Science and Engineering Research Council
Rutherford Appleton Laboratory
Chilton DIDCOT Oxon OX11 0QX
RAL-92-081

A Solution of the Navier-Stokes Equations Using a (υ,ν,ρ) Formulation

C Greenough

December 1992

A Solution of the Navier-Stokes Equations Using a (u,v,p) Formulation

C. Greenough

October 1992

Abstract

This report describes the solution of the Navier-Stokes equation in their (u,v,p) formulation using the NAG/SERC Finite Element Library. The implementation uses a simple iteration schemes to solve the nonlinear algebraic system resulting from the application of the Galerkin method to the governing equations. The program also demonstrates how more complicated system matrices can be formed using the generalised assemble routines. As a test problem the report uses channel flow over a cavity.

In the report the governing equations and their numerical approximation are discussed together with a detailed description of the Fortran 77 program implementing their solution.

Mathematical Software Group
Computational Modelling Division
Rutherford Appleton Laboratory
Chilton, Didcot
Oxfordshire OX11 0QX

Contents

1 Introduction

The solution of the Navier-Stokes equations are very important in many areas of science and engineering [1]. Although the solution of these equations for high Reynolds number is very difficult and requires specialised techniques, there are many applications where such sophistication is not required. This program aims to provide a starting point for the development of such finite element programs and to illustrate the solution of such coupled non-linear systems using the Finite Element Library [2, 3, 4].

The program illustrates how a simple point iteration scheme can be implemented using the Library and how more complicated system matrices can be constructed using the generalised assembly routines. The program also illustrates how the Library can be used to create a numerical formulation that combines different orders of approximation for the dependent variables.

2 Problem

A laminar flow over a square cavity is create and driven by the motion of the top boundary **AB**. It is required to calculate the velocity field and pressure distribution of a viscous fluid moving over a square cavity. The problem geometry, discretisation and boundary conditions are described in Figures 1 and 2.

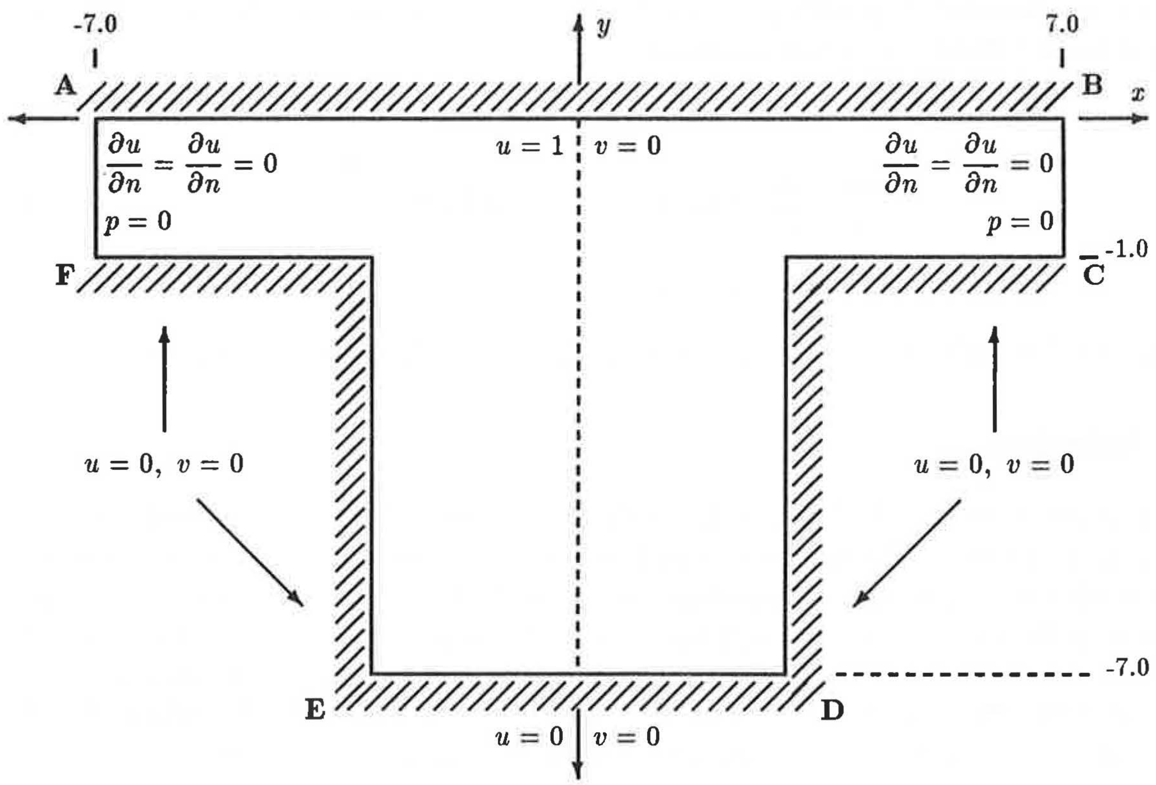

Figure 1: Problem Definition

3 Theory

The partial differential equations governing two-dimensional viscous flow in non-dimensional coordinates are:

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = \frac{l_{x^1}}{F_R} - \frac{\partial p}{\partial x} + \frac{1}{R_E}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{1}$$

$$u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = \frac{l_{x^2}}{F_R} - \frac{\partial p}{\partial y} + \frac{1}{R_E}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \tag{2}$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{3}$$

where u, v and p are the non-dimensional velocity components and pressure. R_E and F_R are the Reynolds and Froude numbers, and l_{x^1} and l_{x^2} are the direction cosines of the x and y global axes to the direction of the gravitational field. The non-dimensional variables are related to the dimensional variables by

$$x = \frac{x^*}{l},\ y = \frac{y^*}{l},\ u = \frac{u^*}{l},\ v = \frac{v^*}{l} \tag{4}$$

and

$$p = \frac{p^*}{\rho u_o^2} \tag{5}$$

where l is a characteristic length, ρ the density and u_o^2 a datum velocity. These equations are subject to the following boundary conditions

$$\begin{aligned} u = v = 0 &\quad \text{on FE, ED and DC} \\ \frac{\partial u}{\partial n} = \frac{\partial v}{\partial n} = p = 0 &\quad \text{on AF and BC} \\ u = u_o \text{ and } v = 0 &\quad \text{on AB} \end{aligned} \tag{6}$$

In this set of boundary conditions the flow is driven by the velocity of the top plate.

4 Solution

The problem domain is divided into 19 8-noded quadrilateral elements, numbered 1 to 19, as shown in Figure 2. Using the (u, v, p) formulation an accepted pratice is to represent the variation in the pressure by shape functions one order lower that those used to define the variation in the velocity components. Thus, the velocity components u and v are approximated by iso-parametric bi-quadratic elements and the pressure, p, by super-parametric bi-linear elements (since the geometry of the pressure elements will be defined by the eight nodes of the bi-quadratic elements). In each element the following approximations are made

$$u = \sum_{i=1}^{i=n} N_i u_i, \quad v = \sum_{i=1}^{i=n} N_i v_i \tag{7}$$

$$p = \sum_{j=1}^{j=m} M_j p_j \tag{8}$$

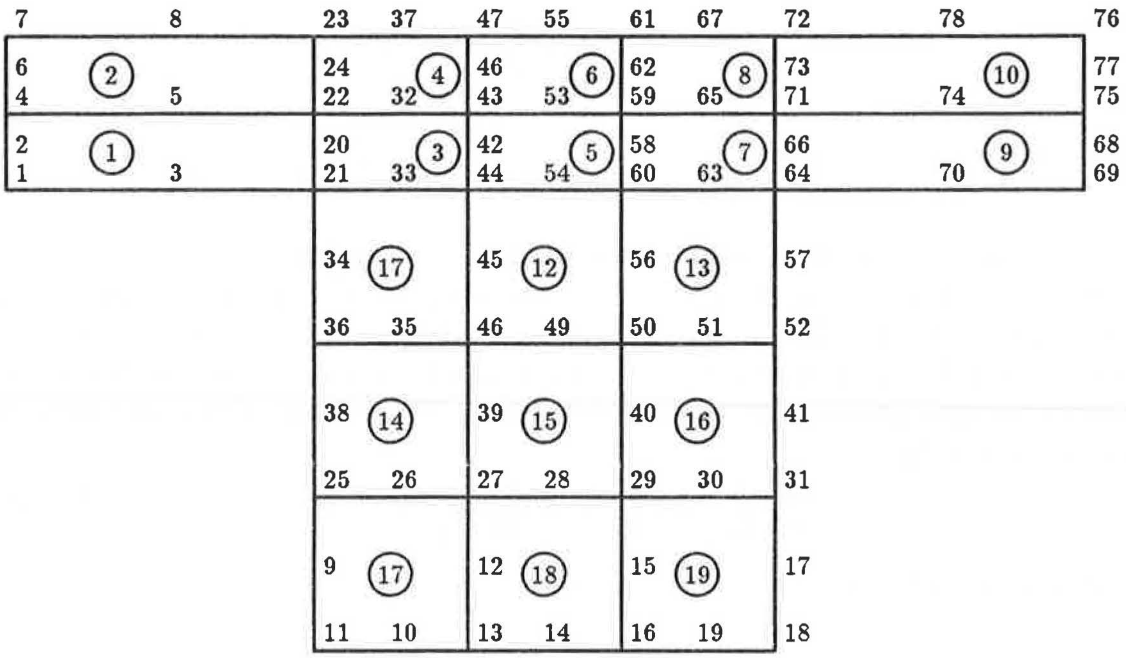

Figure 2: Problem Discretisation

where in this problem n=8 and m=4 and u_j, v_j and p_j are the nodel values of the velocity components and the pressure. Employing the Galerkin weighted residual approach (1) becomes

$$\sum_{1}^{n_e} \int_{\Omega^e} N_i \left[\sum_{k=1}^{n} N_k u_k \sum_{j=1}^{n} \frac{\partial N_j}{\partial x} u_j + \sum_{k=1}^{n} N_k v_k \sum_{j=1}^{n} \frac{\partial N_j}{\partial y} u_j \right.$$

$$\left. + \sum_{l=1}^{m} \frac{\partial M_l}{\partial x} p_l - \frac{l_{x1}}{F_R} - \frac{1}{R_E} \left[\sum_{1}^{n} \frac{\partial^2 N_j}{\partial x^2} u_j + \sum_{1}^{n} \frac{\partial^2 N_j}{\partial y^2} u_j \right] \right] d\Omega^e \quad = 0 \tag{9}$$

where the outer summation is over each element in the problem and the inner summations over the appropriate number of nodes in an element. By using Green's Theorem the second order terms can be reduced by an order. For example these terms in (9) become

$$\frac{1}{R_E} \int_{\Omega^e} N_i \left[\sum_{j=1}^{n} \frac{\partial^2 N_j}{\partial x^2} u_j + \sum_{j=1}^{n} \frac{\partial^2 N_j}{\partial y^2} u_j \right] d\Omega^e =$$

$$\frac{1}{R_E} \int_{\Gamma^e} N_i \sum_{j=1}^{n} \frac{\partial N_j}{\partial n} u_j \, d\Gamma^e - \frac{1}{R_E} \int_{\Omega^e} \left[\frac{\partial N_i}{\partial x} \sum_{j=1}^{n} \frac{\partial N_j}{\partial x} u_j + \frac{\partial N_i}{\partial y} \sum_{j=1}^{n} \frac{\partial N_j}{\partial y} u_j \right] d\Omega^e \tag{10}$$

where Γ_e denotes integration along an element boundary. Within the body of the domain a summation of these boundary integral contributions will vanish. The only non-zero contributions come from the boundary of the problem. Substitution of (10) into (9) produces

$$\sum_1^{n_e}\left[\int_{\Omega^e} N_i N_k u_k \frac{\partial N_j}{\partial x} + N_i N_k v_k \frac{\partial N_j}{\partial y} + N_i \frac{\partial M_l}{\partial x} p_l - N_i \frac{l_{x^1}}{F_R^2}\right.$$

$$\left.\frac{1}{R_E}\left[\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} u_j + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y} u_j\right]\right] d\Omega^e - \int_{\Gamma_1^e} \frac{1}{R_E} N_i \frac{\partial N_j}{\partial n} u_j d\Gamma - \int_{\Gamma_2^e} \frac{1}{R_E} N_i \frac{\partial u_j}{\partial n} d\Gamma^e = 0 \tag{11}$$

where Γ_1^e and Γ_2^e denote the boundaries over which $\partial u/\partial n$ is specified.

The corresponding approximation for (2) can be obtained by simply interchanging x with y and u with v in (11). The system of equations is completed by approximating the continuity equation (3). Again the Galerkin weighted residual approach is used, but this time the weighting functions are now taken to be those associated with the four-noded super-parametric pressure element, M_k.

$$\sum_1^{n_e}\int_{\Omega^e} M_i\left[\frac{\partial N_j}{\partial x} u_j + \frac{\partial N_j}{\partial y} v_j\right] d\Omega^e = 0 \tag{12}$$

The assembled matrix equation takes the form

$$A\lambda = F + B \tag{13}$$

where the form of λ is

$$\lambda = \begin{bmatrix} u_i \\ p_i \\ v_i \end{bmatrix} \tag{14}$$

Each coefficient in the matrix A has the form

$$a_{ij} = \sum_1^{n_e}\int_{\Omega^e}\begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33}\end{bmatrix} d\Omega^e - \int_{\Gamma^e}\begin{bmatrix} \frac{1}{R_E} N_i \frac{\partial N_j}{\partial n} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{R_E} N_i \frac{\partial N_j}{\partial n}\end{bmatrix} d\Gamma^e \tag{15}$$

where

$$C_{11} = N_i N_k \tilde{u}_k \frac{\partial N_j}{\partial x} + N_i N_k \tilde{v}_k \frac{\partial N_j}{\partial y} + \frac{1}{R_E}\left[\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right] \tag{16}$$

$$C_{12} = N_i \frac{\partial M_j}{\partial x},\ C_{13} = 0,\ C_{21} = M_i \frac{\partial N_j}{\partial x} \tag{17}$$

$$C_{22} = 0,\ C_{23} = M_i \frac{\partial N_j}{\partial y},\ C_{31} = 0 \tag{18}$$

$$C_{32} = N_i \frac{\partial M_j}{\partial y} \text{ and } C_{33} = C_{11} \tag{19}$$

where $\tilde{u}_k$ and $\tilde{v}_k$ are approximations of u and v that linearise the system during the solution process. These terms are discussed in Section 5.1.

The surface integrals in (16) correspond to that part of the element on a boundary where Dirichlet boundary conditions are applied. In general it is not necessary to calculate these terms.

Where natural or Neuman boundary conditions are imposed contributions will appear in the right-hand side vector. The basic right-hand side terms are of the form:

$$f_i = \sum_{1}^{n_e} \int_{\Omega^e} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} d\Omega^e \tag{20}$$

where

$$f_1 = N_i \frac{l_{x1}}{F_R^2},\ f_2 = 0,\ f_3 = N_i \frac{l_{x2}}{F_R^2} \tag{21}$$

Similarly, the terms in the right-hand side corresponding to natural boundary conditions are of the form:

$$b_i = \sum_{1}^{n_e} \int_{\Gamma_2^e} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} d\Gamma^e \tag{22}$$

where

$$b_1 = \frac{1}{R_E} N_i \left[\left(\frac{\partial u_j}{\partial n} \right)^{\Gamma^e} \right],\ b_2 = 0,\ b_3 = \frac{1}{R_E} N_i \left[\left(\frac{\partial v_j}{\partial n} \right)^{\Gamma^e} \right] \tag{23}$$

More details can be found in [1].

5 Numerical Methods

5.1 Iterative Procedure

The system of algebraic equations described in (13) are nonlinear. Consequently some form of linearisation is required to obtain a solution. In this example program a simple point iteration scheme is used to linearise the system. Consider the nonlinear component of the advection term in (1)

$$u \frac{\partial u}{\partial x} \tag{24}$$

The procedure is to assume a starting value or initial guess for the velocities u and v, $\tilde{u}$ and $\tilde{v}$. These initail values are assumed constant over each non-linear iteration and consequently (13) linearises. This linear system is assembled and solved for new values of u, v and p. A check is then made on the differece between, for example $\tilde{u}$ and u, and if they are are sufficiently close to iteration is complete. If there is a large difference between $\tilde{u}$ and u then new estimates are calculated using a weighted average of the current and previous values.

5.2 Boundary Conditions

In this problem Dirichlet boundary conditions on u and v are imposed on the walls of the cavity and on p in the inlet and outlet. There are two main ways in which these can be applied numerically.

One method is to delete from the assembled system matrix any entry corresponding to a boundary freedom and make the necessary ajustments to the right-hand side. This has the advantage of reducing the size of the problem although it becomes more difficult if the form of the boundary conditions is complex.

A second method is to bias entries in the system matrix corresponding to boundary freedoms. This can be done by either applying a large weight to the main diagonal entries of such nodes or by setting all the entries in that row to zero apart from the main diagonal entry, which is set to unity. In each case the right-hand side is updated with the boundary value with or without the weighting factor. This procedure forces the required boundary value.

6 Program Description

The programs in Release 4 of the NAG/SERC Finite Element Library have been writen in Fortran 77. No assumptions have been made concerning machine-dependent memory initialisation. Readers might find it helpful to read the Finite Element Library Introduction [3] before studying this section.

The program can be divide into seven major sections:

1. Delcarations and parameter specifications (statements 1 – 13).
2. Variable initialisation and data input (statements 14 – 69).
3. Main iteration loop (statements 70 – 147).
4. Assembly of element matrices (statements 79 – 113).
5. System matrix assembly (statements 114 – 119).
6. Inclusion of boundary conditions (statements 121 – 133).
7. Solution for velocity components u and v and pressure p (statement 135).
8. Update of iteration loop (statements 136 – 147).
9. Output of final results (statements 148 – 163).

In the description of the program, the names of the Fortran 77 variables and subroutines will be given in capitals (e.g. `ELTOP`).

6.1 Delcarations and parameter specifications

All Fortran 77 variables in the program are explicity declared. These declarations are broken into four groups: basic variable definitions, problem size dependent definitions, scalar variables and array variables. The values of the basic variables are set to accommodate the choice of elements required for the example problem. All problem size dependent arrays have had their sizes specified in seperate `PARAMETER` statements.

6.2 Variable initialisation and data input

In this section of the program all the basic model information is read in from the input stream. This included: phyiscal propertries (statements 15 – 18), nodal geometry (statements 19 – 27), element topologies (statements 28 – 35) and boundary conditions (statements 36 –53). During the input of the boundary condition data it is transformed into non-dimensional values using the reference density and velocity. In statement 54 the main iteration control information is read.

The special routine **NFSET** initialises the nodel freedom array **NF** with information about the mixed interpolation. This is used in setting up the final form of **NF** in statements 56 to 65.

The semi-bandwidth of the system matrix is calculated using the routine **BNDWTH** and the initialisation is completed with setting the solution vectors to zeros.

6.3 Main iteration loop

The main iteration loop comprises the body of the program. Statement 70 initialises the iteration counter to zero. The setup for each iteration is in statements 70 to 74. The iteration counter is incremented and the system matrix initialised to zero.

The main loop ends with the test for convergence in statement 147. The other elements of the main loop are discussed in the following sections.

6.4 Assembly of element matrices

The main assembly loop of the program starts at statement 78. This loops over all elements in the mesh. Initially the number of nodes in each element type (**NODELP** for pressure and **NODELV** for velocities) and the numerical integration quadrature are set in statements 75 to 77.

In this formulation the number of freedoms at each node of an element is variable. At the corners of each element u, v and p are freedoms. At the midside nodes only u and v are freedoms. One could view the solution as being formed by the assembly of 8-noded bi-quatratic elements for velocities and a 4-noded bi-linear element for pressure. The program calculates and assembles element matrix contributions for both the velocities and the pressure. Because the geometric desciption of the elements is based on the coordinates of the eight nodes the velocity elements are said to be isoparametric and the pressure elements are super-parametric.

To direct the assembly of these element matrices three steering vectors are constructed in statements 79 to 81 for the current element using calls to the routine **DIRGEN**. These steering vectors are **STRU** and **STRV** for the velocity components and **STRP** for the pressure.

During the assembly process an averaged velocity component is required by the linearisation. The routine **SELECT**, using the velocity steering vectors and the current solution vector **SOLO**, constructs the element velocity vectors **UELM** and **VELM** in statements 82 and 83. These can then be used to calculate the component average.

In this implementation five element matrices are constructed. These relate to the contributions of each dependent variable and their coupling. The identifiers used for these matrices correspond to those used in (16). In statements 85 to 89 **C11, C12, C21, C23** and **C32** are initialised.

The numerical integration loop starts at statement 90. The same order of integration is used for all element types. Statements 91 and 92 assign the current quadrature point abscii and statement 93 and 94 calculate the values of the shape functions (**FUNP** and **FUNV**) at these points.

Statements 95 to 98 construct the transformation Jacobian and the derivatives of the shape function with respect to the global coordinates x and y. The derivatives are held in the arrays **GDV** and **GDP**. The fix values of velocity are calculated at the current quadrature points in statements 99 and 100.

The final construction of the element matrix contributions occupies statements 101 to 112 and the numerical integration loop completed at statement 113.

The assembly of these element contributions is performed in statement 114 to 119. As can be seen from (17) to (19) there are only six non-zero terms in the full element matrix. Statements 114 and 119 assemble `C11` and `C33` using the standard assembly routine for unsymmetric systems `ASUSM`. As the other element matrices are not square the generalised assembly routine `AUSMG` must be used. This routine using two steering vectors to direct the assembly. The main assembly loop is completed by statement 120.

6.5 Inclusion of boundary conditions

The boundary conditions in this implementation are applied by modifying the rows in the system matrix corresponding to boundary freedoms. Each row is set to zero and the diagonal entry set to unity. The right hand side vector `SOLO` is set to the required boundary value. Statement 123 to 133 perform this process once the current solution vector has been saved in `SOL1` and the new solution vector (`SOLO`) initialised to zero.

6.6 Solution for velocity components u and v and pressure p

Statement 135 solves the system of linear equation resulting from the assembly process. The program performs a Gaussian decomposition followed by a forward and backward subsitution.

`GAUSOL`

On return the nodal values of the velocity components u and v and the pressure p are stored in the vector `SOLO`.

6.7 Update of iteration loop

Once the new values of u, v and p have been calculated the current values in the iteration can be updated and a new estimate of the error formed.

The maximum error in the new solution values is calculated in statements 136 to 142 and the current values of u, v and p are updated in the loop starting on statement 143. Statement 147 tests the error and performs a new iteration if necessary.

6.8 Output of final results

Statements 148 to 162 output the final solution once the iteration process has converged to the specified tolerance. As all the calculations are performed in non-dimensional variables the final solution values must be converted to their dimensional form. After conversion the values are output using the routine `PRTVAL` in statement 162.

7 Listings

This section contains a complete listing of the example program followed by the example data, example results and a list of important variables. On each statement of the data the corresponding variable names are given. Throughout this section the programs and results refer to a `DOUBLE PRECISION` implementation of the Library.

7.1 Program

```
C***********************************************************************
C
C     Segment 5.4 - Navier-Stokes Solution using velocity/pressure
C                   formulation on mixed elements
C
C     Copyright (C) 1992 : SERC, Rutherford Appleton Laboratory
C                          Chilton, DIDCOT, Oxfordshire OX11 0QX
C     ..
C     .. Parameters ..
C
1         INTEGER IABSS,IC11,IFUNV,IUELM,IGDV,IGEOM,IJAC,IJACIN,ILDV,ISTRU,
     *            IVELM,IWGHT,JABSS,JCOORD,JC11,JGDV,JGEOM,JJAC,JJACIN,JLDV,
     *            JNF
2         INTEGER ILDP,JLDP,IFUNP,IGDP,JGDP,IC12,JC12,IC21,JC21,IC23,JC23,
     *            IC32,JC32,ISTRV,ISTRP
3         PARAMETER (IABSS=3,IC11=8,IFUNV=8,IUELM=8,IGDV=3,IGEOM=8,IJAC=3,
     *              IJACIN=3,ILDV=3,ISTRU=8,IVELM=8,IWGHT=9,JABSS=9,
     *              JCOORD=3,JC11=8,JGDV=8,JGEOM=3,JJAC=3,JJACIN=3,JLDV=8,
     *              JNF=3,ILDP=3,JLDP=4,IFUNP=4,IGDP=3,JGDP=4,IC12=8,JC12=4,
     *              IC21=4,JC21=8,IC23=4,JC23=8,IC32=8,JC32=4,ISTRV=8,
     *              ISTRP=8)
C     ..
C     .. Parameters (Problem Size) ..
C
4         INTEGER IBFRE,IBNODE,IBVAL,ICOORD,IELTOP,INF,ISOL0,ISYSK,JELTOP,
     *            JSYSK,ILOWER,JLOWER,IROPIV,ISOL1,DOFNOD
5         PARAMETER (IBFRE=150,IBNODE=150,IBVAL=150,ICOORD=300,IELTOP=300,
     *              INF=300,ISOL0=1000,ISYSK=1000,JELTOP=10,JSYSK=250,
     *              ILOWER=1000,JLOWER=150,IROPIV=1000,ISOL1=1000,
     *              DOFNOD=3)
6         INTEGER NIN,NOUT
7         PARAMETER (NIN=5,NOUT=6)
C     ..
C     .. Local Scalars ..
C
8         DOUBLE PRECISION CHRLEN,CHRVEL,DENSTY,DET,ETA,MOLVSC,QUOT,REYNLD,
     *                     THETA,TOL,TOLMAX,UBAR,VALTOL,VBAR,XI
9         INTEGER BNDNOD,DIMEN,ELNUM,ELTYP,GLBFRE,HBAND,I,IQUAD,ITERMX,
     *            ITEST,J,K,LOCFRE,NELE,NITER,NODE,NODEL,NODELP,NODELV,
     *            NODNUM,NQP,TOTDOF,TOTELS,TOTNOD
C     ..
C     .. Local Arrays ..
C
10        DOUBLE PRECISION ABSS(3,9),BVAL(IBVAL),C11(8,8),C12(8,4),
     *                     C21(4,8),C23(4,8),C32(8,4),
     *                     COORD(ICOORD,JCOORD),FUNP(4),FUNV(8),
     *                     GDP(3,4),GDV(3,8),GEOM(8,3),JAC(3,3),
     *                     JACIN(3,3),LDP(3,4),LDV(3,8),
     *                     LOWER(ILOWER,JLOWER),SOL0(ISOL0),SOL1(ISOL1),
```

```
     *                 SYSK(ISYSK,JSYSK),UELM(8),VELM(8),WGHT(9)
      INTEGER BFRE(IBFRE),BNODE(IBNODE),ELTOP(IELTOP,JELTOP),
     *        NF(INF,JNF),ROPIV(IROPIV),STRP(8),STRU(8),STRV(8)
C     ..
C     .. External Subroutines ..
C
      EXTERNAL ASUSM,ASUSMG,BNDWTH,DIRGEN,ELGEOM,GAUSOL,MATINV,
     *         MATMUL,MATNUL,NFSET,PRTVAL,QQUA9,QUAM4,QUAM8,SCAPRD,
     *         SELECT,VECCOP,VECNUL
C     ..
C     .. Intrinsic Functions ..
C
      INTRINSIC ABS,MAX
C     ..
      ITEST = 0
C
C     **********************
C     *                    *
C     * Input Data Section *
C     *                    *
C     **********************
C
C
C     Flow characteristics
C
      READ (NIN,9810) DENSTY,CHRVEL,CHRLEN,MOLVSC
      WRITE (NOUT,9990) DENSTY,CHRVEL,CHRLEN,MOLVSC
      REYNLD = CHRVEL*CHRLEN*DENSTY/MOLVSC
      WRITE (NOUT,9980) REYNLD
C
C     Input of nodal geometry
C
      READ (NIN,9830) TOTNOD,DIMEN
      WRITE (NOUT,9970) TOTNOD,DIMEN
      DO 1010 I = 1,TOTNOD
         READ (NIN,9820) NODNUM, (COORD(NODNUM,J),J=1,DIMEN)
         WRITE (NOUT,9800) NODNUM, (COORD(NODNUM,J),J=1,DIMEN)
         DO 1000 J = 1,DIMEN
            COORD(NODNUM,J) = COORD(NODNUM,J)/CHRLEN
 1000    CONTINUE
 1010 CONTINUE
C
C     Input of element topology
C
      READ (NIN,9830) TOTELS
      WRITE (NOUT,9960) TOTELS
      DO 1020 I = 1,TOTELS
         READ (NIN,9830) ELNUM,ELTYP,NODEL,
     *      (ELTOP(ELNUM,J+2),J=1,NODEL)
         WRITE (NOUT,9950) ELNUM,ELTYP,NODEL,
     *      (ELTOP(ELNUM,J+2),J=1,NODEL)
```

```
            ELTOP(ELNUM,1) = ELTYP
            ELTOP(ELNUM,2) = NODEL
 1020 CONTINUE
C
C     Input of number of degrees of freedom per node, input of
C     boundary conditions and construction of nodal freedom array NF
C
      WRITE (NOUT,9940) DOFNOD
      READ (NIN,9830) BNDNOD
      WRITE (NOUT,9930) BNDNOD
      IF (BNDNOD.NE.0) THEN
         DO 1050 I = 1,BNDNOD
            READ (NIN,9910) BNODE(I),BFRE(I),BVAL(I)
            WRITE (NOUT,9910) BNODE(I),BFRE(I),BVAL(I)
            J = BFRE(I)
            GO TO (1040,1030,1040) J
            WRITE (NOUT,9920) I,BNODE(I),BFRE(I),BVAL(I)
            STOP
 1030       CONTINUE
            BVAL(I) = BVAL(I)/(CHRVEL**2*DENSTY)
            GO TO 1050
 1040       CONTINUE
            BVAL(I) = BVAL(I)/CHRVEL
 1050    CONTINUE
      END IF
C
      READ (NIN,9900) ITERMX,TOL,THETA
C
      CALL NFSET(TOTELS,ELTOP,IELTOP,JELTOP,TOTNOD,NF,INF,JNF,DOFNOD,
     *           ITEST)
      TOTDOF = 0
      DO 1070 I = 1,TOTNOD
         DO 1060 J = 1,DOFNOD
            IF (NF(I,J).NE.0) THEN
               TOTDOF = TOTDOF + 1
               NF(I,J) = TOTDOF
            END IF
 1060    CONTINUE
 1070 CONTINUE
      WRITE (NOUT,9890) TOTDOF
C
C     Calculation of semi-bandwidth HBAND
C
      CALL BNDWTH(ELTOP,IELTOP,JELTOP,NF,INF,JNF,DOFNOD,TOTELS,HBAND,
     *            ITEST)
      WRITE (NOUT,9880) HBAND
C
C     ************************************
C     *                                  *
C     * System Stiffness Matrix Assembly *
C     *                                  *
```

```
C     ************************************
C
      CALL VECNUL(SOLO,ISOLO,TOTDOF,ITEST)
      CALL VECNUL(SOL1,ISOL1,TOTDOF,ITEST)
      NITER = 0
 1080 CONTINUE
      NITER = NITER + 1
      WRITE (NOUT,9870) NITER
      CALL MATNUL(SYSK,ISYSK,JSYSK,TOTDOF,2*HBAND-1,ITEST)
C
C     Start main assembly loop
C
      NODELP = 4
      NODELV = 8
      CALL QQUA9(WGHT,IWGHT,ABSS,IABSS,JABSS,NQP,ITEST)
C
      DO 1130 NELE = 1,TOTELS
C
C     Setup steering vectors to guide assembly
C
         CALL DIRGEN(NELE,ELTOP,IELTOP,JELTOP,NF,INF,JNF,1,1,1,1,STRU,
     *               ISTRU,ITEST)
         CALL DIRGEN(NELE,ELTOP,IELTOP,JELTOP,NF,INF,JNF,1,2,1,2,STRP,
     *               ISTRP,ITEST)
         CALL DIRGEN(NELE,ELTOP,IELTOP,JELTOP,NF,INF,JNF,1,3,1,1,STRV,
     *               ISTRV,ITEST)
C
C     Set up element velocity vectors
C
         CALL SELECT(SOLO,ISOLO,STRU,ISTRU,NODELV,UELM,IUELM,ITEST)
         CALL SELECT(SOLO,ISOLO,STRV,ISTRV,NODELV,VELM,IVELM,ITEST)
C
C     Set up element geometry array
C
         CALL ELGEOM(NELE,ELTOP,IELTOP,JELTOP,COORD,ICOORD,JCOORD,GEOM,
     *               IGEOM,JGEOM,DIMEN,ITEST)
C
C     Integration loop for element stiffness using NQP quadrature
C     points
C
         CALL MATNUL(C11,IC11,JC11,NODELV,NODELV,ITEST)
         CALL MATNUL(C12,IC12,JC12,NODELV,NODELP,ITEST)
         CALL MATNUL(C21,IC21,JC21,NODELP,NODELV,ITEST)
         CALL MATNUL(C23,IC23,JC23,NODELP,NODELV,ITEST)
         CALL MATNUL(C32,IC32,JC32,NODELV,NODELP,ITEST)
C
         DO 1120 IQUAD = 1,NQP
C
C     Form linear shape function and space derivatives in the local
C     corrdinates. Transform local derivatives to global coordinate
C     system
```

```
C
             XI = ABSS(1,IQUAD)
             ETA = ABSS(2,IQUAD)
C
             CALL QUAM4(FUNP,IFUNP,LDP,ILDP,JLDP,XI,ETA,ITEST)
             CALL QUAM8(FUNV,IFUNV,LDV,ILDV,JLDV,XI,ETA,ITEST)
C
C     Form transformation jacobian
C
             CALL MATMUL(LDV,ILDV,JLDV,GEOM,IGEOM,JGEOM,JAC,IJAC,JJAC,
     *                   DIMEN,NODELV,DIMEN,ITEST)
             CALL MATINV(JAC,IJAC,JJAC,JACIN,IJACIN,JJACIN,DIMEN,DET,
     *                   ITEST)
C
C     Form global derivatives of FUNV and FUNP
C
             CALL MATMUL(JACIN,IJACIN,JJACIN,LDV,ILDV,JLDV,GDV,IGDV,JGDV,
     *                   DIMEN,DIMEN,NODELV,ITEST)
             CALL MATMUL(JACIN,IJACIN,JJACIN,LDP,ILDP,JLDP,GDP,IGDP,JGDP,
     *                   DIMEN,DIMEN,NODELP,ITEST)
C
C     Calculate UBAR and VBAR
C
             CALL SCAPRD(UELM,IUELM,FUNV,IFUNV,NODELV,UBAR,ITEST)
             CALL SCAPRD(VELM,IVELM,FUNV,IFUNV,NODELV,VBAR,ITEST)
C
C     Formation of element matrices
C
             QUOT = ABS(DET)*WGHT(IQUAD)
             DO 1110 I = 1,NODELV
                DO 1090 J = 1,NODELV
                   C11(I,J) = (GDV(1,I)*GDV(1,J)+GDV(2,I)*GDV(2,J))/
     *                        REYNLD*QUOT + C11(I,J) +
     *                        (GDV(1,J)*UBAR+GDV(2,J)*VBAR)*FUNV(I)*QUOT
 1090           CONTINUE
                DO 1100 J = 1,NODELP
                   C12(I,J) = FUNV(I)*GDP(1,J)*QUOT + C12(I,J)
                   C21(J,I) = FUNP(J)*GDV(1,I)*QUOT + C21(J,I)
                   C23(J,I) = FUNP(J)*GDV(2,I)*QUOT + C23(J,I)
                   C32(I,J) = FUNV(I)*GDP(2,J)*QUOT + C32(I,J)
 1100           CONTINUE
 1110        CONTINUE
 1120     CONTINUE
C
C     Assembly of system stiffness matrix
C
          CALL ASUSM(SYSK,ISYSK,JSYSK,C11,IC11,JC11,STRU,ISTRU,HBAND,
     *               NODELV,ITEST)
          CALL ASUSMG(SYSK,ISYSK,JSYSK,C12,IC12,JC12,STRU,ISTRU,STRP,
     *                ISTRP,HBAND,NODELV,NODELP,ITEST)
          CALL ASUSMG(SYSK,ISYSK,JSYSK,C21,IC21,JC21,STRP,ISTRP,STRU,
```

```
     *                ISTRU,HBAND,NODELP,NODELV,ITEST)
            CALL ASUSMG(SYSK,ISYSK,JSYSK,C23,IC23,JC23,STRP,ISTRP,STRV,
     *                ISTRV,HBAND,NODELP,NODELV,ITEST)
            CALL ASUSMG(SYSK,ISYSK,JSYSK,C32,IC32,JC32,STRV,ISTRV,STRP,
     *                ISTRP,HBAND,NODELV,NODELP,ITEST)
            CALL ASUSM(SYSK,ISYSK,JSYSK,C11,IC11,JC11,STRV,ISTRV,HBAND,
     *               NODELV,ITEST)
 1130 CONTINUE
C
C     *********************
C     *                   *
C     * Equation Solution *
C     *                   *
C     *********************
C
C     Modification of stiffness matrix and right-hand side to
C     implement boundary conditions
C
      CALL VECCOP(SOLO,ISOLO,SOL1,ISOL1,TOTDOF,ITEST)
      CALL VECNUL(SOLO,ISOLO,TOTDOF,ITEST)
      K = 2*HBAND - 1
      DO 1150 I = 1,BNDNOD
         NODE = BNODE(I)
         LOCFRE = BFRE(I)
         GLBFRE = NF(NODE,LOCFRE)
         DO 1140 J = 1,K
            SYSK(GLBFRE,J) = 0.0D0
 1140    CONTINUE
         SYSK(GLBFRE,HBAND) = 1.0D0
         SOLO(GLBFRE) = BVAL(I)
 1150 CONTINUE
C
C     Solution of system matrix for the nodal values of the
C     potential
C
      WRITE (NOUT,9860)
      CALL GAUSOL(SYSK,ISYSK,JSYSK,LOWER,ILOWER,JLOWER,TOTDOF,HBAND,
     *            ROPIV,IROPIV,SOLO,ISOLO,ITEST)
C
      TOLMAX = 0.0D0
      DO 1160 I = 1,TOTDOF
         IF (ABS(SOLO(I)).GT.TOL) THEN
            VALTOL = ABS(SOLO(I)-SOL1(I))/SOLO(I)
            TOLMAX = MAX(TOLMAX,VALTOL)
         END IF
 1160 CONTINUE
C
      DO 1170 I = 1,TOTDOF
         SOLO(I) = (1.0D0-THETA)*SOL1(I) + SOLO(I)*THETA
 1170 CONTINUE
      WRITE (NOUT,9850) TOLMAX
```

```
      IF ((TOLMAX.GT.TOL) .AND. (NITER.NE.ITERMX)) GO TO 1080
C
      DO 1210 I = 1,TOTNOD
         DO 1200 J = 1,DOFNOD
            K = NF(I,J)
            IF (K.NE.0) THEN
               GO TO (1190,1180,1190) J
 1180          CONTINUE
               SOLO(K) = SOLO(K)*DENSTY*CHRVEL**2
               GO TO 1200
 1190          CONTINUE
               SOLO(K) = SOLO(K)*CHRVEL
            END IF
 1200    CONTINUE
 1210 CONTINUE
C
      WRITE (NOUT,9840)
      CALL PRTVAL(SOLO,ISOLO,NF,INF,JNF,DOFNOD,TOTNOD,NOUT,ITEST)
      STOP
C
 9990 FORMAT (' Density                   =',D12.5,
     *       /' Characteristic Velocity =',D12.5,
     *       /' Characteristic Length   =',D12.5,
     *       /' Molecular Viscosity     =',D12.5)
 9980 FORMAT (/' Reynolds Number           =',D12.5)
 9970 FORMAT (/' Node in mesh  =',I5,'  Dimensions =',I5)
 9960 FORMAT (' Elements in mesh =',I5)
 9950 FORMAT (' ',15I5)
 9940 FORMAT (' Maximum number of freedoms per node =',I5)
 9930 FORMAT (' Number of boundary conditions =',I5)
 9920 FORMAT (' *** ERROR - Invalid Freedom Number',
     *       /,' Boundary Condition Number ',I2,': Values : ',3I5)
 9910 FORMAT (2I5,2F10.5)
 9900 FORMAT (I5,2F10.0)
 9890 FORMAT (' Total number of unknowns (TOTDOF) = ',I5)
 9880 FORMAT (' Semi-Bandwidth (HBAND) = ',I5)
 9870 FORMAT (' Iteration ',I5,/,' System Assembly Started')
 9860 FORMAT (' Solution Started')
 9850 FORMAT (' Maximum Error = ',D12.5)
 9840 FORMAT (//' Final Results',/)
 9830 FORMAT (16I5)
 9820 FORMAT (I5,6F10.0)
 9810 FORMAT (5F10.0)
 9800 FORMAT (' ',I5,6F10.5)
      END
```

7.2 Data

To make data input as simple as possible standard FORMATs are used for all *real* and `INTEGER` data. These are `F10.0` for *reals* and `I5` for `INTEGERs`.

```
       1.0       1.0       6.0     0.05   DENSTY,CHRVEL,CHRLEN,MOLVSC
   78  2                                  TOTELS, DIMEN
    1  -7.0000   -1.0000
    2  -7.0000   -0.7500
    3  -5.0000   -1.0000
    4  -7.0000   -0.5000
    5  -5.0000   -0.5000
    6  -7.0000   -0.2500
    7  -7.0000    0.0000
    8  -5.0000    0.0000
    9  -3.0000   -6.0000
   10  -2.0000   -7.0000
   11  -3.0000   -7.0000
   12  -1.0000   -6.0000
   13  -1.0000   -7.0000
   14   0.0000   -7.0000
   15   1.0000   -6.0000
   16   1.0000   -7.0000
   17   3.0000   -6.0000
   18   3.0000   -7.0000
   19   2.0000   -7.0000
   20  -3.0000   -0.7500
   21  -3.0000   -1.0000
   22  -3.0000   -0.5000
   23  -3.0000    0.0000
   24  -3.0000   -0.2500
   25  -3.0000   -5.0000
   26  -2.0000   -5.0000
   27  -1.0000   -5.0000
   28   0.0000   -5.0000
   29   1.0000   -5.0000
   30   2.0000   -5.0000
   31   3.0000   -5.0000
   32  -2.0000   -0.5000
   33  -2.0000   -1.0000
   34  -3.0000   -2.0000
   35  -2.0000   -3.0000
   36  -3.0000   -3.0000
   37  -2.0000    0.0000
   38  -3.0000   -4.0000
   39  -1.0000   -4.0000
   40   1.0000   -4.0000
   41   3.0000   -4.0000
   42  -1.0000   -0.7500
   43  -1.0000   -0.5000
   44  -1.0000   -1.0000
```

```
45   -1.0000   -2.0000
46   -1.0000   -3.0000
47   -1.0000    0.0000
48   -1.0000   -0.2500
49    0.0000   -3.0000
50    1.0000   -3.0000
51    2.0000   -3.0000
52    3.0000   -3.0000
53   -0.0000   -0.5000
54    0.0000   -1.0000
55    0.0000    0.0000
56    1.0000   -2.0000
57    3.0000   -2.0000
58    1.0000   -0.7500
59    1.0000   -0.5000
60    1.0000   -1.0000
61    1.0000    0.0000
62    1.0000   -0.2500
63    2.0000   -1.0000
64    3.0000   -1.0000
65    2.0000   -0.5000
66    3.0000   -0.7500
67    2.0000    0.0000
68    7.0000   -0.7500
69    7.0000   -1.0000
70    5.0000   -1.0000
71    3.0000   -0.5000
72    3.0000    0.0000
73    3.0000   -0.2500
74    5.0000   -0.5000
75    7.0000   -0.5000
76    7.0000    0.0000
77    7.0000   -0.2500
78    5.0000    0.0000
19                                                         TOTELS
 1    2    8    1    2    4    5   22   20   21    3 ELNUM,ELTYP,NODEL,
 2    2    8    4    6    7    8   23   24   22    5 (ELTOP(ELNUM,J+2),
 3    2    8   21   20   22   32   43   42   44   33            J=1,NODEL)
 4    2    8   22   24   23   37   47   48   43   32
 5    2    8   44   42   43   53   59   58   60   54
 6    2    8   43   48   47   55   61   62   59   53
 7    2    8   60   58   59   65   71   66   64   63
 8    2    8   59   62   61   67   72   73   71   65
 9    2    8   64   66   71   74   75   68   69   70
10    2    8   71   73   72   78   76   77   75   74
11    2    8   21   33   44   45   46   35   36   34
12    2    8   44   54   60   56   50   49   46   45
13    2    8   60   63   64   57   52   51   50   56
14    2    8   36   35   46   39   27   26   25   38
15    2    8   46   49   50   40   29   28   27   39
16    2    8   50   51   52   41   31   30   29   40
```

```
17   2   8  25  26  27  12  13  10  11   9
18   2   8  27  28  29  15  16  14  13  12
19   2   8  29  30  31  17  18  19  16  15
74                                              BNDNOD
 1   1    0.0                                   BNODE,BFRE,BVAL
 1   3    0.0
 3   1    0.0
 3   3    0.0
21   1    0.0
21   3    0.0
34   1    0.0
34   3    0.0
36   1    0.0
36   3    0.0
38   1    0.0
38   3    0.0
25   1    0.0
25   3    0.0
 9   1    0.0
 9   3    0.0
11   1    0.0
11   3    0.0
10   1    0.0
10   3    0.0
13   1    0.0
13   3    0.0
14   1    0.0
14   3    0.0
16   1    0.0
16   3    0.0
19   1    0.0
19   3    0.0
18   1    0.0
18   3    0.0
17   1    0.0
17   3    0.0
31   1    0.0
31   3    0.0
41   1    0.0
41   3    0.0
52   1    0.0
52   3    0.0
57   1    0.0
57   3    0.0
64   1    0.0
64   3    0.0
70   1    0.0
70   3    0.0
69   1    0.0
69   3    0.0
 7   1    1.0
```

```
  7   3     0.0
  8   1     1.0
  8   3     0.0
 23   1     1.0
 23   3     0.0
 37   1     1.0
 37   3     0.0
 47   1     1.0
 47   3     0.0
 55   1     1.0
 55   3     0.0
 61   1     1.0
 61   3     0.0
 67   1     1.0
 67   3     0.0
 72   1     1.0
 72   3     0.0
 78   1     1.0
 78   3     0.0
 76   1     1.0
 76   3     0.0
  1   2     0.0
  4   2     0.0
  7   2     0.0
 69   2     0.0
 75   2     0.0
 76   2     0.0
100   0.01     0.5                                 ITERMX,TOL,THETA
```

7.3 Results

The results below were produced using the example data and the current Level 1 Program. Only a sample of the output is shown, together with a contour plot of the velocity vectors and stream function (the stream function values are not calculated by this program).

```
Density                    = 0.10000D+01
Characteristic Velocity = 0.10000D+01
Characteristic Length   = 0.60000D+01
Molecular Viscosity     = 0.10000D+01

Reynolds Number            = 0.12000D+03

Node in mesh  =   78   Dimensions =    2

     .........

Elements in mesh =   19

     .........

Maximum number of freedoms per node =    3
Number of boundary conditions =   74
```

```
Total number of unknowns (TOTDOF) =   186
Semi-Bandwidth (HBAND) =    65

    .........

Iteration     1
System Assembly Started
Solution Started
Maximum Error =  0.10000D+01
Iteration     2
System Assembly Started
Solution Started
Maximum Error =  0.15524D+01

    .........

Iteration    14
System Assembly Started
Solution Started
Maximum Error =  0.76905D-02

Final Results
```

NODE	VALUE	VALUE	VALUE
1	0.00000D+00	0.00000D+00	0.00000D+00
2	0.25991D+00	0.00000D+00	0.29986D-02
3	0.00000D+00	0.00000D+00	0.00000D+00
4	0.49958D+00	0.00000D+00	0.27634D-02
5	0.53200D+00	0.00000D+00	-0.30590D-02
6	0.75287D+00	0.00000D+00	0.18207D-02
7	0.99994D+00	0.00000D+00	0.00000D+00
8	0.99994D+00	0.00000D+00	0.00000D+00
9	0.00000D+00	0.00000D+00	0.00000D+00
10	0.00000D+00	0.00000D+00	0.00000D+00
11	0.00000D+00	0.10865D-01	0.00000D+00
12	-0.27338D-01	0.00000D+00	0.27786D-02
13	0.00000D+00	-0.16828D-01	0.00000D+00
14	0.00000D+00	0.00000D+00	0.00000D+00
15	-0.17045D-01	0.00000D+00	-0.77211D-02
16	0.00000D+00	0.84837D-02	0.00000D+00
17	0.00000D+00	0.00000D+00	0.00000D+00
18	0.00000D+00	-0.45774D-01	0.00000D+00
19	0.00000D+00	0.00000D+00	0.00000D+00
20	0.27146D+00	0.00000D+00	-0.14221D-01
21	0.00000D+00	-0.53100D-01	0.00000D+00
22	0.53369D+00	-0.45892D-01	-0.76370D-02
23	0.99994D+00	-0.48571D-01	0.00000D+00
24	0.77214D+00	0.00000D+00	-0.35541D-03
25	0.00000D+00	-0.13660D-01	0.00000D+00
26	-0.10509D-01	0.00000D+00	0.25919D-01
27	-0.45140D-01	-0.55701D-02	0.25422D-01
28	-0.67901D-01	0.00000D+00	0.99267D-02

29	-0.66523D-01	-0.88489D-02	-0.10494D-01
30	-0.37913D-01	0.00000D+00	-0.18497D-01
31	0.00000D+00	0.16805D-01	0.00000D+00
32	0.51114D+00	0.00000D+00	-0.19388D-03
33	0.13379D+00	0.00000D+00	0.56251D-01
34	0.00000D+00	0.00000D+00	0.00000D+00
35	-0.90939D-02	0.00000D+00	0.75278D-01
36	0.00000D+00	-0.32987D-02	0.00000D+00
37	0.99994D+00	0.00000D+00	0.00000D+00
38	0.00000D+00	0.00000D+00	0.00000D+00
39	-0.45715D-01	0.00000D+00	0.41723D-01
40	-0.40146D-01	0.00000D+00	-0.22854D-01
41	0.00000D+00	0.00000D+00	0.00000D+00
42	0.31455D+00	0.00000D+00	0.27806D-01
43	0.51293D+00	-0.11227D-01	0.20472D-01
44	0.18322D+00	-0.12943D-01	0.41204D-01
45	-0.30224D-01	0.00000D+00	0.89754D-01
46	-0.20817D-01	-0.19024D-01	0.48264D-01
47	0.99994D+00	-0.81872D-02	0.00000D+00
48	0.74473D+00	0.00000D+00	0.52941D-02
49	-0.43433D-01	0.00000D+00	0.30834D-01
50	-0.20280D-01	-0.49703D-02	-0.61226D-01
51	-0.30805D-01	0.00000D+00	-0.16022D+00
52	0.00000D+00	-0.32001D-01	0.00000D+00
53	0.58772D+00	0.00000D+00	-0.42579D-02
54	0.31965D+00	0.00000D+00	0.20201D-01
55	0.99994D+00	0.00000D+00	0.00000D+00
56	-0.16578D-01	0.00000D+00	-0.20727D-01
57	0.00000D+00	0.00000D+00	0.00000D+00
58	0.46323D+00	0.00000D+00	0.22629D-01
59	0.63008D+00	-0.15261D-01	-0.25176D-02
60	0.31508D+00	-0.34167D-01	0.67825D-01
61	0.99994D+00	-0.19215D-01	0.00000D+00
62	0.79526D+00	0.00000D+00	0.67828D-02
63	0.18252D+00	0.00000D+00	-0.11237D+00
64	0.00000D+00	0.13287D+00	0.00000D+00
65	0.53401D+00	0.00000D+00	-0.17270D-01
66	0.25143D+00	0.00000D+00	0.11667D-01
67	0.99994D+00	0.00000D+00	0.00000D+00
68	0.28104D+00	0.00000D+00	0.69757D-02
69	0.00000D+00	0.00000D+00	0.00000D+00
70	0.00000D+00	0.00000D+00	0.00000D+00
71	0.49662D+00	0.11679D+00	-0.72218D-02
72	0.99994D+00	0.11721D+00	0.00000D+00
73	0.74162D+00	0.00000D+00	-0.73198D-02
74	0.52999D+00	0.00000D+00	0.19918D-01
75	0.53667D+00	0.00000D+00	0.75986D-02
76	0.99994D+00	0.00000D+00	0.00000D+00
77	0.78631D+00	0.00000D+00	0.57626D-02
78	0.99994D+00	0.00000D+00	0.00000D+00

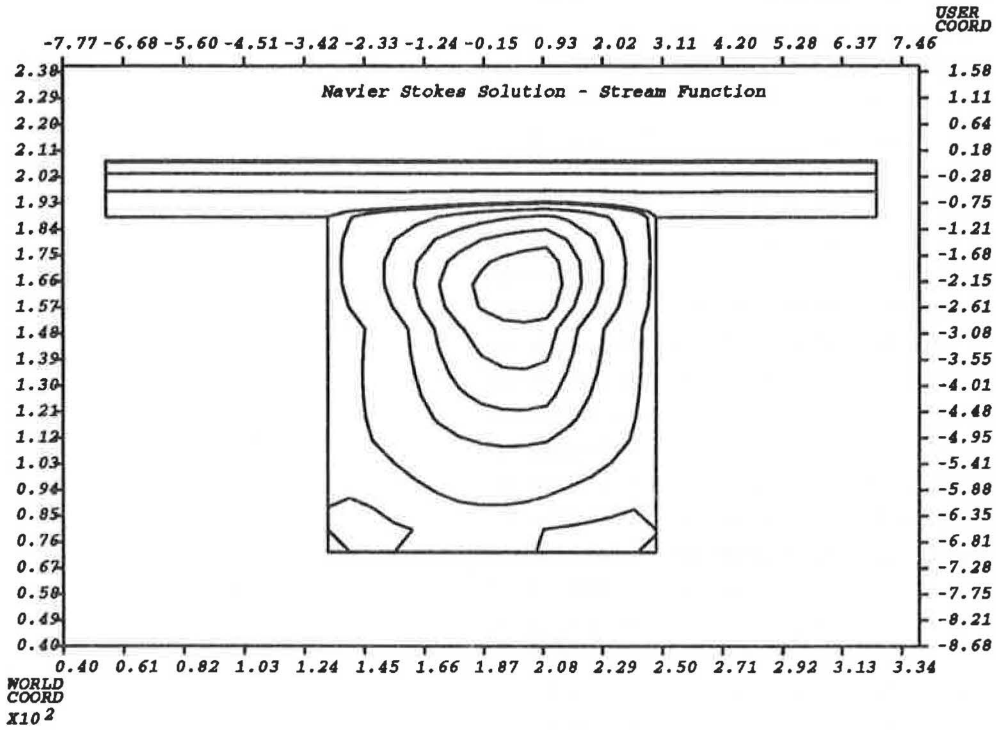

Figure 3: Contours of Stream Function

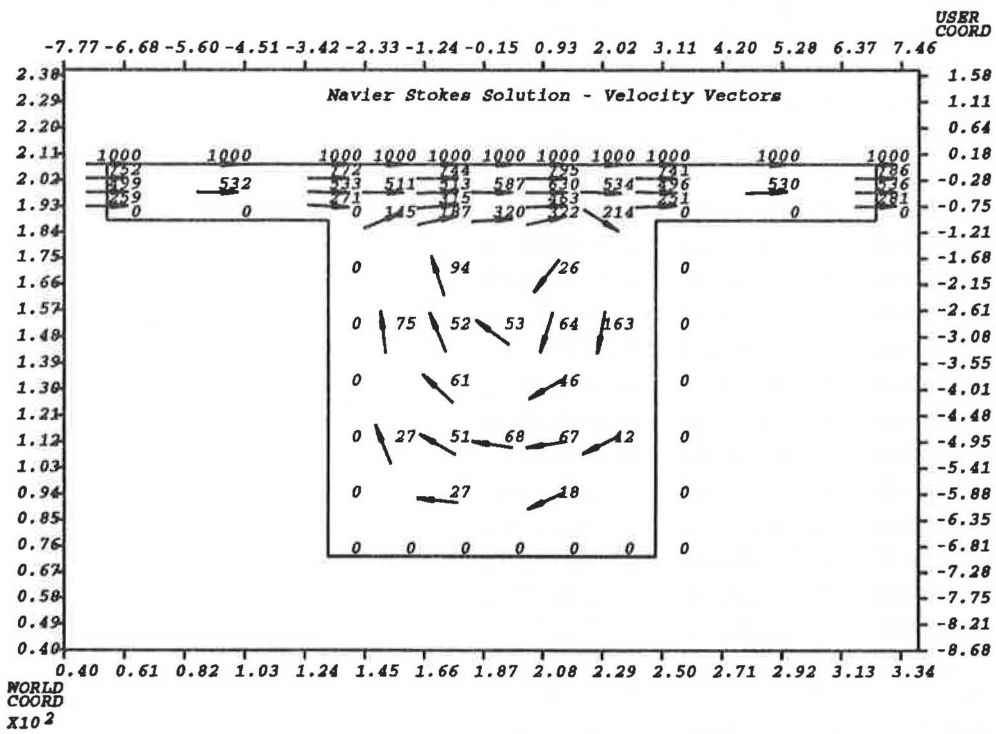

Figure 4: Plot of Velocity Vectors

7.4 List of important variables

In order to make the Level 1 Library programs as readable as possible, appropriate standard FORTRAN variable names have been used. A list of the most commonly used variables, together with an indication of their usage, is given below.

1. **INTEGER VARIABLES**

BFRE	Boundary node freedom numbers.
BNDNOD	Number of boundary nodes.
BNODE	Array containing list of boundary nodes.
DIMEN	Dimensionality of problem.
ELNUM	Element number.
ELTOP	Element topology array.
ELTYP	Type of element.
HBAND	Semi-bandwidth of system matrices.
ITERMX	Maximum number of nonlinear iterations.
ITEST	Error indicator and control.
NF	Nodal freedom array.
NITER	Maximum number of iterations.
NODEL	Number of nodes per element.
NODELP	Number of nodes on a pressure element.
NODELV	Number of nodes on a velocity element.
NODNUM	Node number.
NQP	Number of quadrature points.
ROPIV	Array containing pivoting history.
STRP	Pressure steering vector.
STRU	U-velocity steering vector.
STRV	V-velocity steering vector.
TOTDOF	Total number of degrees of freedom.
TOTELS	Total number of elements.
TOTNOD	Total number of nodes.

2. **REAL VARIABLES**

ABSS	Abscissae of quadrature points.
BVAL	Values of potential on boundary.
CHRLEN	Characteristic length in the problem.
CHRVEL	Characteristic velocity in the problem.
COORD	Global coordinates of nodes.
DENSTY	Fluid density.
ETA	Local coordinate variable.
FUNP	Shape function array for pressure.
FUNV	Shape function array for velocities.

GEOM	Local element topology array.
JAC	Transformation Jacobian.
JACIN	Jacobian inverse.
LOWER	Array containing lower triangle of reduced.
MOLVSC	Molecular viscosity of fluid.
REYNLD	Fluid Reynold's number.
SOL0	Solution vector containing old solution.
SOL1	Solution vector containg new solution.
SYSK	System stiffness matrix.
THETA	Scheme parameter.
TOLMAX	Solution tolerance.
WGHT	Quadrature weights.
XI	Local coordinate variable.

8 Conclusion

A numerical solution of the two-dimensional Navier-Stokes equations have been found using a mixed interpolation finite element method. The primitive variables u, v and p have been used and the resulting nonlinear algebraic system linearised using a simple point iteration scheme.

The numerical schemes have been implemented using the NAG/SERC Finite Element Library and a detailed description of the program given. Although this implementation is not efficient it provides a starting point for developing other finite element based approximations to the Navier-Stokes equations.

References

[1] Taylor, C. and Hughes, T.G., *Finite Element Programming of the Navier-Stokes Equations*, Pineridge Press Limited, Swansea, 1981

[2] Greenough, C. and Robinson, K., "Examples in the Use of the Finite Element Library: Steady State Potential Flow", Rutherford Appleton Laboratory Report, RL–82–060, (1982).

[3] Greenough, C. and Robinson, K., "The NAG/SERC Finite Element Library – Release 3 Level 1 User Documentation", Numerical Algorithms Group Ltd., Oxford, United Kingdom, (1990).

[4] Greenough, C. and Robinson, K., "The NAG/SERC Finite Element Library – Release 3 Level 0 User Documentation", Numerical Algorithms Group Ltd., Oxford, United Kingdom, (1990).

[5] Greenough, C. and Robinson, K., "Examples in the Use of the Finite Element Library: Steady State Potential Flow", Rutherford Appleton Laboratory Report, RL–82–060, (1982).

[6] Emson, C.R.I. and Greenough, C., "Examples in the Use of the Finite Element Library: Non-Linear Magnetostatic Solution", Rutherford Appleton Laboratory Report, RAL–84–105, (1984).