ESTIFM(IROWU,6)
ESTIFM(IROWU,13) = ESTIFM(IROWU,13)+ESTIFM(IROWU,9)
ESTIFM(IROWU,17) = ESTIFM(IROWU,17)+ESTIFM(IROWU,21)
ESTIFM(IROWU,29) = ESTIFM(IROWU,29)+ESTIFM(IROWU,25)
ESTIFM(IROWU,42) = ESTIFM(IROWU,42)+ESTIFM(IROWU,36)
ESTIFM(IROWU,45) = ESTIFM(IROWU,45)+ESTIFM(IROWU,48)
ESTIFM(IROWU,54) = ESTIFM(IROWU,54)+ESTIFM(IROWU,51)
ESTIFM(IROWU,66) = ESTIFM(IROWU,66)+ESTIFM(IROWU,60)
100 CONTINUE
RETURN
END
```

END

1 2 0 6 • 8 6

FIN