ACFAMP

Aircraft **C**artridge **F**ilter **A**nalysis **M**odelling **P**rogram

User's Manual

OVERVIEW – ABOUT ACFAMP

Aircraft Cartridge Filter Analysis Modelling Program (ACFAMP) is a Finite Element Based (FEM) FORTRAN computer code designed to simulate the hydrodynamics and design parameters for pleated filter cartridge fabrication. It allows the user to enter the cartridge filter specifications, filtration operating conditions such as inlet volumetric flow rates, physical properties of the fluids used in filter testing and the filter media characteristics in order to evaluate critical parameters such as optimal pressure drop. **ACFAMP** also quantifies the compression effects and apparent losses in cartridge filtration area, which are of prime importance in cartridge design, sizing and manufacture.

TECHNICAL SPECIFICATIONS

---

1. **ACFAMP** is written in FORTRAN 90.

2. **ACFAMP** is a 2 - dimensional Porous flow and combined Free/Porous flow simulation package. The solution scheme for **ACFAMP** is based on the **U-V-P** finite element method. Porous flow simulation is based on **Artificial Compressibility Method** whilst the combined flow simulator employs both the **Artificial Compressibility** and **Taylor-Hood** schemes.

3. **ACFAMP** solves the **DARCY** and **STOKES** momentum balance equations and the **CONTINUITY** equation for incompressible flow problems in the Cartesian co-ordinate system using the Weighted Residual Standard Galerkin Finite Element Method **(WRSGFEM).**

4. **ACFAMP** can program the solution of non-Newtonian and non-isothermal incompressible transient and steady flow problems.

5. **ACFAMP** has inbuilt mathematical permeability models (based on actual experimental test data) for different filter media.

6. **ACFAMP** can perform **"2-D hydrodynamic simulation for a pleated cartridge without any compression effects"** and also **"simulates compression factors and apparent losses in filtration area"** for a given filter cartridge.

7. Algebraic equations in **ACFAMP** are solved by a frontal method.

8. Post-processing files generated by **ACFAMP** can directly be exported to **SURFER, TECPLOT** and **COSMOS** for purposes of visualisation.

9. **AFCAMP** has 23 subroutines that are directly accessible to the user and can be easily modified depending on the modelling requirements.

SYSTEM REQUIREMENTS & INSTALLATION

---

1. System Requirements (Essential)

a. The recommended hardware for the system consists of a **Desktop PC** with a **Pentium** type processor having at least **32MB RAM**. However, we would specifically recommend a **Pentium - 4 (3.2 GHz)** processor with **512 MB RAM** for faster computations.

b. **Microsoft Windows 95, 98, ME, NT, XP or 2000** operating system.

c. **Compaq Visual Fortran 6.6** for **ACFAMP** compilation and execution.

2. System Requirements (Auxiliary)

a. Installed **COSMOS/M 2.6** for mesh generation and visualisation

b. Installed **TECHPLOT 8.0** for visualisation.

c. Installed **GOLDEN SOFTWARE SURFER** for visualisation.

3. Installation Procedure

a. Make a sub-directory for **ACFAMP** in the main **WINDOWS** directory and position the **ACFAMP** program with accompanying **Input data files** for comprehensive cartridge filter analysis in the sub-directory.

b. Right click on the **ACFAMP** program and open it with **COMPAQ VISUAL FORTRAN 6.6** compiler**.**

c. Go to the **COMPAQ VISUAL FORTRAN 6.6** main menu**,** click on **Build command Compile ACFAMP Execute ACFAMP**

GETTING STARTED

Working Step 1

Use the auxiliary FORTRAN program *FILTERdataprep.f* to prepare the information data file containing the geometrical characteristics and operational experimental data pertaining to a specific filter cartridge. It should be noted that approximate loss in area factors are required for performing simulations using ACFAMP and should be manually referred from the filter database PLEBASE.

Working Step 2

For the execution of ACFAMP three data files are the requisites

File created using the auxiliary program *FILTERdataprep.f*
GFM file created after the mesh generation step using COSMOS 2.6/M
Default file *df.dat* containing the elemental connectivity of the filter mesh

Once ACFAMP is executed it prompts the user to input the names of the three files and thereafter the number crunching starts.

Working Step 3

The simulated results are presented in three separate files which can be directly used for post-processing procedures,

Aerout.txt contains the data for finite element simulations in form of nodal velocities and pressures which can be visualised in COSMOS 2.6/M, TECHPLOT and SURFER

Excel.txt contains cartridge design parameters such as apparent losses in filtration area, medium compression and pressure gradients inclusive and exclusive of medium compression and loss in area effects.

PROGRAM SUBROUTINES - HEIRARCHY

SUBROUTINES, LEVEL 1

1.	AREALOSS	:	Called by **ACFAMP**
2.	CARTRIDGE	:	Called by **ACFAMP**
3.	EXCELPRINT	:	Called by **ACFAMP**
4.	GETELM	:	Called by **ACFAMP**
5.	GETMAT	:	Called by **ACFAMP**
6.	GETNOD	:	Called by **ACFAMP**
7.	GFMFEM	:	Called by **ACFAMP**
8.	HYDROFILTER	:	Called by **ACFAMP**
9.	PLEATBC	:	Called by **ACFAMP**

SUBROUTINES, LEVEL 2

1.	CLEAN	:	Called by HYDROFILTER
2.	CONTOL	:	Called by HYDROFILTER
3.	FLOW	:	Called by HYDROFILTER
4.	OUTPUT	:	Called by HYDROFILTER
5.	PUTBCV	:	Called by HYDROFILTER
6.	SECINV	:	Called by HYDROFILTER
7.	SETPRM	:	Called by HYDROFILTER

SUBROUTINES, LEVEL 3

1.	DERIV	:	Called by FLOW
2.	FRONT	:	Called by FLOW
3.	GAUSSP	:	Called by FLOW
4.	PERM	:	Called by FLOW
5.	SHAPE	:	Called by FLOW
6.	VISCA	:	Called by FLOW
7.	MINIMAX	:	Called by OUTPUT

SUBROUTINES, LEVEL 4

1.	BACSUB	:	Called by FRONT

Please note that all the subroutines at a particular level have been listed alphabetically

PROGRAM SUBROUTINES - FUNCTIONS

AREALOSS	Evaluates the loss in filtration area factor for velocity correction and the apparent percentage loss in filtration area at a given flow rate
BACSUB	Uses the Back substitution method for finding the final solution vector
CARTRIDGE	Reads the cartridge specifications such as number of pleats, height of the filter element, length of the pleat, diameter of the inner core, total filtration area, different inlet flow rates, viscosity of the fluid coordinates of the cartridge filter mesh
CLEAN	Cleans the arrays and prepares them for solution
CONTOL	Makes a check for the convergence of variables
DERIV	Calculates the Jacobean matrix, the determinant of the Jacobean matrix and global derivatives of the shape functions used in finite element calculations
EXCELPRINT	Prints the simulated pressure drop values with no compression and no loss in area, with compression and no loss in area, and with compression effects and loss in filtration area; percentage compression and apparent loss in filtering area at a given flow rate
FLOW	Calculates the velocities and pressures based on finite element method
FRONT	Employs the frontal method for solving the final set of equations
GAUSSP	Specifies the gauss points and weights for quadrature integration

PROGRAM SUBROUTINES - FUNCTIONS

GETELM	Specifies the nodal connectivity array for every element
GETMAT	Reads the material parameters
GETNOD	Reads the coordinates of the filter mesh from the MESH.FEM file
GFMFEM	Generates the mesh coordinates of the filter to be used in finite element simulations
HYDROFILTER	Simulates the 2-D hydrodynamic field in a filter without any compression and loss in area effect, with compression effects and with compression effects and apparent losses in filtration area
OUTPUT	Prints the final solution in form of 2 - dimensional velocities (x and y components) and the pressure field at different times
PERM	Calculates the permeability of the filter cartridge media used in analysis at different pressure drop values
PLEATBC	Generates the inlet velocity boundary conditions for filter cartridge simulation from the specified inlet volumetric flow rates
PUTBCV	Imposes the primary boundary conditions for velocity
SECINV	Calculates the second invariant of rate of deformation stress tensor, which depends on the velocity gradients for a particular time step

PROGRAM SUBROUTINES - FUNCTIONS

SETPRM	Sets the location data for nodal degrees of freedom that are the velocities and the pressures associated with each node in the cartridge mesh
SHAPE	Calculates the shape functions and their derivatives used in finite element calculations
VISCA	Calculates the viscosity according to the power law model valid for shear thickening hydraulic fluids used in aircraft filters

LIST OF PRIME VARIABLES - AFCAMP

---

aa:	Left hand side finite element stiffness matrix
actpress:	Normalised value of nodal pressure
alpha:	Time-stepping parameter
ak:	Right hand side finite element stiffness matrix
areabase:	Percentage loss in area factor
b:	Measure of Gaussian integration
bc:	Array for storing boundary constraint value
cartrid:	Reference ID for each filter cartridge element
cg:	Weighting functions for Gauss integration
comfactor:	Compression factor for filter medium
cord:	Array storing coordinate values of each nodal point
da:	Global derivative of shape function
del:	Local derivatives of shape function
deltap:	Experimental pressure drop
deltat:	Value of time step used in calculation iterations
density:	Density of fluid used for each set of experiment
detj:	Determinant of Jacobian matrix
di:	Diameter of inner core of the filter cartridge element
el:	Length of pleat
enp:	Number of pleats in each filter cartridge element
errp:	Square of discrepancy between pressure values calculated in two successive iterations
errv:	Square of discrepancy between velocity values calculated in two successive iterations
eta:	y-coordinate in local coordinate system
excel:	Array for storing simulated values of pressure drop in each step of calculation procedure
farea:	Filtering surface area of cartridge
frate:	Flow rate values for each experiment

LIST OF PRIME VARIABLES - AFCAMP

---

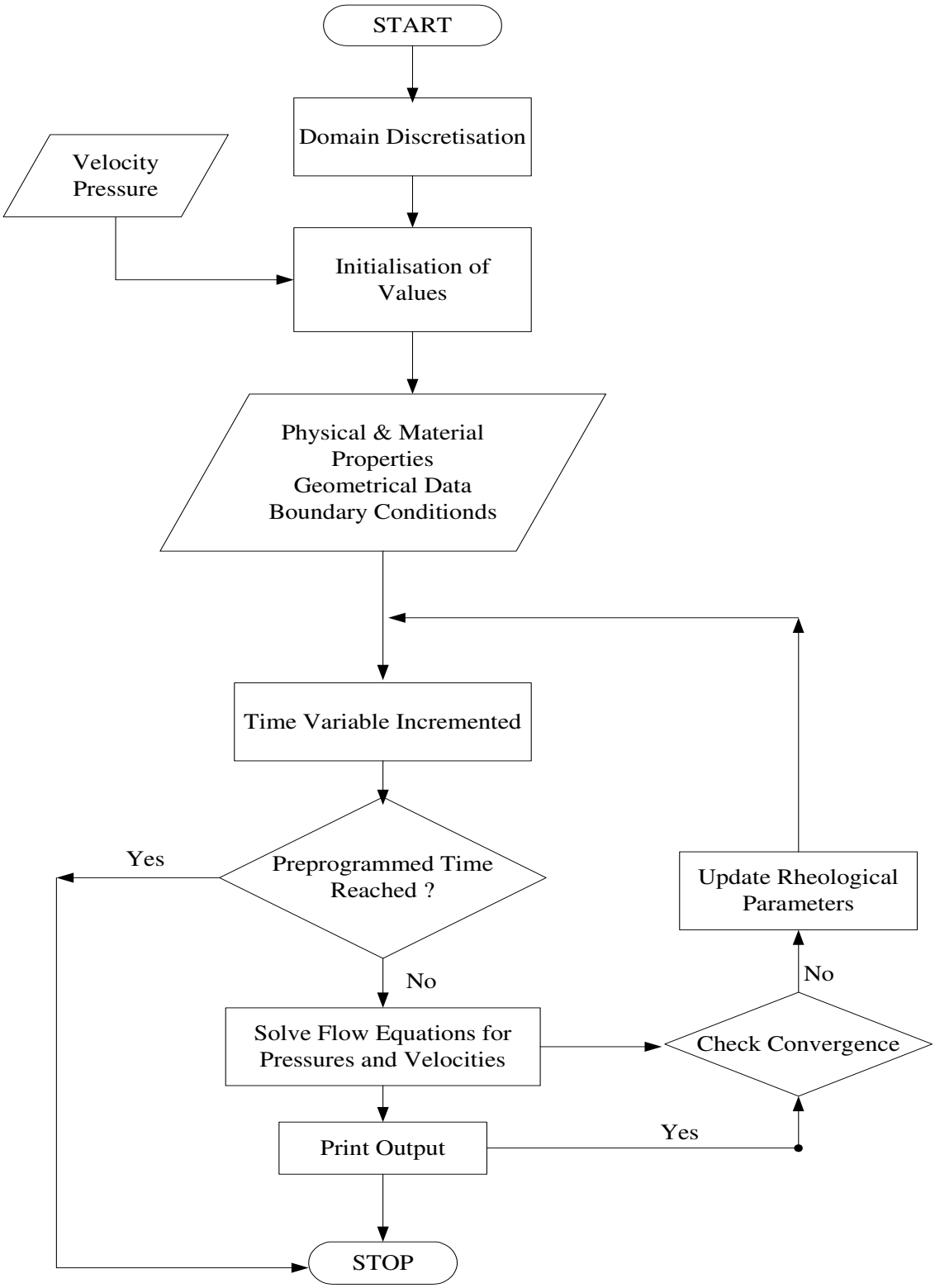
gamad:	Rate of deformation of viscous stress tensor
guessp:	Pressure with compression and loss in area
hp:	Height of filter cartridge element
icord:	Switch for changing the coordinate system of equations
maxbc:	Limit for maximum number of boundary constraints that can be specified
maxdf:	Limit for maximum number of nodal degrees of freedom
maxel:	Limit for maximum number of elements in mesh
maxnp:	Limit for maximum number of nodal points in mesh
maxst:	Maximum dimension of finite element stiffness matrix
nbc:	Total number of boundary-node constraints
ncn:	Number of nodes associated with each finite element
ncod:	Array for constraint switch defined for every degree of freedom
ndf:	Total number of nodal degrees of freedom
ndim:	Dimensions of the solution domain
nel:	Total number of elements in pleated cartridge mesh
ngaus:	Number of quadrature points required for Gaussian integration
nmat:	Total number of fluids
nnp:	Total number of nodal points in pleated cartridge mesh
node:	Array for nodal connectivity of elements
ntd:	Number of sets of experiment
nter:	Number of solution iterations
num:	Number of integration points per element
p:	Finite element shape or interpolation functions
pd:	Experimental pressure value for each set of data
percomp:	Percentage value of filter medium compression
permstep1:	Permeability of filter medium without compression
permstep2:	Permeability of filter medium with compression
permx:	Permeability of the filter medium in x-direction
permy:	Permeability of the filter medium in y-direction
pmat:	Array storing values of physical properties of the fluid

LIST OF PRIME VARIABLES - AFCAMP

---

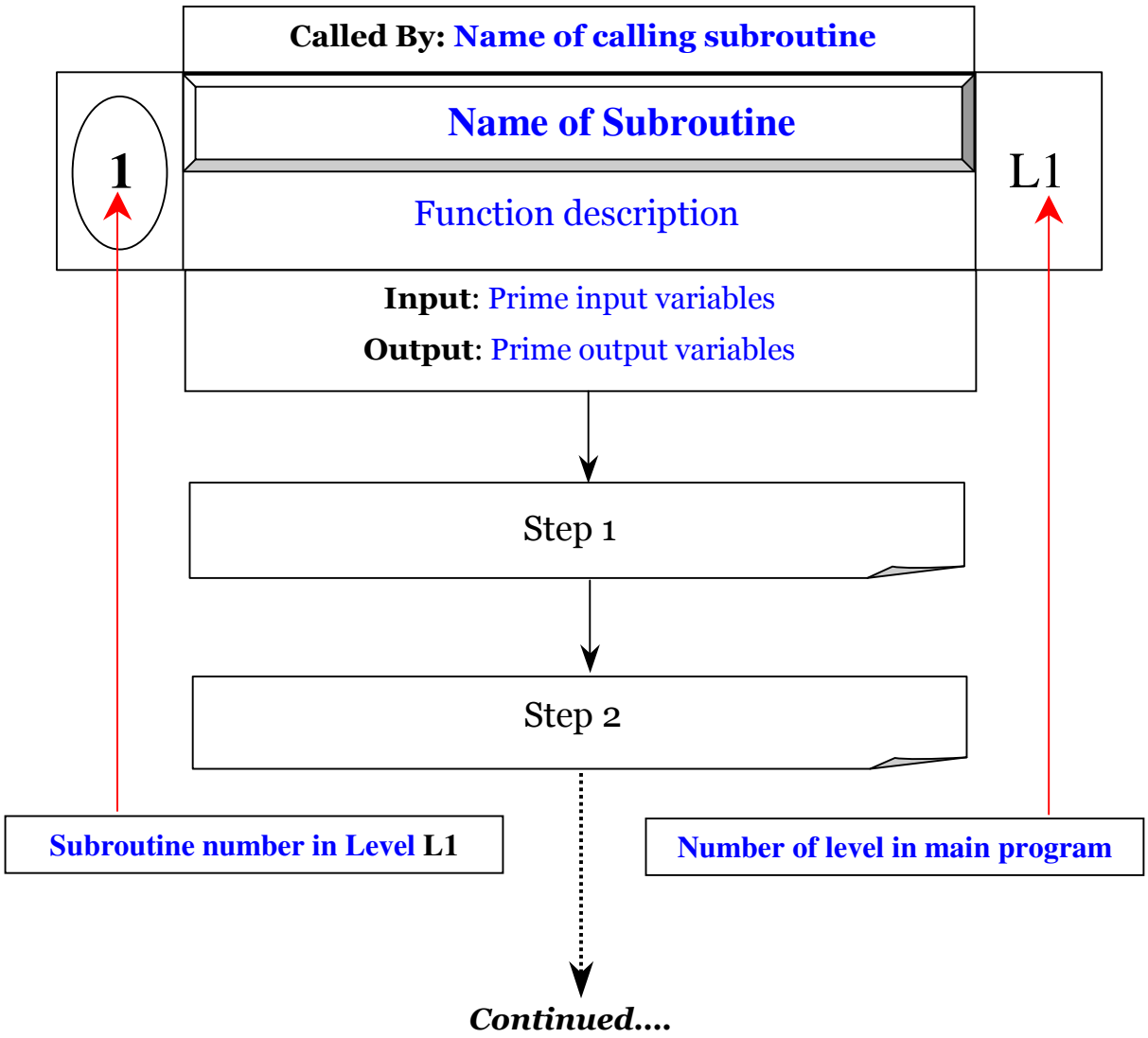
power:	Power law index for the fluid viscosity
press:	Array for nodal pressure
pressexp:	Experimental pressure drop
rpef:	Reference pressure
rr:	Array for storing the elemental load vector
rtem:	Reference temperature
rvisc:	Consistency coefficient for viscosity of fluid
sinv:	Array storing components of second invariant of rate of deformation of stress tensor
soln:	Array for storing solution values calculated by Frontal Solver
step:	Stage of the calculation procedure
taco:	Coefficient a in the power law model
tbco:	Coefficient b in the power law model
time:	Amount of time for iterations
tolv:	Maximum allowable tolerance for fluctuation in velocity value
tolp:	Maximum allowable tolerance for fluctuation in pressure value
vel:	Array for nodal velocities
velsound:	Velocity of sound in fluid
viisc:	Viscosity of fluid used for each set of experiment
visc:	Updated value of viscosity according to power law model
xg:	Coordinates of Gauss quadrature points for integration
xi:	x-coordinate in local coordinate system

SOLUTION ALGORITHM - AFCAMP



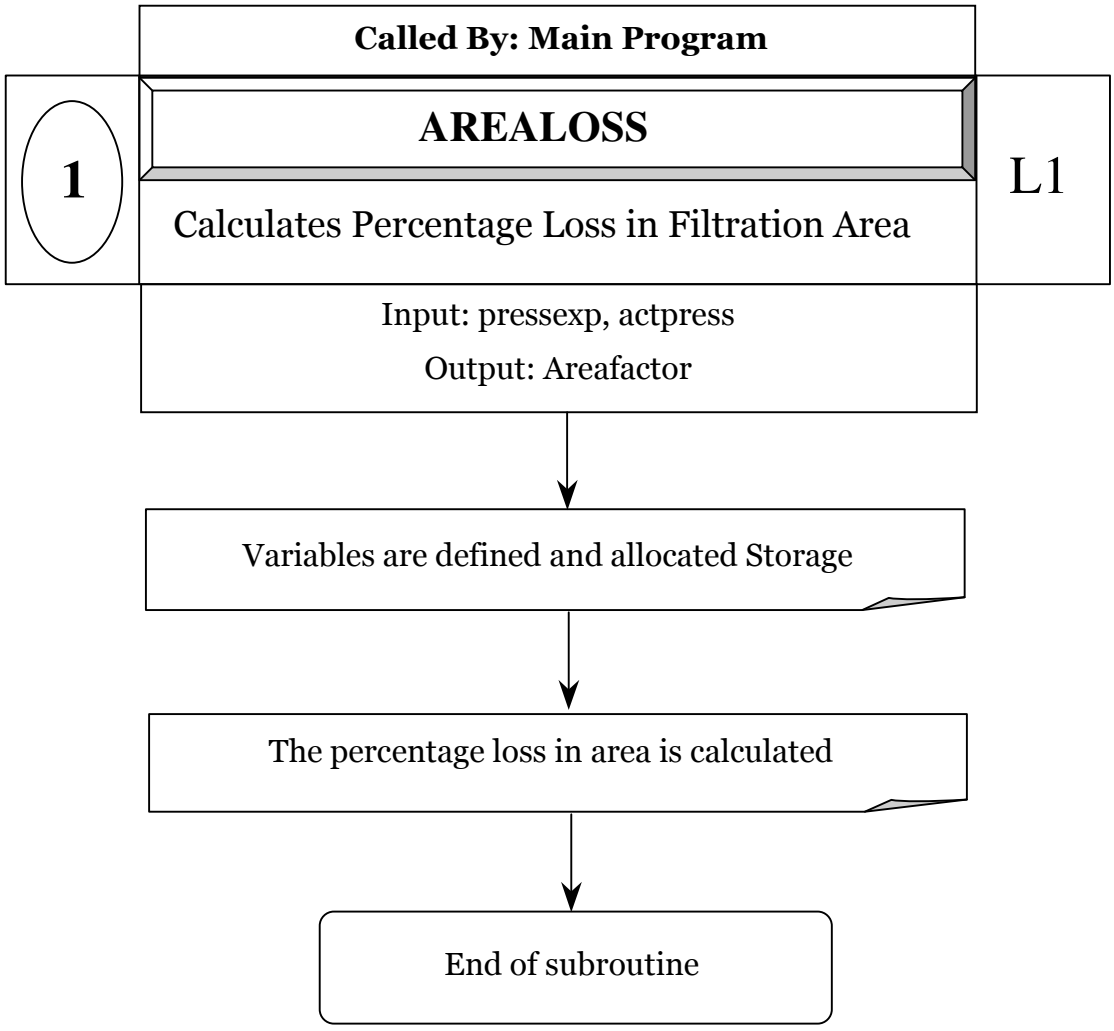
PROGRAM SUBROUTINES - INTERACTIONS

A PROTOTYPE CONVENTION



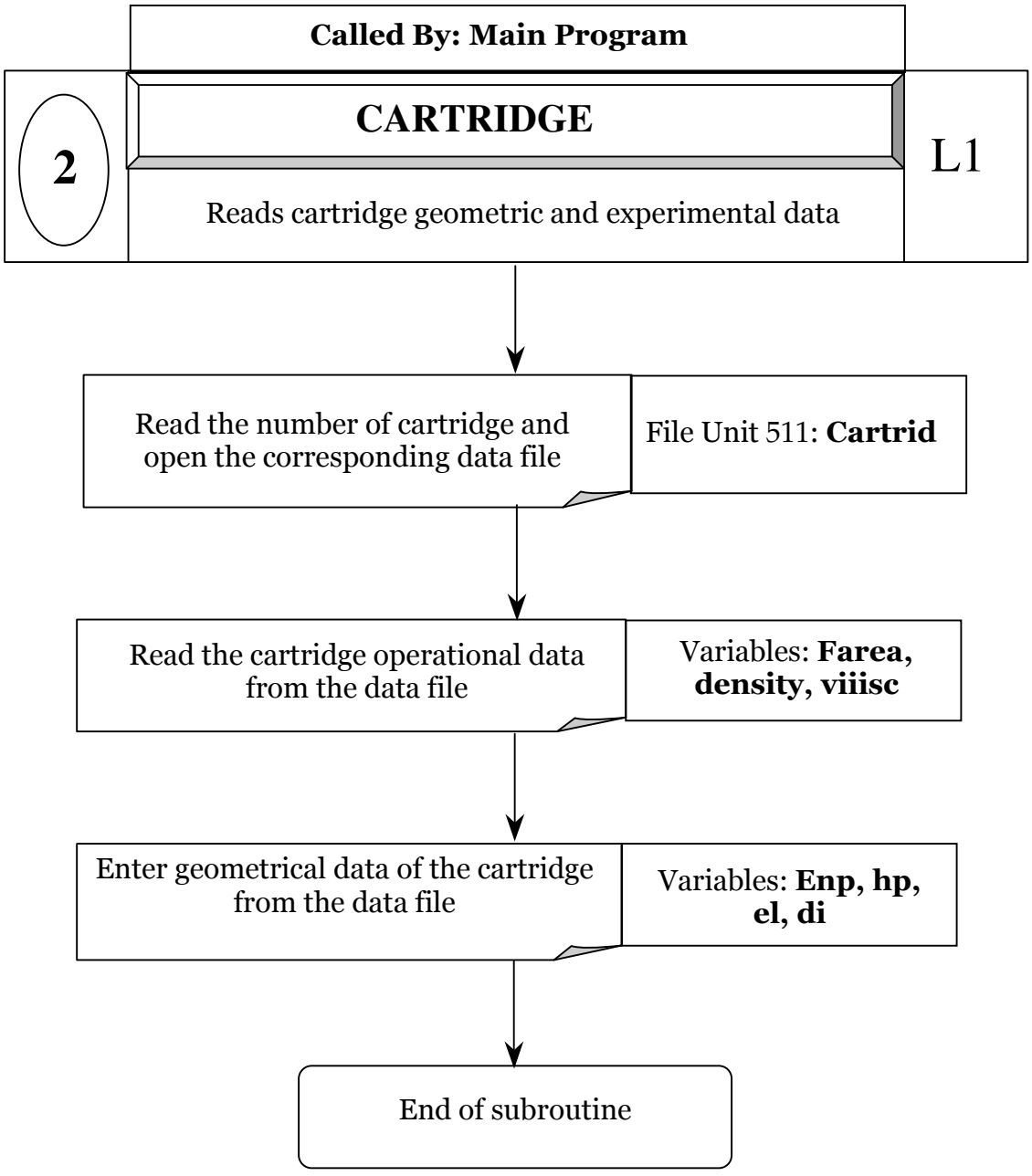
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE AREALOSS, LEVEL 1



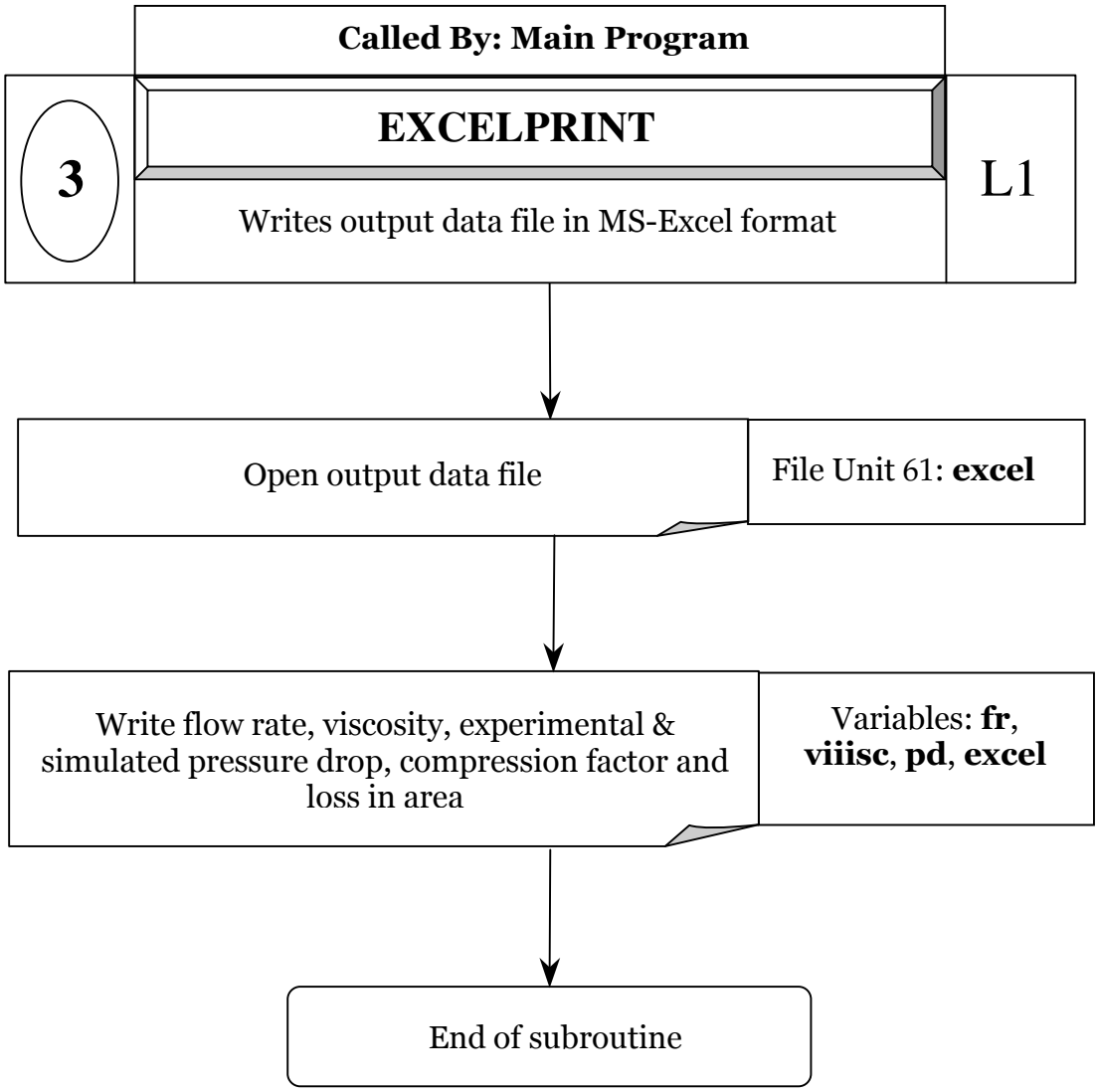
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE CARTRIDGE, LEVEL 1



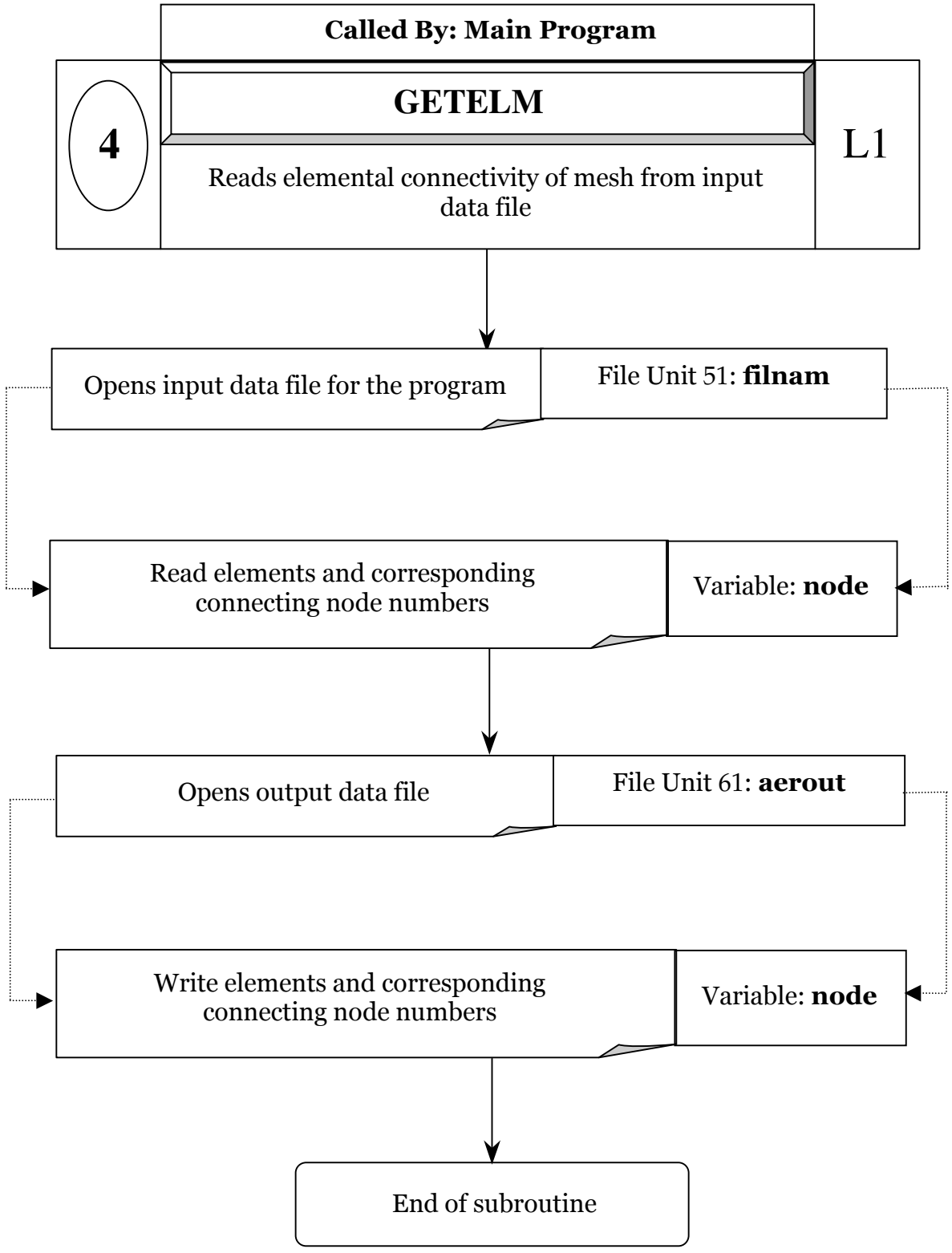
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE EXCELPRINT, LEVEL 1



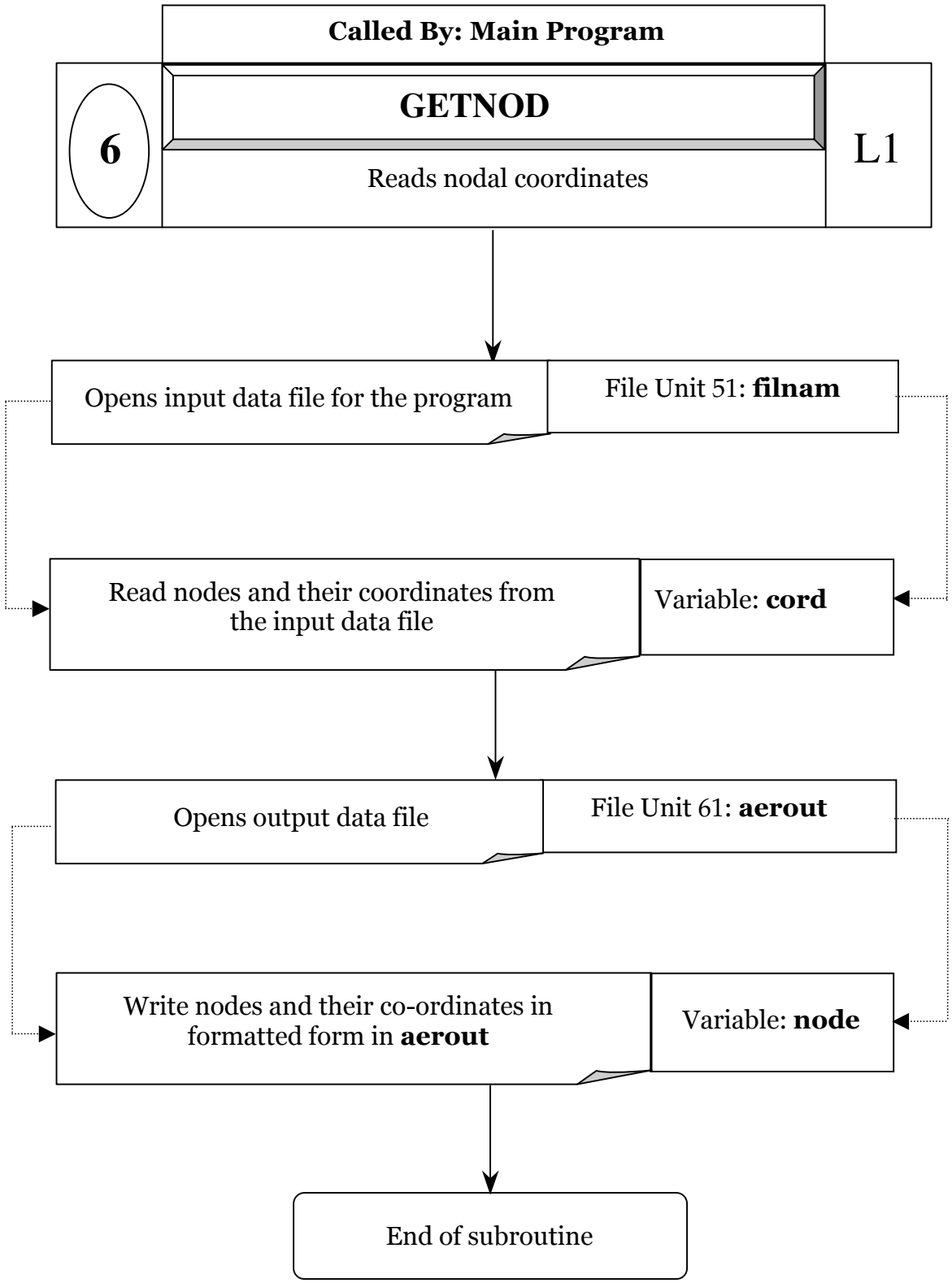
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE GETELM, LEVEL 1



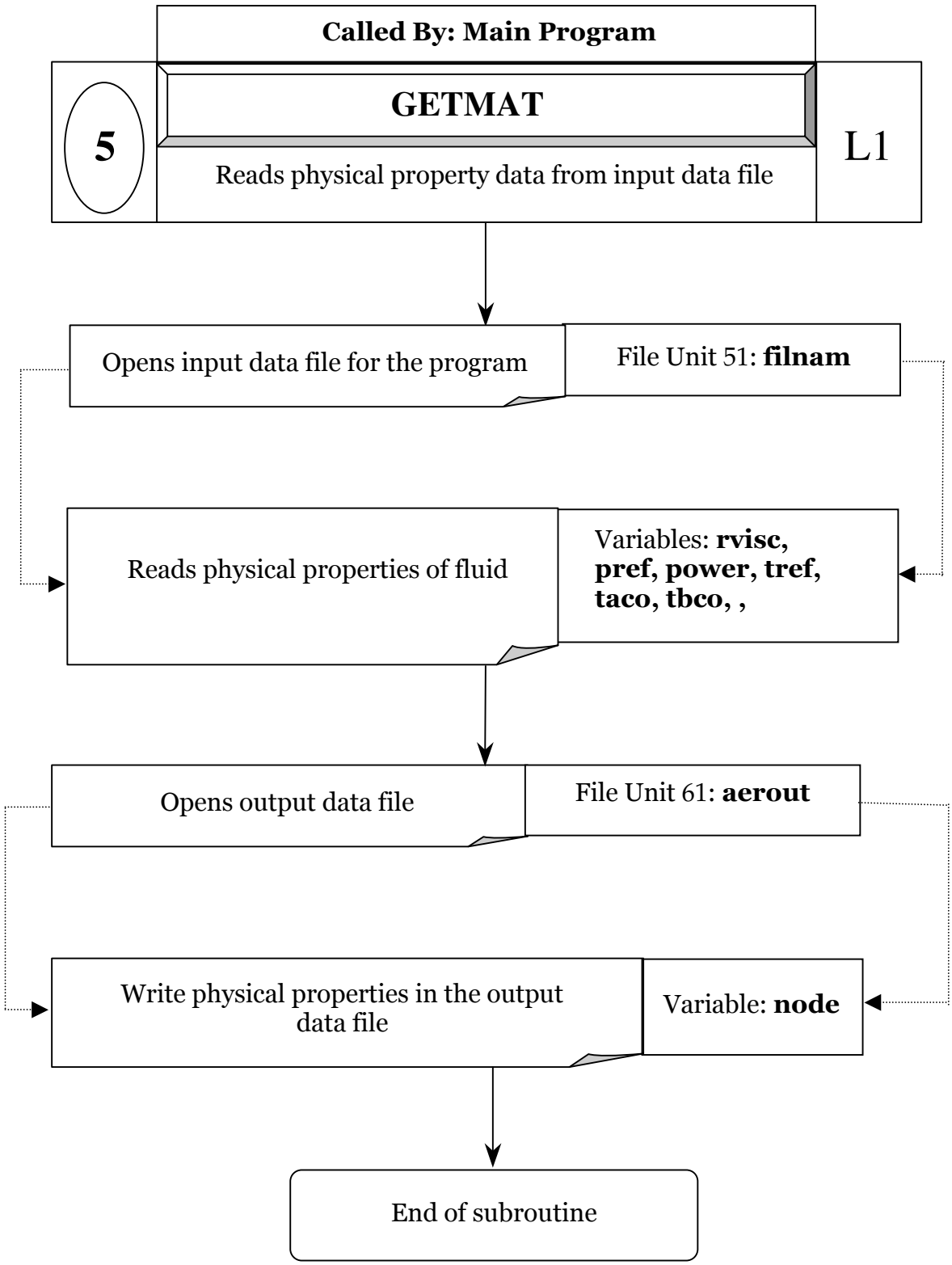
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE GETNOD, LEVEL 1



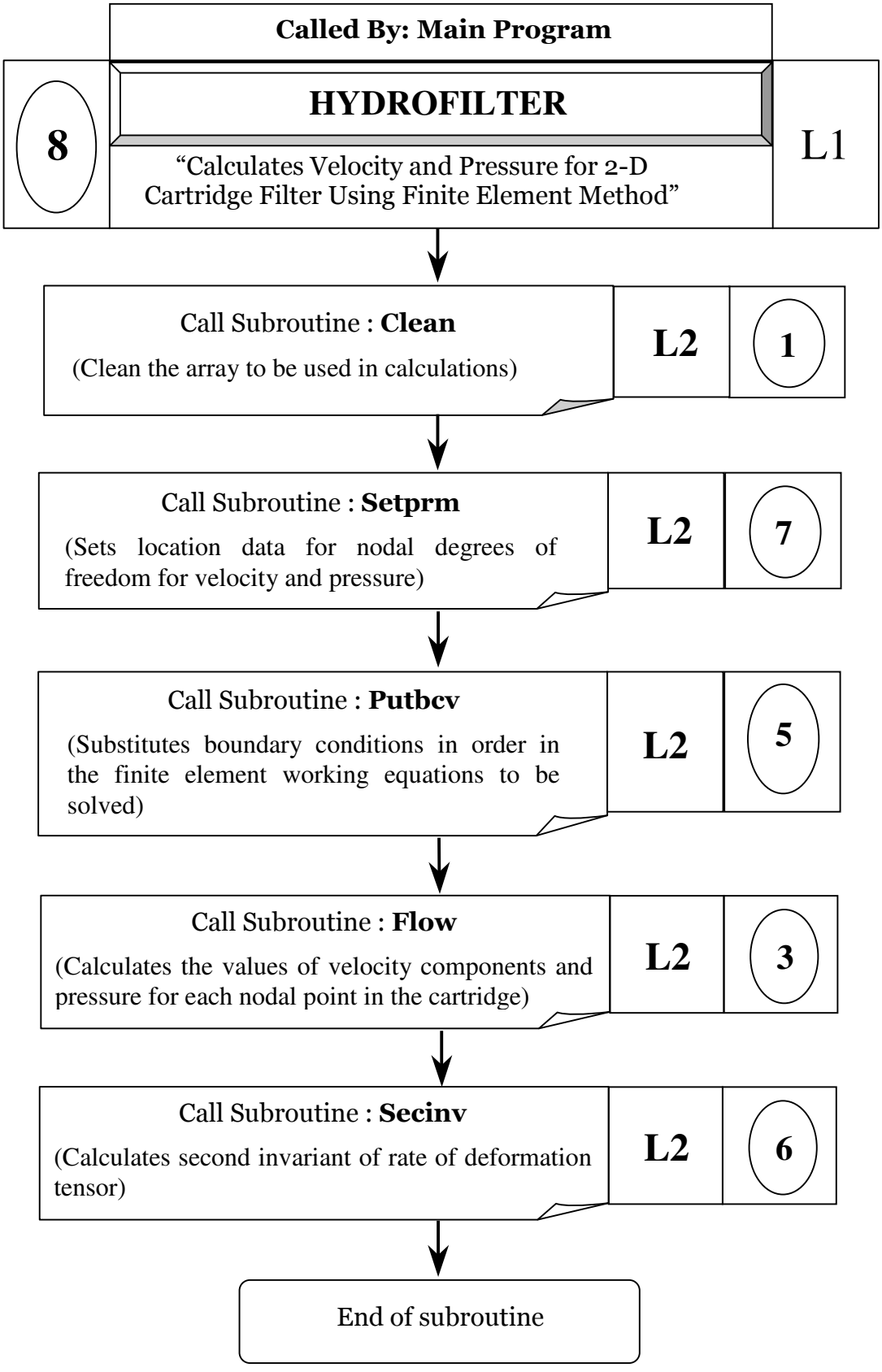
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE GETMAT, LEVEL 1



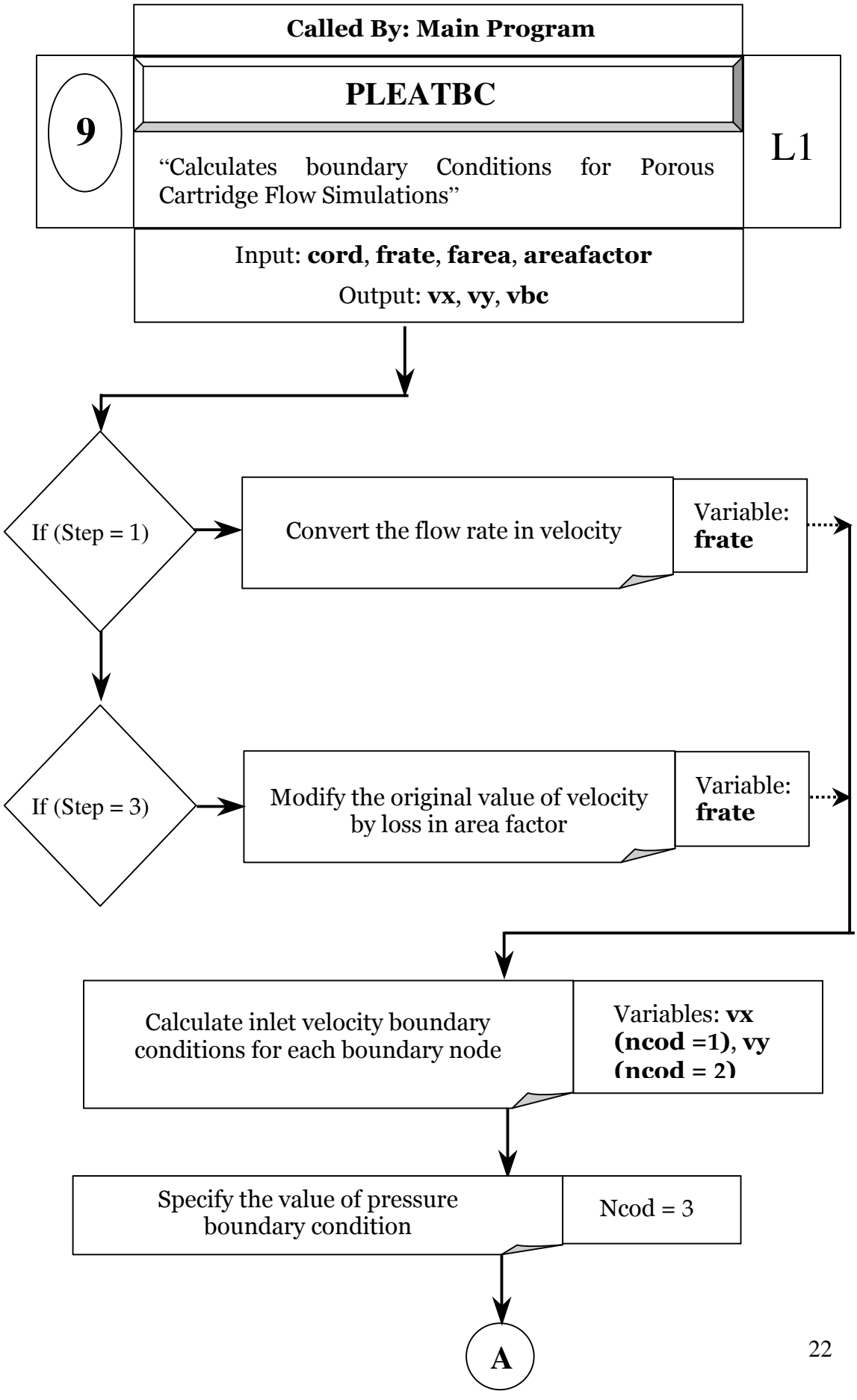
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE HYDROFILTER, LEVEL 1

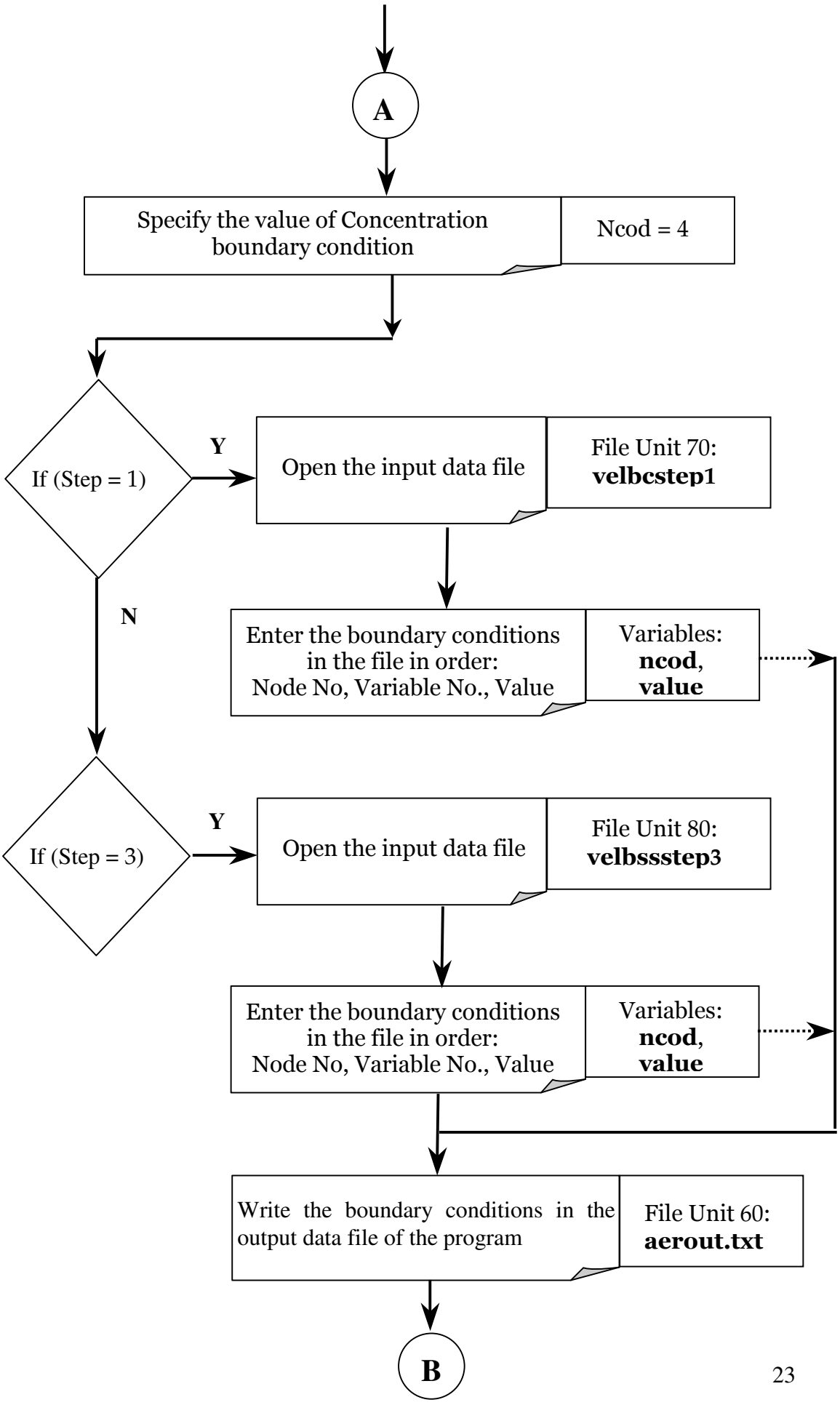


PROGRAM SUBROUTINES - INTERACTIONS

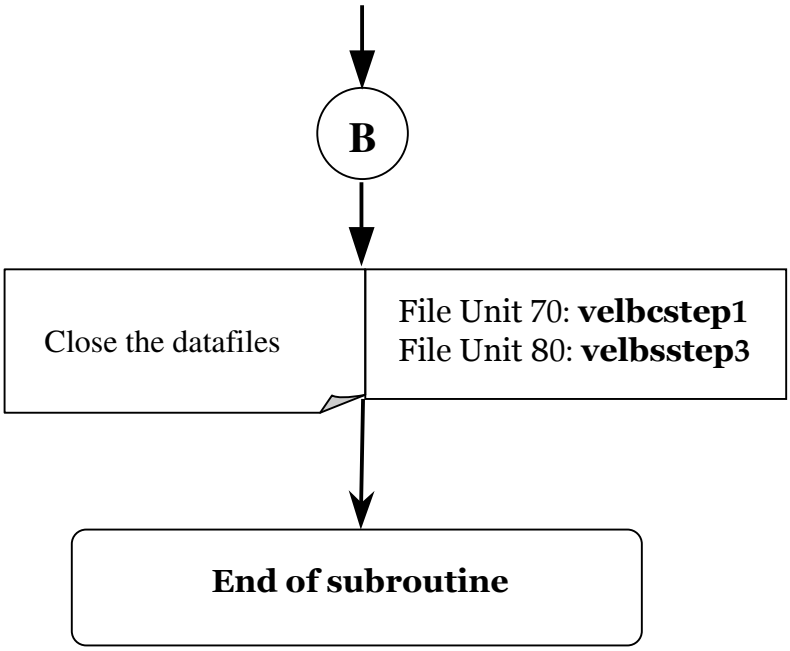
SUBROUTINE PLEATBC, LEVEL 1



PROGRAM SUBROUTINES - INTERACTIONS

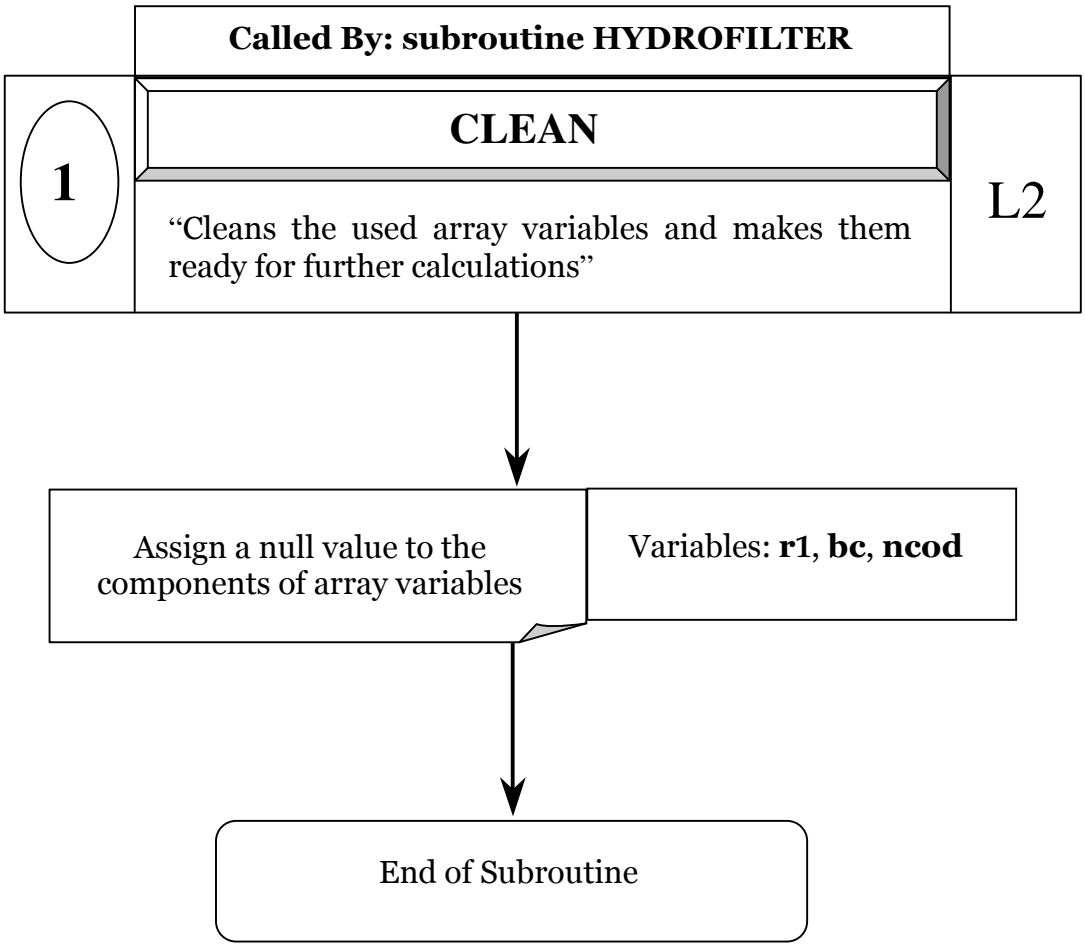


PROGRAM SUBROUTINES - INTERACTIONS



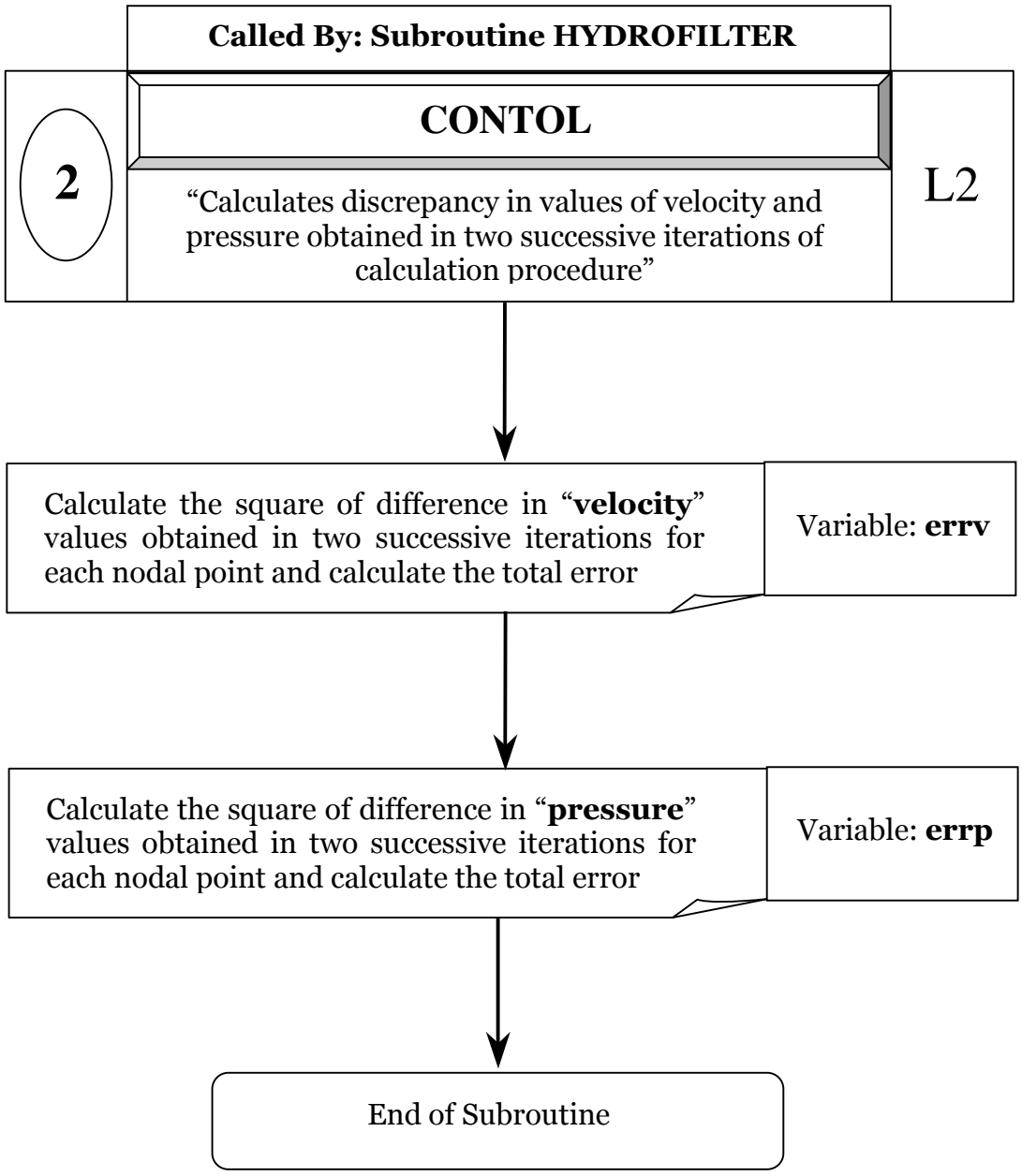
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE CLEAN, LEVEL 2



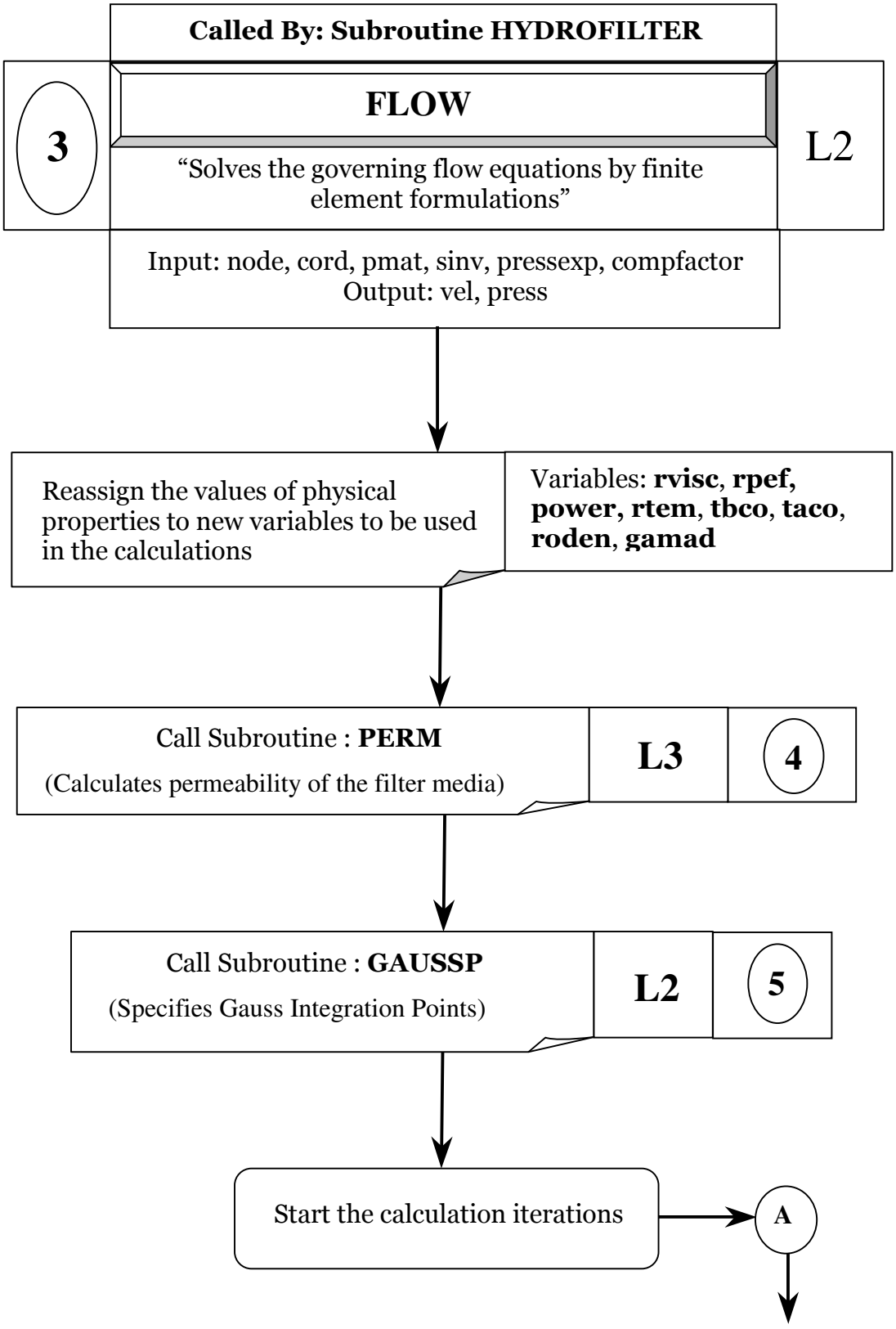
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE CONTOL, LEVEL 2

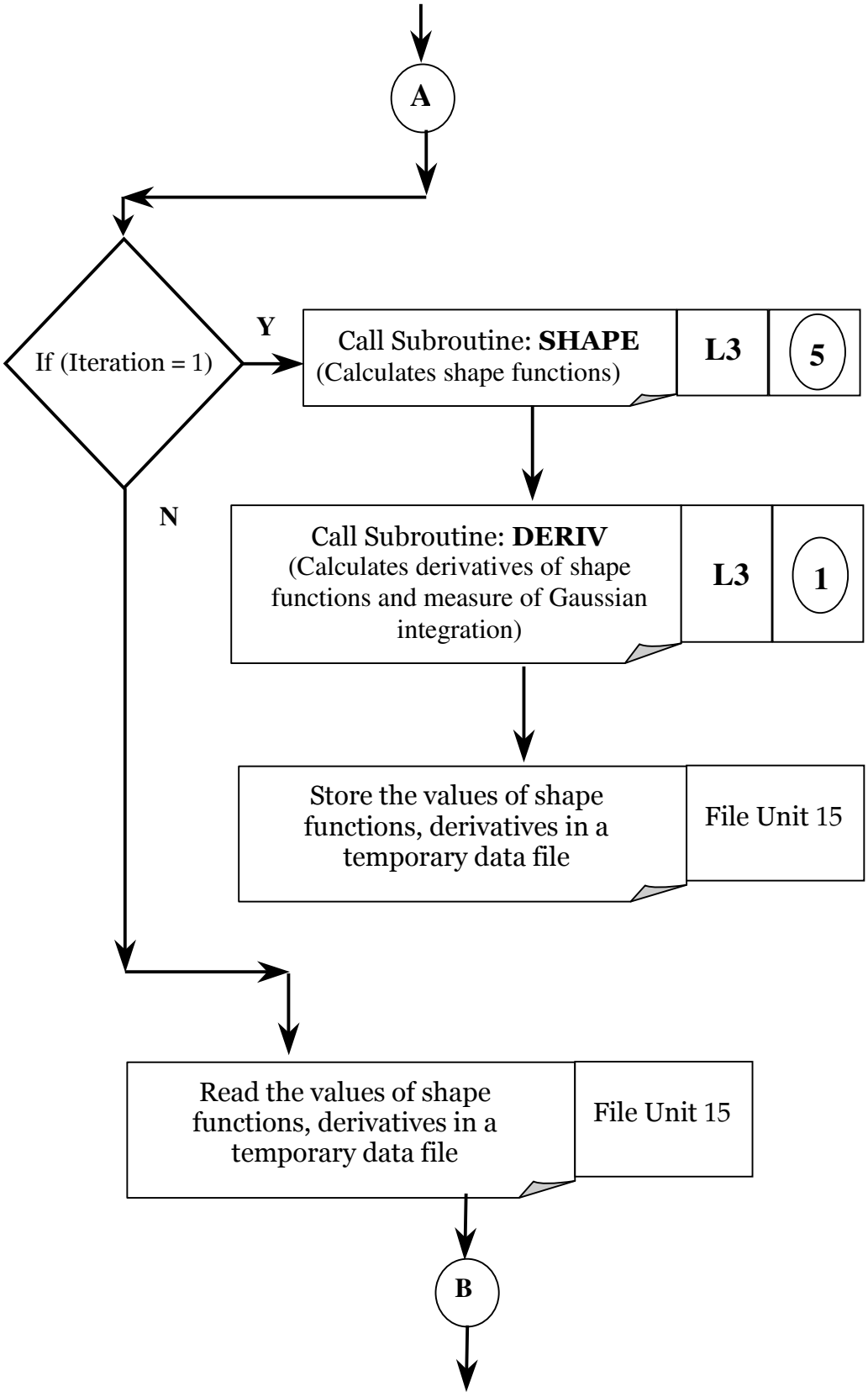


PROGRAM SUBROUTINES - INTERACTIONS

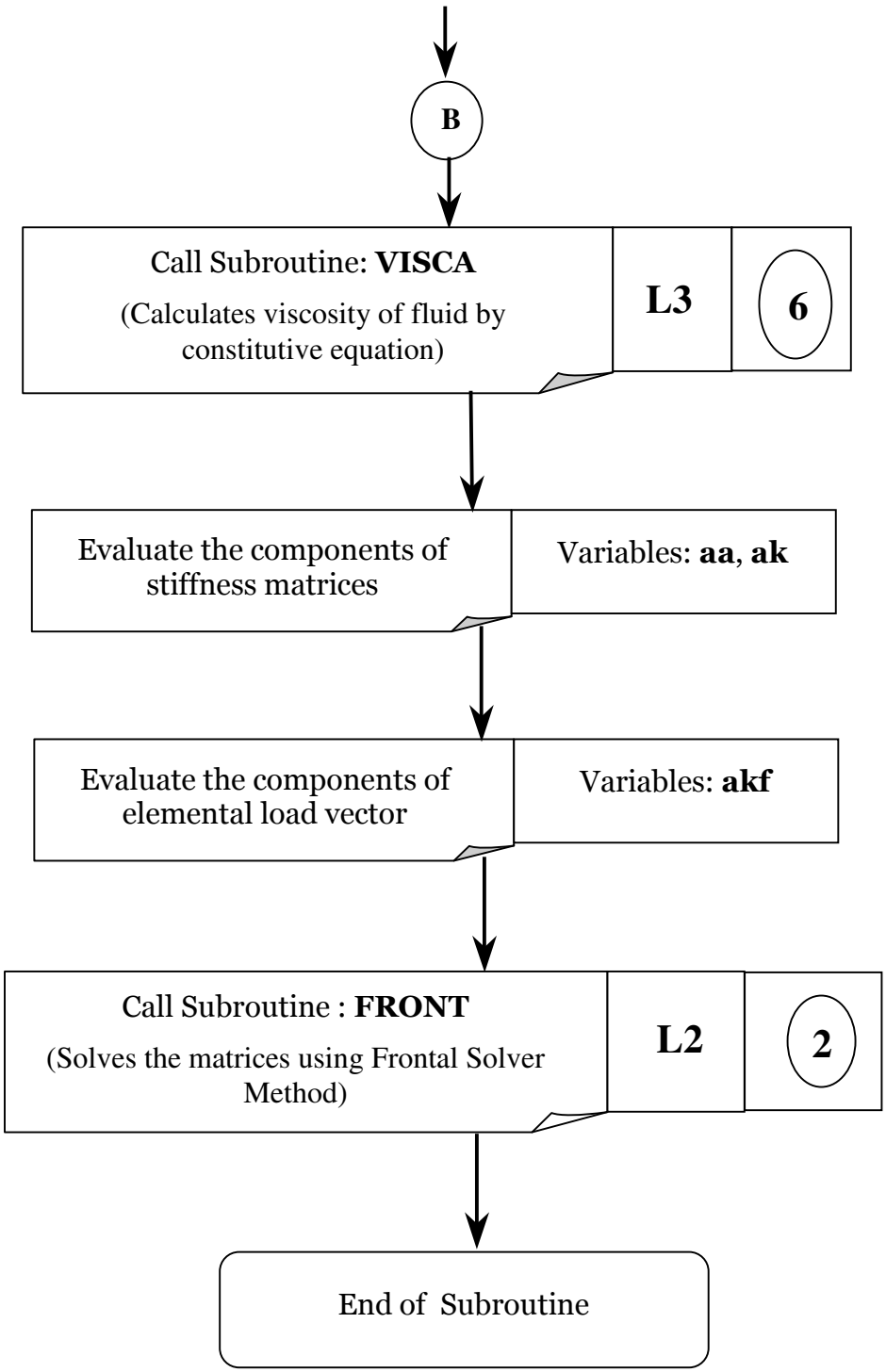
SUBROUTINE FLOW, LEVEL 2



PROGRAM SUBROUTINES - INTERACTIONS

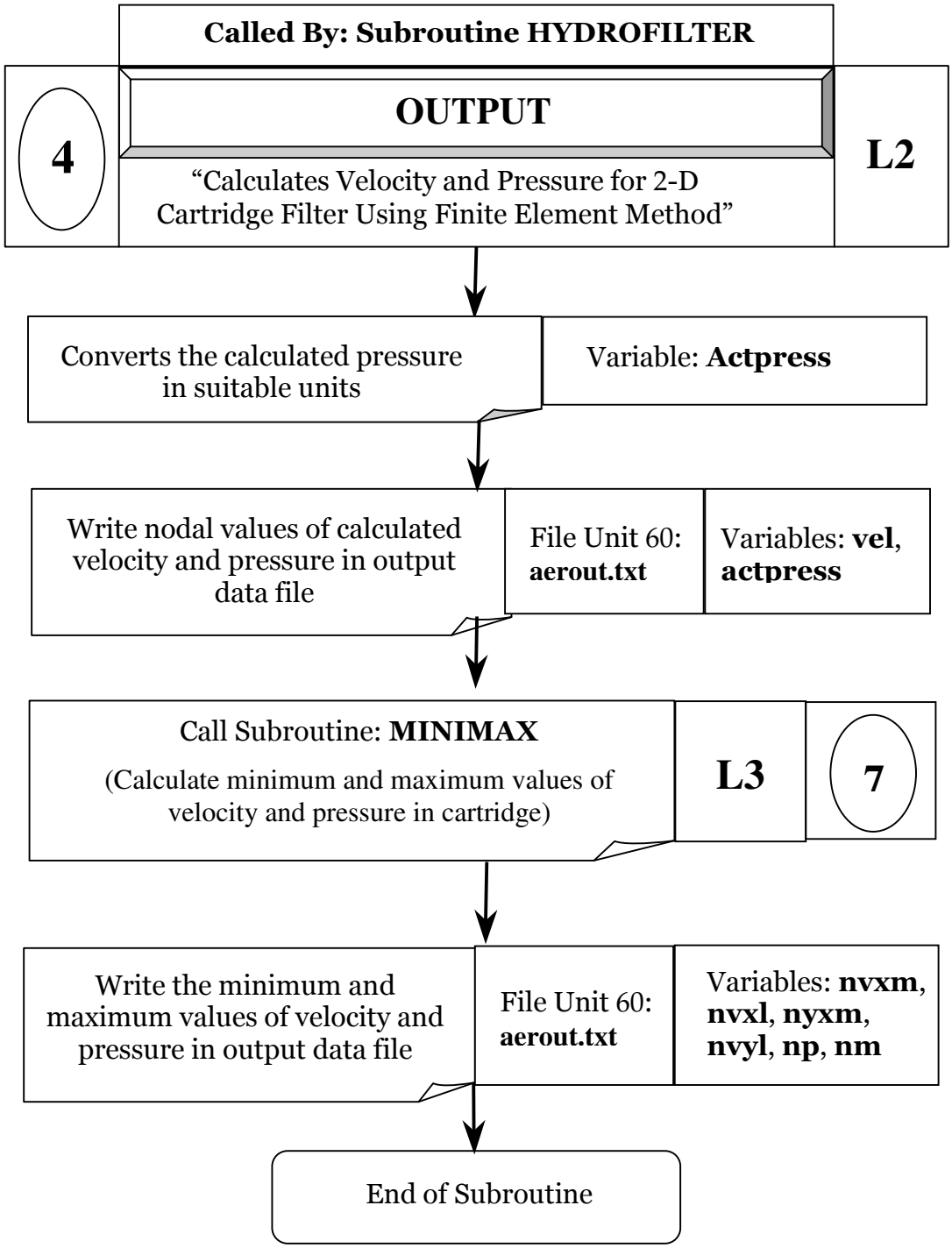


PROGRAM SUBROUTINES - INTERACTIONS



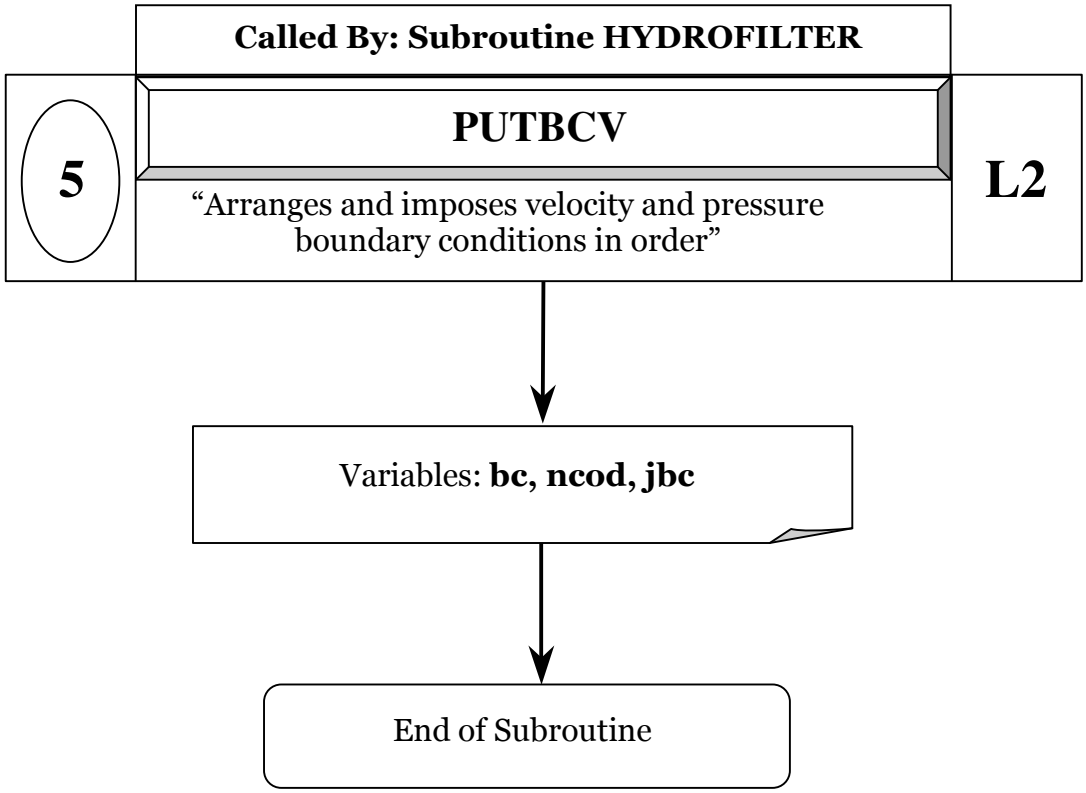
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE OUTPUT, LEVEL 2



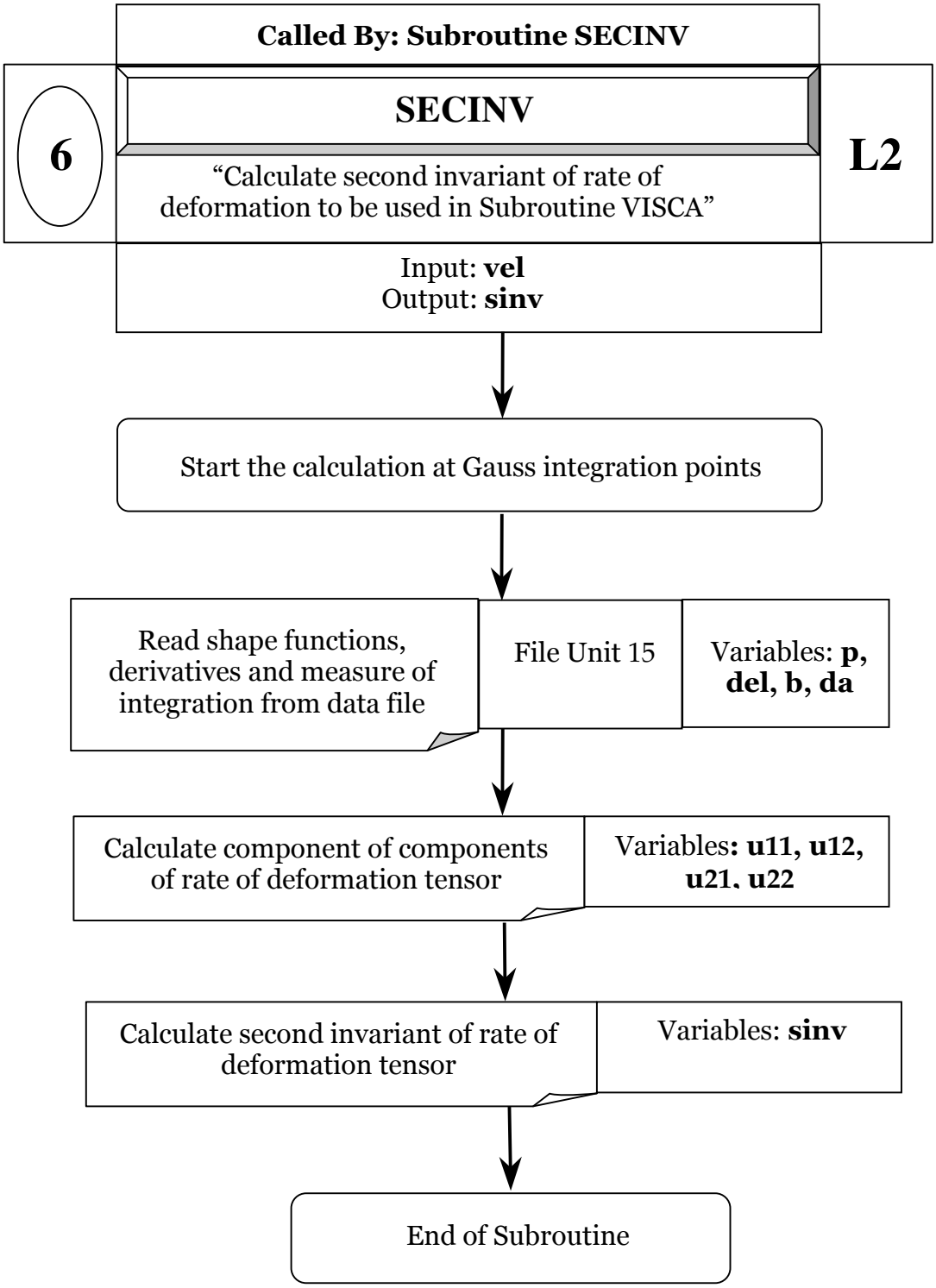
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE PUTBCV, LEVEL 2



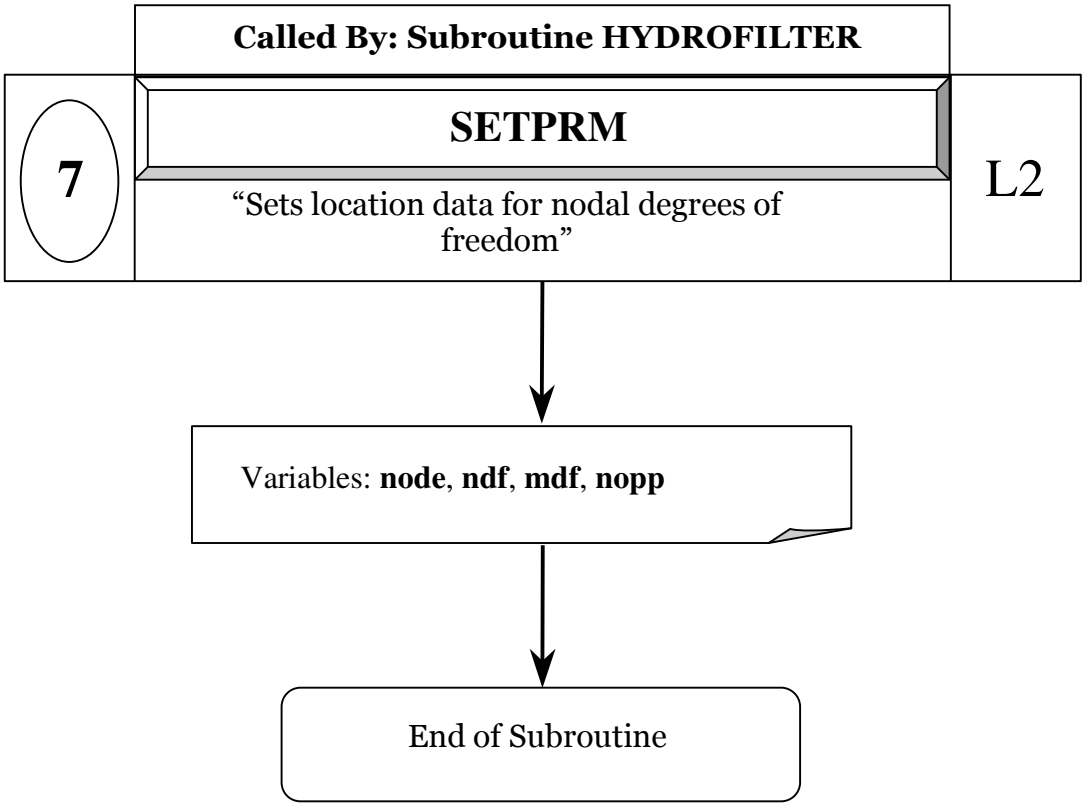
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE SECINV, LEVEL 2



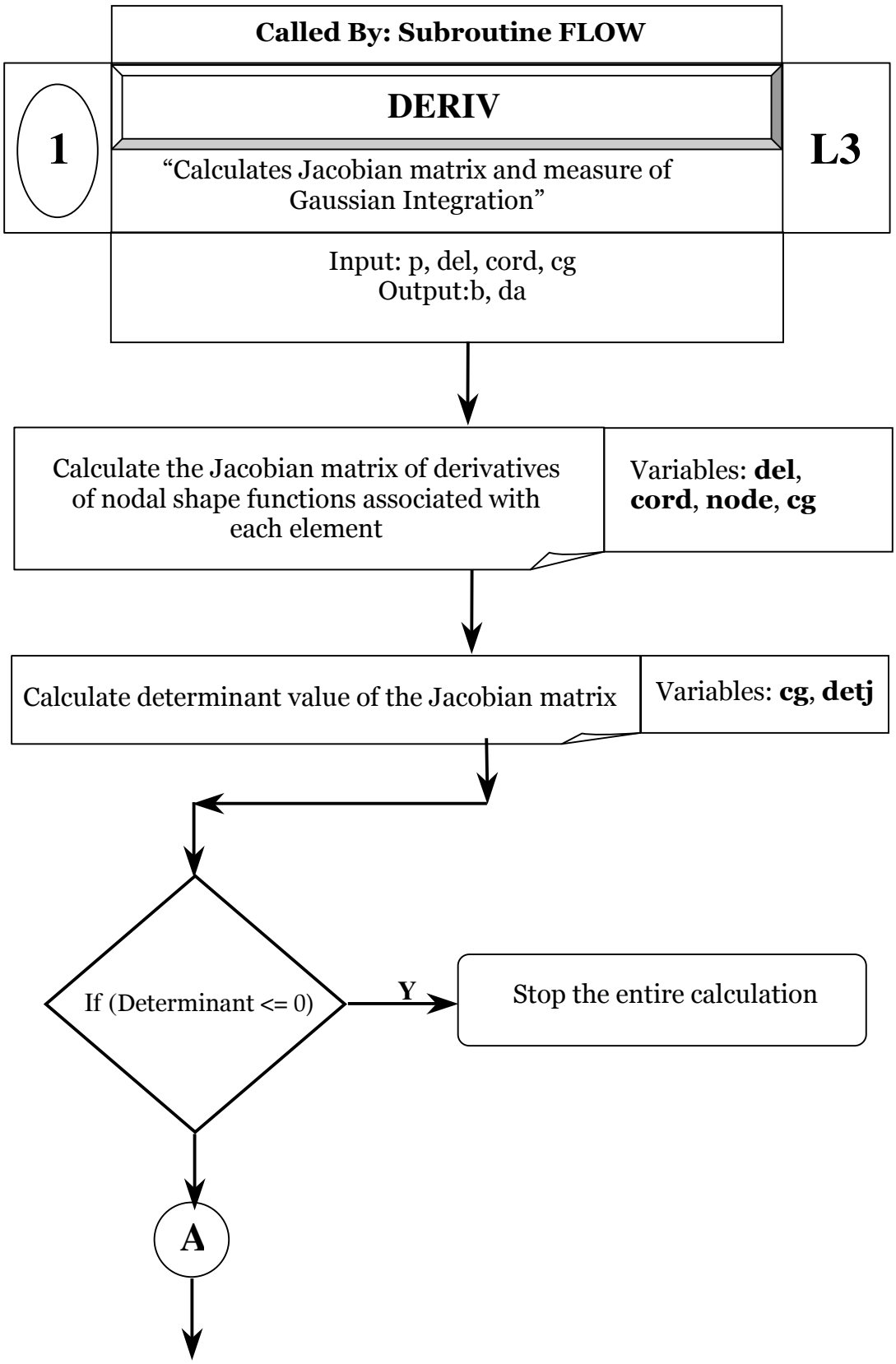
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE SETPRM, LEVEL 2

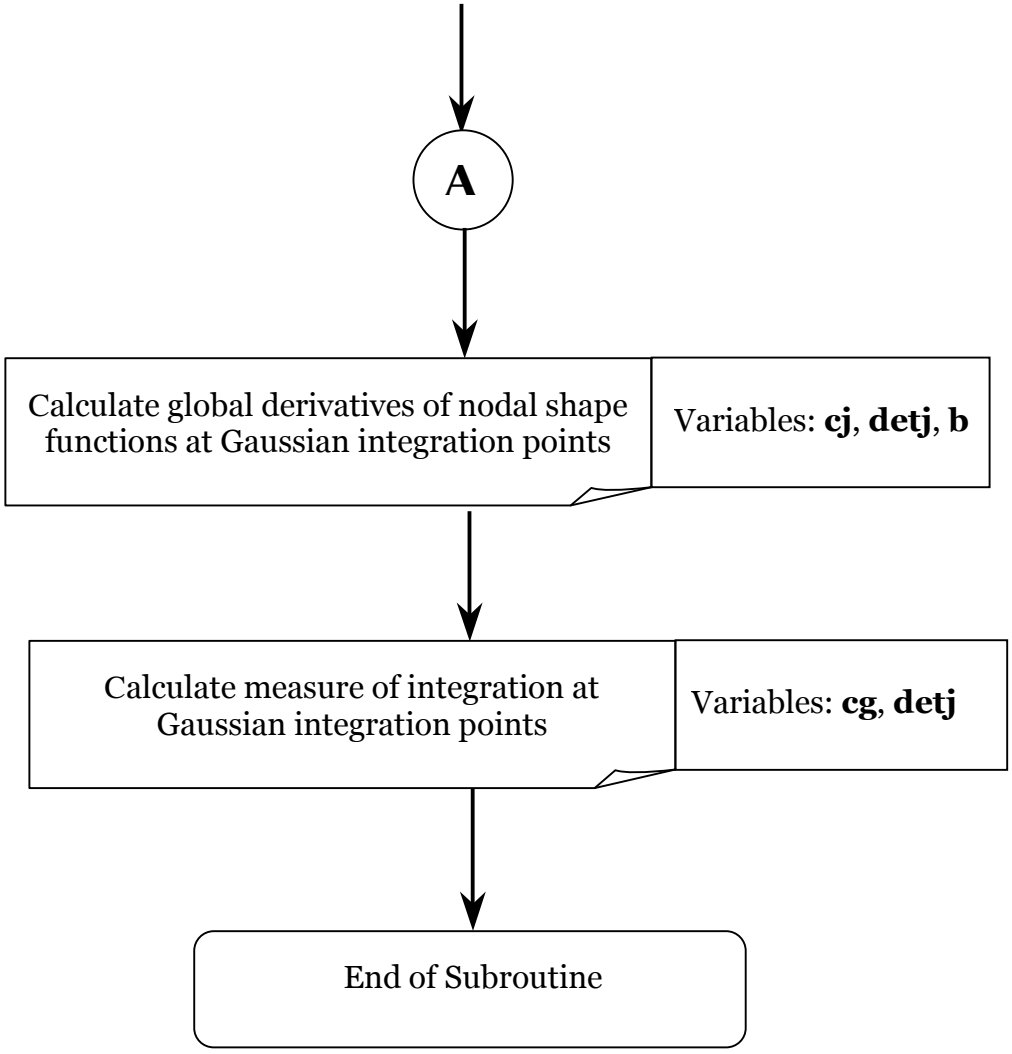


PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE DERIV, LEVEL 3

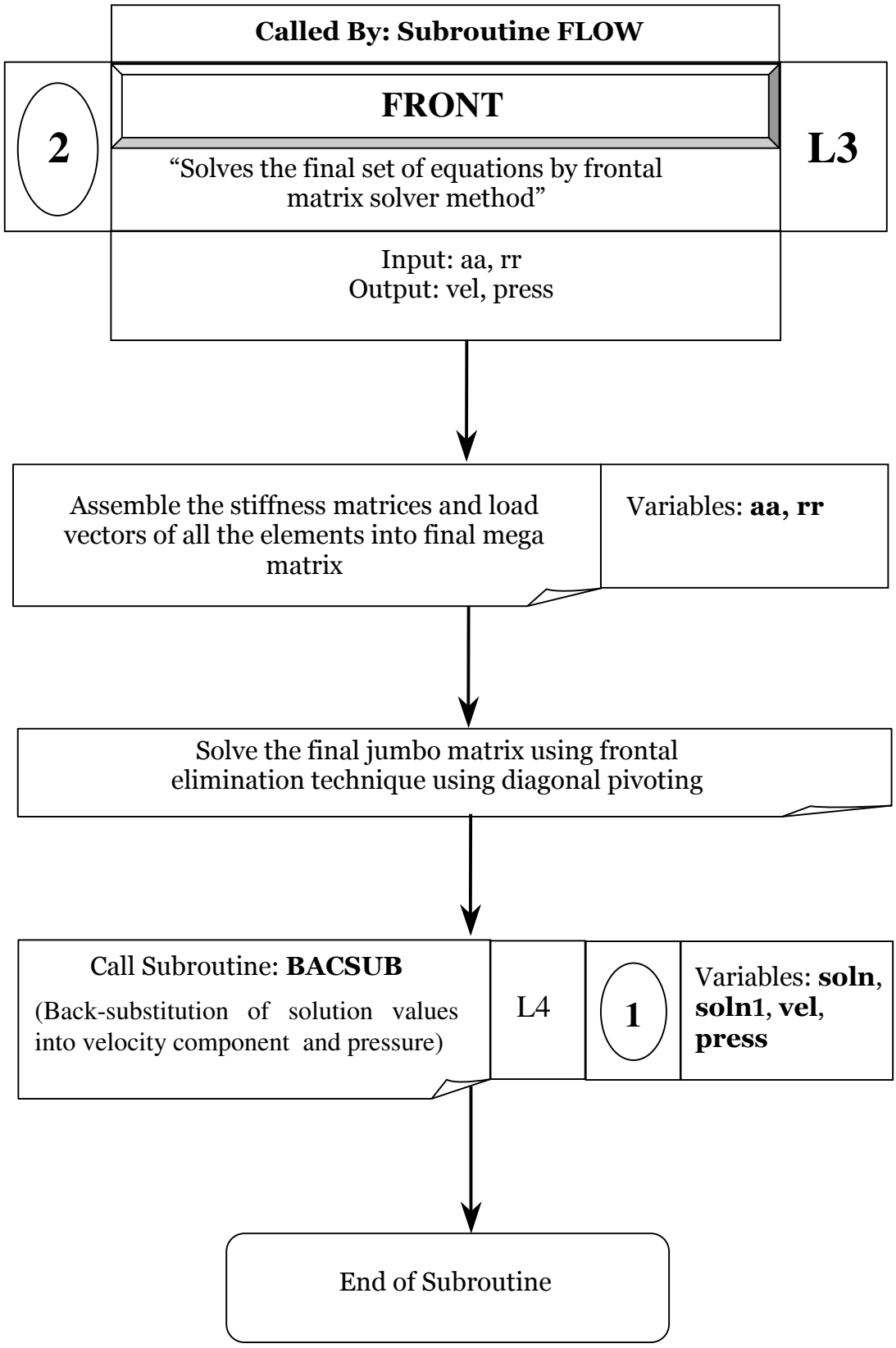


PROGRAM SUBROUTINES - INTERACTIONS



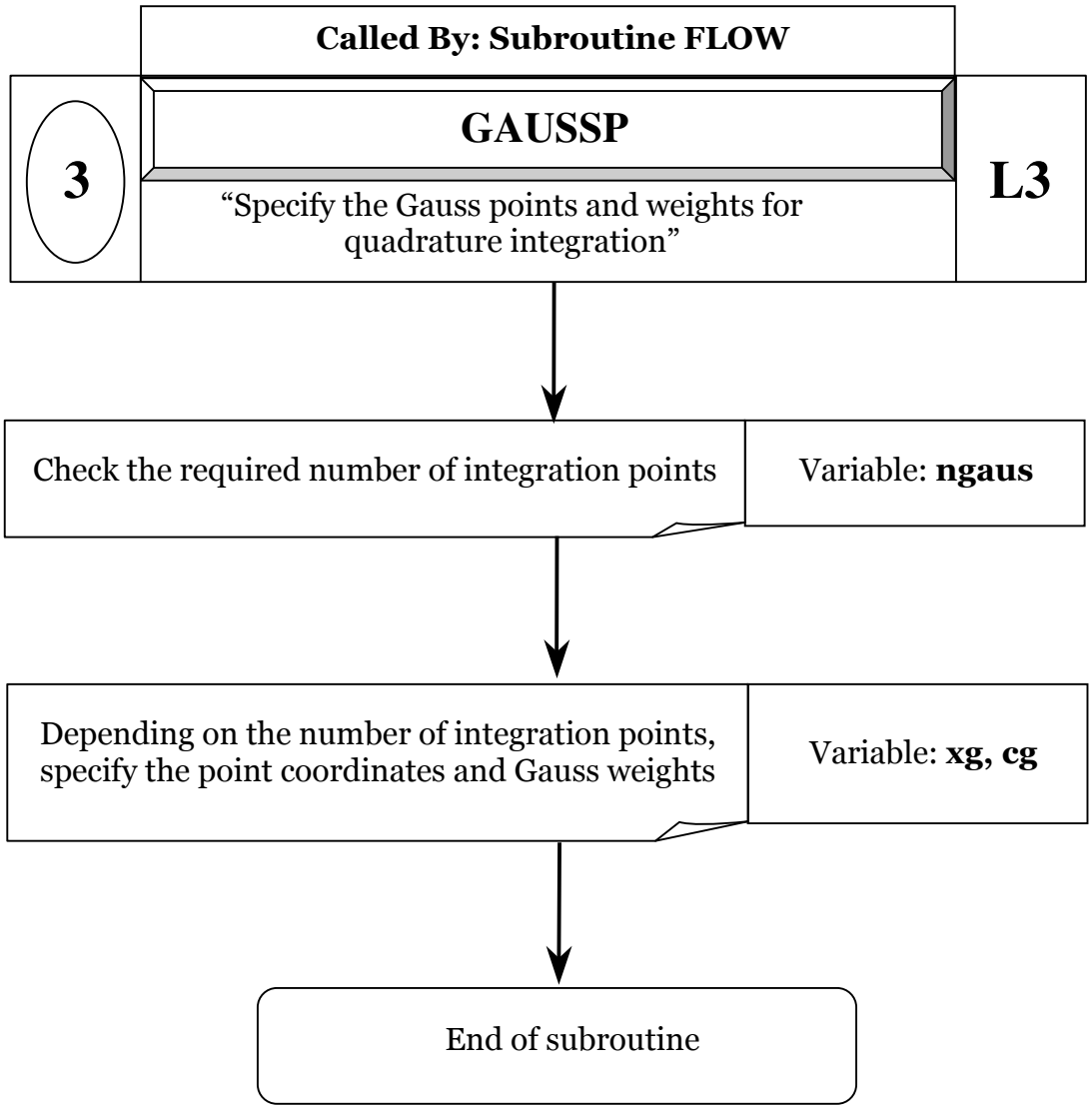
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE FRONT, LEVEL 3



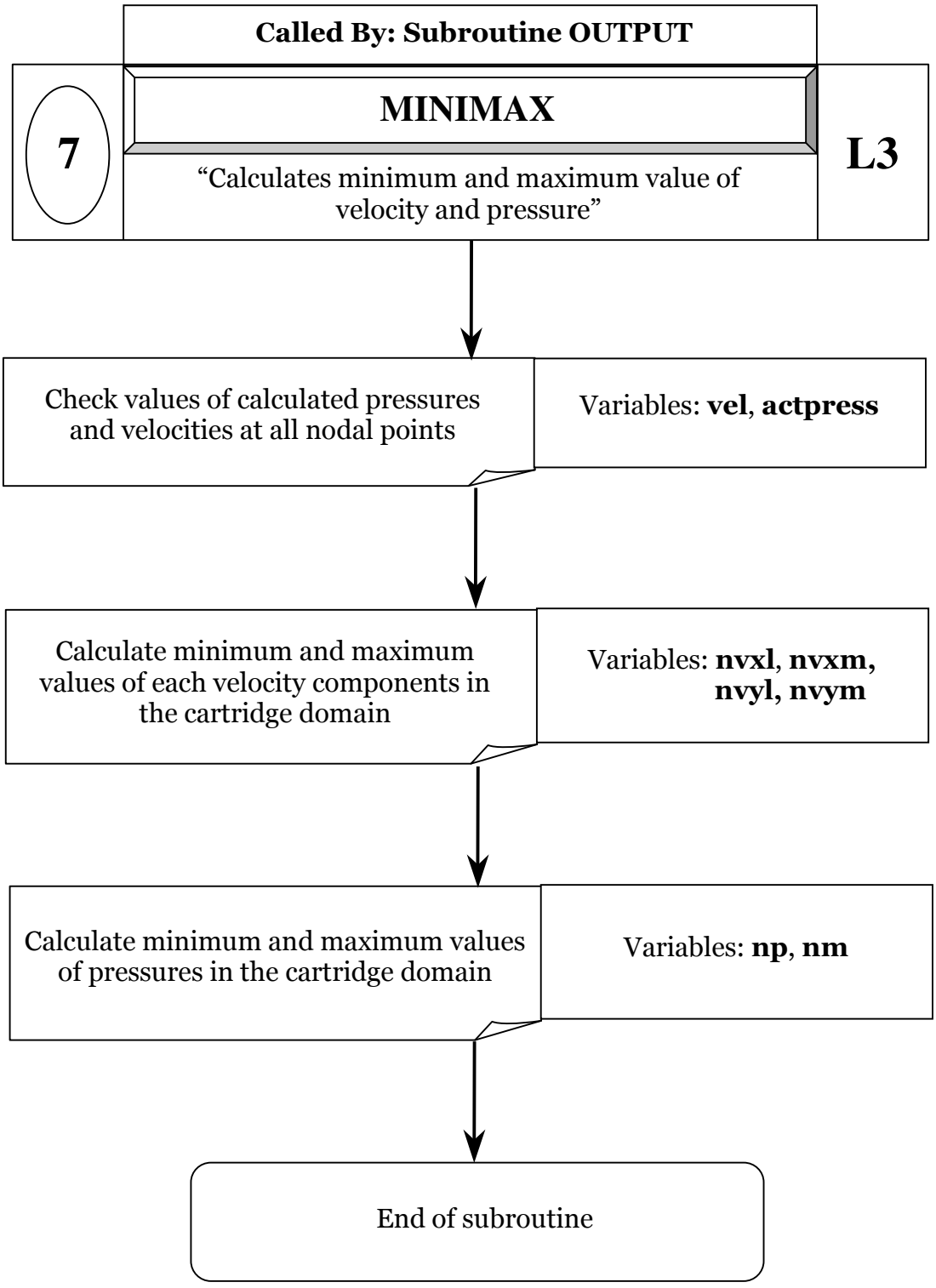
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE GAUSSP, LEVEL 3



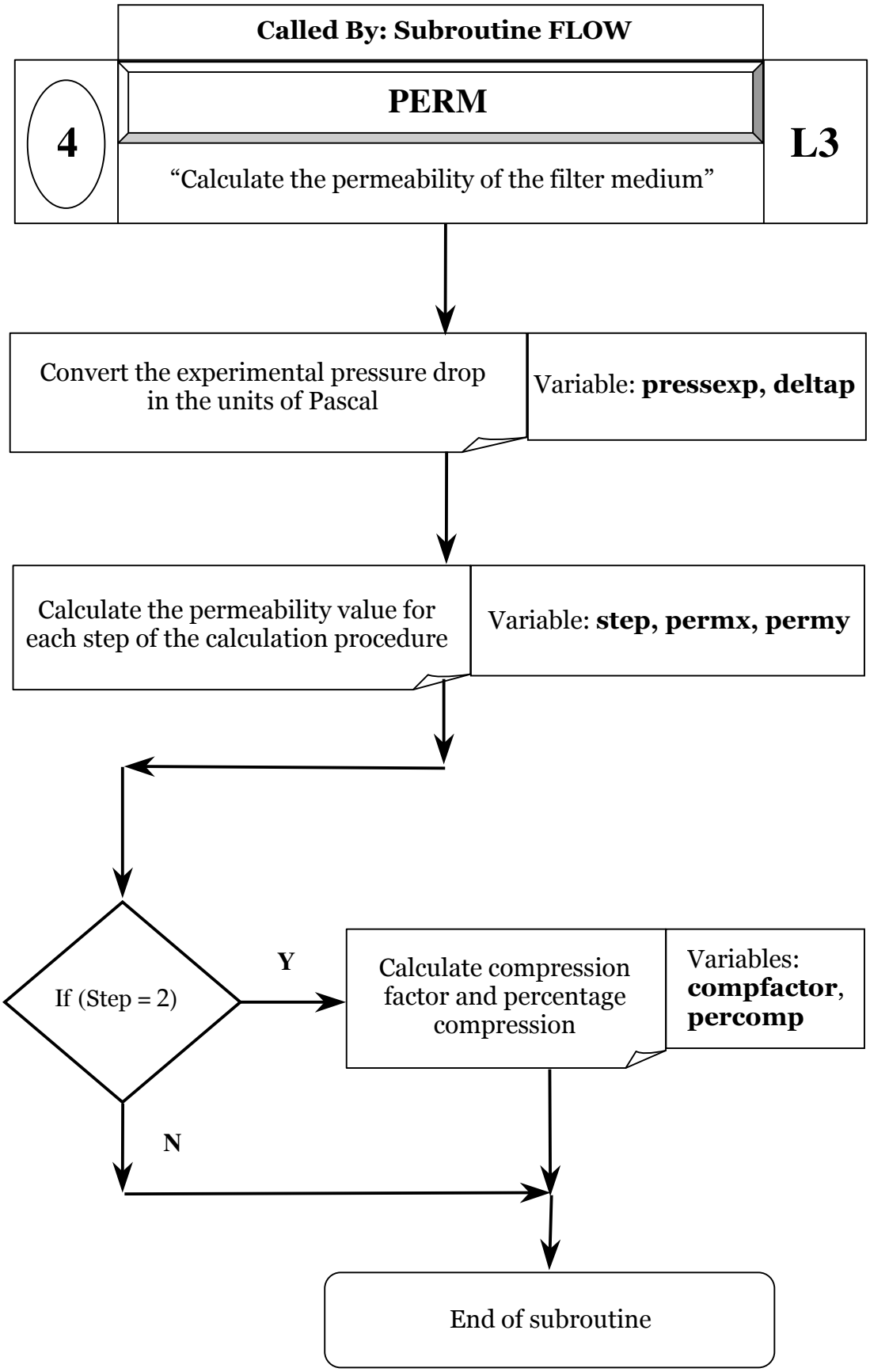
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE MINIMAX, LEVEL 3



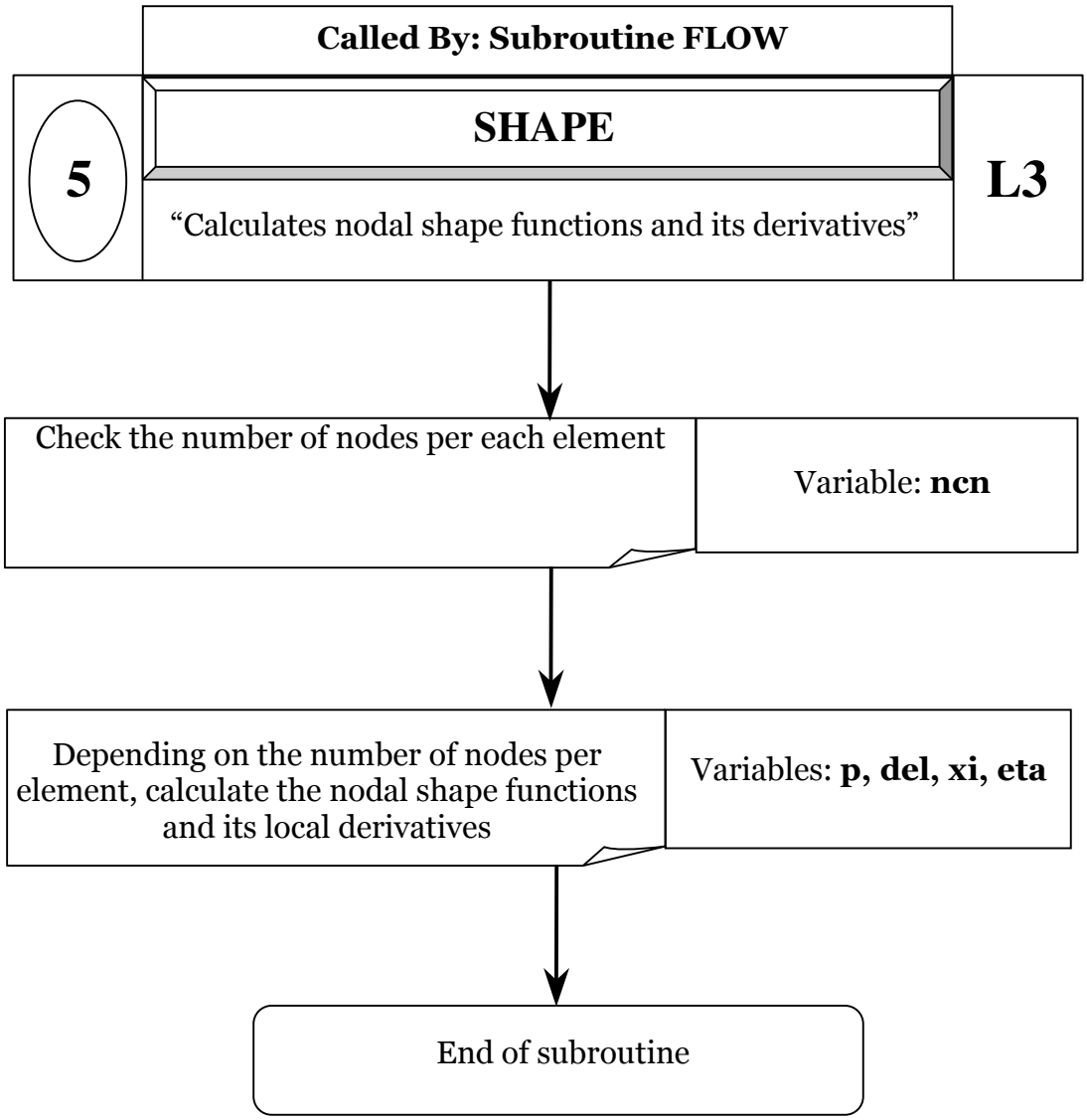
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE PERM, LEVEL 3



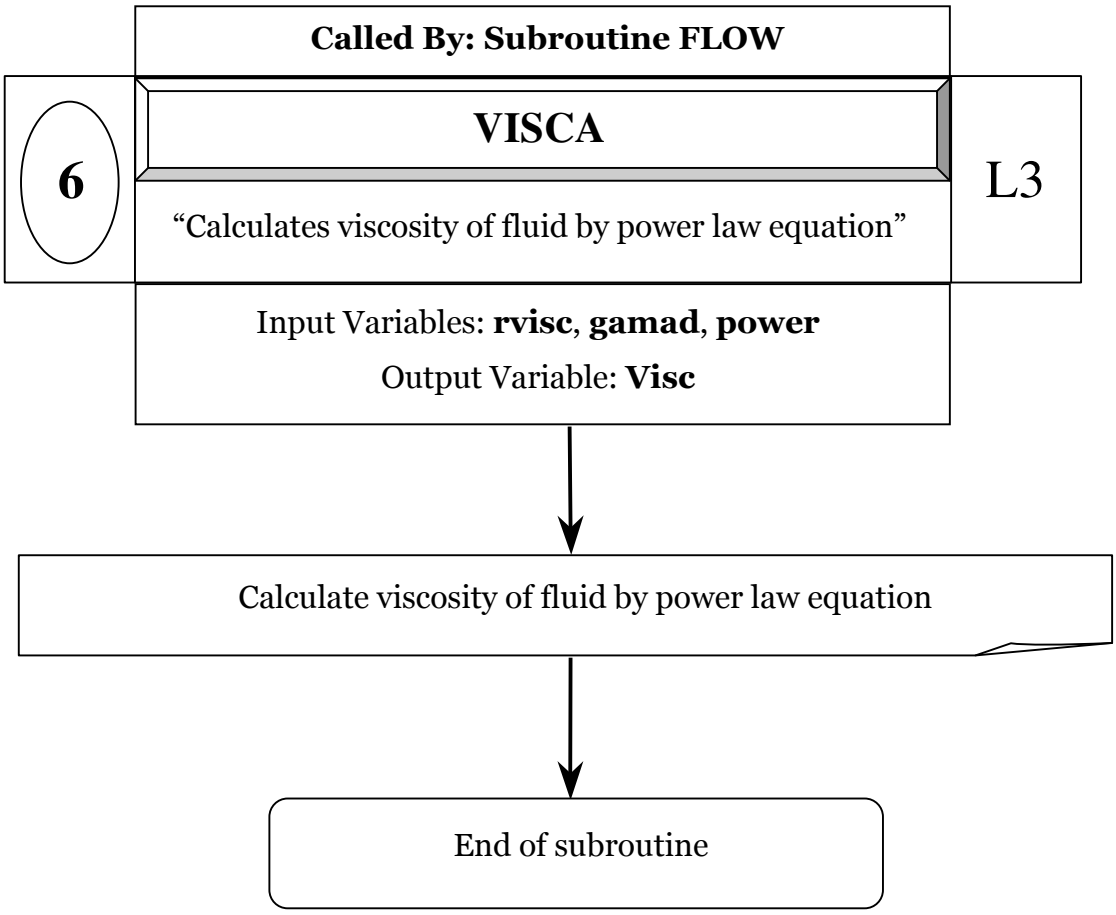
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE SHAPE, LEVEL 3



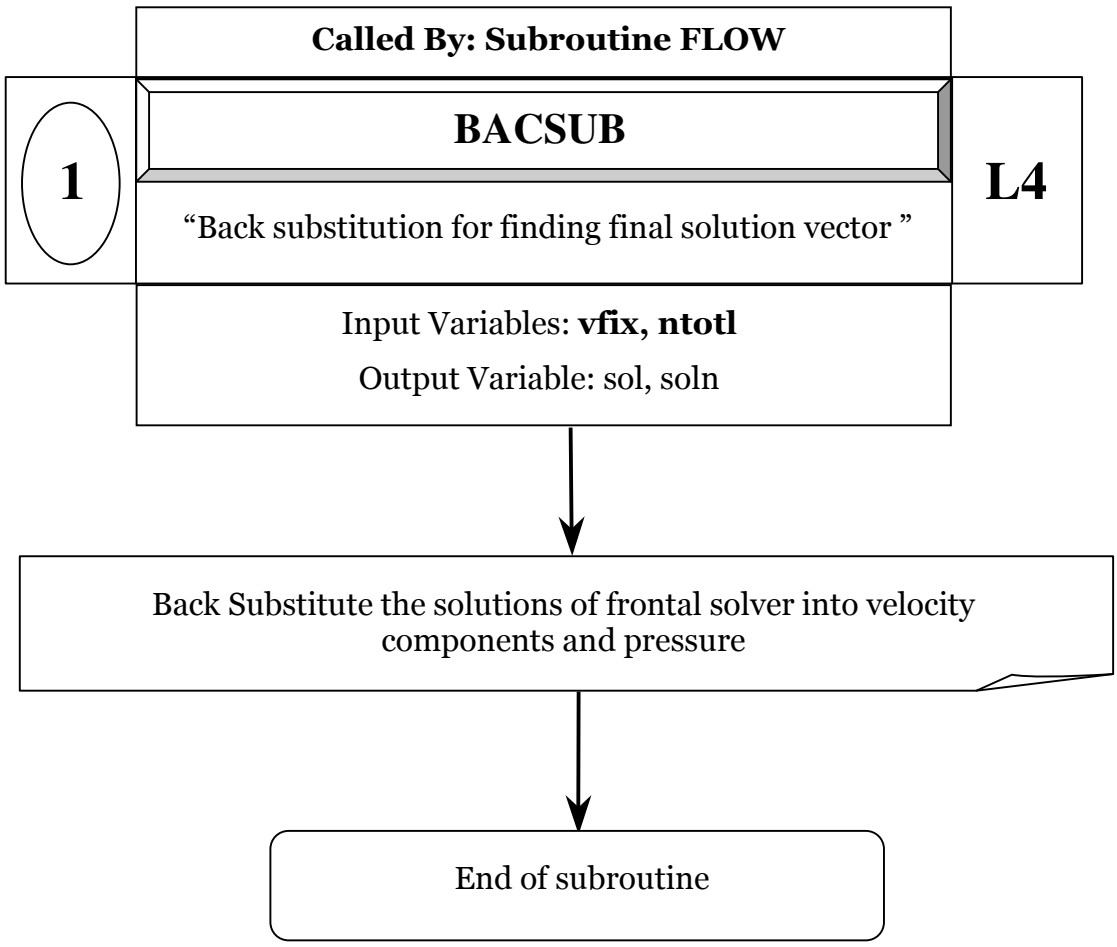
PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE VISCA, LEVEL 3



PROGRAM SUBROUTINES - INTERACTIONS

SUBROUTINE BACSUB, LEVEL 4



LISTING OF SOURCE CODES

1. ACFAMP
2. Auxilliary Program

```
c   ************************************************************
c
c   ACFAMP : Aircraft Cartridge Filter Analysis Modelling Program
c
c        Developed by  : Navraj S Hanspal & Atul N Waghode
c
c        Supervised by : Prof V.Nassehi & Prof. R.J.Wakeman
c
c                  Latest Version: 23rd May, 2005
c
c   ************************************************************
c   -----------
c   Description
c   -----------
c
c   Project Title: ! AEROFIL !
c
c   -------------------------
c   TECHNICAL SPECIFICATIONS
c   -------------------------
c
c   1. The program has been developed using a windows based FORTRAN 90 compiler.
c
c   2. This program stores the basic data required for simulating hydrodynamics
c      in pleated cartridge filters.
c
c   3. This program prompts the user to input the data in form of 3 separate data files.
c
c        (a) Default data file containing nodal connectivity: "df.dat"
c        (b) Finite element mesh GFM file
c        (c) Reference cartridge ID file generated by auxilliary program "filterdataprep.f"
c
c   4. This program is virtually linked with the auxilliary program which generates the
c      data file containing the geometrical specifications and the physical data.
c
c   5. This is a program for the solution of non-newtonian, isothermal,incompressible fluid
c      flow problems in porous media using the weighted residual galerkin finite
c      element method.
c
c   6. The solution scheme is based on the U-V-P method.
c
c   7. Velocity components and pressure are the prime unknowns in the flow field.
c
c   8. Pressure gradients inclusive and exclusive of compression and apparent
c      loss in filtration area effects can be determined.
c
c   9. Percentage compression and loss in are are also evaluted for a given
c      cartridge filter specification.
c
c  10. Algebraic equations are solved by a frontal method.
c
c  11. The program consists of a main module and subroutines.
c
c   ============================================================================================
c   --------------
c   main work files
c   --------------
c     unit         contents
c     ----         --------
c   ============================================================================================
c        1    i    GFM file for finite element computational mesh
c             i
c       14    i    used as a work file in the solver routine
c             i
c       15    i    stores shape functions and their derivatives at
c             i    'full' integration points
c             i
c       51    i    default data file
c             i
c       60    i    output file for documentation
c             i
c       61    i    output file for printing data in excel format
c             i
c      511    i    input data file containing cartridge characteristic data
c ============================================================================
c
c     =================
c     List of variables
c     =================
c     aa:             Left hand side finite element stiffness matrix
c     actpress:       Normalised value of nodal pressure
c     alpha:          Time-stepping parameter
```



```
c     ak:             Right hand side finite element stiffness matrix
c     areabase:       Percentage loss in area factor
c     b:              Measure of Gaussian integration
c     bc:             Array for storing boundary constraint value
c     cartrid:        Reference ID for each filter cartridge element
c     cg:             Weighting functions for Gauss integration
c     comfactor:      Compression factor for filter medium
c     cord:           Array storing coordinate values of each nodal point
c     da:             Global derivative of shape function
c     del:            Local derivatives of shape function
c     deltap:         Experimental pressure drop
c     deltat:         Value of time step used in calculation iterations
c     density:        Density of fluid used for each set of experiment
c     detj:           Determinant of Jacobian matrix
c     di:             Diameter of inner core of the filter cartridge element
c     el:             Length of pleat
c     enp:            Number of pleats in each filter cartridge element
c     errp:           Square of discrepancy between pressure values calculated
c                     in two successive iterations
c     errv:           Square of discrepancy between velocity values calculated
c                     in two successive iterations
c     eta:            y-coordinate in local coordinate system
c     excel:          Array for storing simulated values of pressure drop in
c                     each step  of calculation procedure
c     farea:          Filtering surface area of cartridge
c     frate:          Flow rate values for each experiment
c     gamad:          Rate of deformation of viscous stress tensor
c     guessp:         Pressure drop with compression and loss in area
c     hp:             Height of filter cartridge element
c     icord:          Switch for changing the coordinate system of
c                     equations
c     maxbc:          Limit for maximum number of boundary constraints
c                     that can be specified
c     maxdf:          Limit for maximum number of nodal degrees of freedom
c     maxel:          Limit for maximum number of elements in mesh
c     maxnp:          Limit for maximum number of nodal points in mesh
c     maxst:          Maximum dimension of finite element stiffness matrix
c     nbc:            Total number of boundary-node constraints
c     ncn:            Number of nodes associated with each finite element
c     ncod:           Array for constraint switch defined for every degree
c                     of freedom
c     ndf:            Total number of nodal degrees of freedom
c     ndim:           Dimensions of the solution domain
c     nel:            Total number of elements in pleated cartridge mesh
c     ngaus:          Number of quadrature points required for Gaussian integration
c     nmat:           Total number of fluids
c     nnp:            Total number of nodal points in pleated cartridge mesh
c     node:           Array for nodal connectivity of elements
c     ntd:            Number of sets of experiment
c     nter:           Number of solution iterations
c     num:            Number of integration points per element
c     p:              Finite element shape or interpolation functions
c     pd:             Experimental pressure value for each set of data
c     percomp:        Percentage value of filter medium compression
c     permstep1:      Permeability of filter medium without compression
c     permstep2:      Permeability of filter medium with compression
c     permx:          Permeability of the filter medium in x-direction
c     permy:          Permeability of the filter medium in y-direction
c     pmat:           Array storing values of physical properties of the fluid
c     power:          Power law index for the fluid viscosity
c     press:          Array for nodal pressure
c     pressexp:       Experimental pressure drop
c     rpef:           Reference pressure
c     rr:             Array for storing the elemental load vector
c     rtem:           Reference temperature
c     rvisc:          Consistency coefficient for viscosity of fluid
c     sinv:           Array storing components of second invariant of rate of
c                     deformation of stress tensor
c     soln:           Array for storing solution values calculated by Frontal Solver
c     step:           Stage of the calculation procedure
c     taco:           Coefficient a in the power law model
c     tbco:           Coefficient b in the power law model
c     time:           Amount of time for iterations
c     tolv:           Maximum allowable tolerance for fluctuation in velocity value
c     tolp:           Maximum allowable tolerance for fluctuation in pressure value
c     vel:            Array for nodal velocities
c     velcartridge:   Velocity of fluif through filter media
c     velsound:       Velocity of sound in fluid
c     viisc:          Viscosity of fluid used for each set of experiment
c     visc:           Updated value of viscosity according to power law model
c     xg:             Coordinates of Gauss quadrature points for integration
c     xi:             x-coordinate in local coordinate system
```

```
c
c     ================================================================================
c
c     ===================
c     List of Subroutines
c     ===================
c
c     arealoss            evaluates apparent percentage loss in filtration area
c     bacsub              backsubstitution method for finding the final
c                         solution vector
c     cartridge           reads the cartridge specification data
c     clean               cleans the arrays and prepares them for
c                         solution
c     contol              makes a check for the convergence
c     deriv               calculates the jacobian matrix, its determinant
c                         and global derivatives of the shape functions
c     excelprint          prints the simulated results in excel format
c     flow                calculates the velocities and pressures
c     front               frontal method for solving the final set
c                         of equations
c     gaussp              specifies the gauss points and weights for
c                         quadrature integration
c     getelm              specifies the nodal connectivity array
c     getmat              reads the input material data
c     getnod              reads the nodal co-ordinates for cartesian
c                         and axisymmetric systems
c     gfmfem              generates mesh co-ordinate files
c     hydrofilter         simulates 2-D hydrodynamics in cartridge filters
c     output              prints the final solution
c     perm                calculates permeability of filter cartridge media
c     pleatbc             generates inlet velocity and exit pressure boundary
c                         conditions
c     putbcv              imposes the primary boundary conditions for
c                         velocity
c     secinv              calculates second invariant of rate of deformation
c                         of stress tensor
c     setprm              Sets the location data for nodal degrees of
c                         freedom
c     shape               calculates the shape functions and their
c                         derivatives
c     visca               calculates the viscosity
c
c     ================================================================
      Program DarcyFILTER

      parameter (maxel   = 2500    )
      parameter (maxnp   =11000    )
      parameter (maxbc   = 2000    )
      parameter (maxdf   = maxnp*3 )
      parameter (maxst   = 27      )
      parameter (maxfr   = 1000    )
      parameter (ndim    = 3       )


      implicit real*8 (a-h,o-z)
c
c     Storage allocation
c     ==================
c
      dimension title (         80)
      dimension node  (maxel,   27) ,pmat (maxel,  9) ,cord (maxnp,  2)
      dimension ncod  (maxdf      ), bc   (maxdf    )
      dimension ibc   (maxbc      ) ,jbc  (maxbc    ) ,vbc  (maxbc    )
      dimension vel   (maxdf      ) ,press(maxnp    )
      dimension r1    (maxdf      ) ,sinv (maxel, 13)
      dimension vet   (maxdf      ) ,pet  (maxnp    )
      dimension nopp  (maxdf      )
      dimension aa    (maxst,maxst) ,rr   (maxst    )
      dimension xg    (           3) ,cg   (        3)
      dimension p     (           9) ,del  (    2,  9) ,b    (    2,  9)
      dimension ldest (maxst      ) ,kdest(maxst    ) ,nk   (maxst    )
      dimension eq    (maxfr,maxfr) ,lhed (maxfr    ) ,khed (maxfr    )
      dimension kpiv  (maxfr      ) ,lpiv (maxfr    ) ,jmod (maxfr    )
      dimension qq    (maxfr      ) ,pvkol(maxfr    )
      dimension mdf   (maxdf      ) ,ndn  (maxdf    )
      dimension ldsc  (22         ) ,excel(        6)
      dimension temp  (maxnp      ) ,actpress(maxnp )
      dimension fr    (20         ) ,velcartridge(20) ,pd   (20       )
      dimension viiisc(20         ) ,areabase(    20)

      character *20 filnam
      character *20 filnam1
```



```
c
c     Opening of input and output data files
c     ======================================
c
c     This data file contains the Numerical Parameters and Nodal Connectivity
c     -----------------------------------------------------------------------

      print *, "
     1* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *"
      print *, "
     1     *                                                           **
      print *, "
     1     *        #     ###### #######    #    #     # ######    **
      print *, "
     1     *       # #   #       #         # #   ##   ## #     #   **
      print *, "
     1     *      #   #  #       #        #   #  # # # # #     #   **
      print *, "
     1     *     #     # #       #####   #     # #  #  # ######    **
      print *, "
     1     *     ####### #       #       ####### #     # #         **
      print *, "
     1     *     #     # #       #       #     # #     # #         **
      print *, "
     1     *     #     #  ###### #       #     # #     # #         **
      print *, "
     1     *                                                           **
      print *, "
     1     *                                                           **
      print *, "
     1     *     May 2005                                              **
      print *, "
     1     *                                                           **
      print *, "
     1     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *"

      write(*, 1101)

      write(*, 1111)

      write(*, 1121)

      write(*, 1131)

      print *, "
     1    Program Developers: N.S Hanspal and A.N. Waghode"
      print *, "
     1  Project Advisors: Prof. R.J. Wakeman and Prof. V. Nassehi"
      print *, "
     1          Advanced Separation Technologies Group"
      print *, "
     1           Department of Chemical Engineering"
      print *, "
     1                Loughborough University"
      print *, "
     1             Loughborough, Leicestershire"
      print *, "
     1              LE11 3TU, United Kingdom"

      write(*, 1131)

      write (*, 1101)

c     ================================================================================


      print*,'Enter the name of DEFAULT data file'
      read(*,2000) filnam

      open(unit=51,File=filnam,access='sequential',form='formatted',
     1     status="unknown",iostat=ios)

c     This is the output file containing the results
c     ----------------------------------------------

      open(unit=60,file='aerout.txt',access='sequential',form='formatted',
     1     status="unknown",iostat=ios)

      open(unit=61,file='excel.txt',access='sequential',form='formatted',
     1     status="unknown",iostat=ios)

c     These are scratch files used in computation
```

```
c     ------------------------------------------

      open(unit=14,form='unformatted',status='scratch',iostat=ios)
      open(unit=15,form='unformatted',status='scratch',iostat=ios)

c     Checks for opening of files
c     ---------------------------

        if(ios==0)then
c         print*,"files opened"
        else
c         print*,"files not opened"
        stop
        end if

      rewind 20
      rewind 51
      rewind 60
      rewind 61

c
c    Initialize arrays
c    =================

      do 5010 itl = 1,maxel
      do 5010 ivl = 1,27
                        node (itl,ivl) = 0
5010  continue
      do 5020 itl = 1,maxel
      do 5020 ivl = 1,8
                        pmat (itl,ivl) = 0.0
5020  continue
      do 5030 itl = 1,maxnp
      do 5030 ivl = 1,2
                        cord (itl,ivl) = 0.0
5030  continue
      do 5050 itl = 1,maxdf
                        vel  (itl    ) = 0.0
5050  continue
      do 5060 itl = 1,maxdf
                        ncod   (itl)   = 0
                        r1     (itl)   = 0.0
                        bc     (itl)   = 0.0
                        vet    (itl)   = 0.0
                        mdf    (itl)   = 0
                        ndn    (itl)   = 0
                        nopp   (itl)   = 0
5060  continue
      do 5070 itl = 1,maxnp
                        pet    (itl)   = 0.0
                        press  (itl)   = 0.0
5070  continue
      do 5080 itl = 1,maxbc
                        ibc    (itl)   = 0
                        jbc    (itl)   = 0
                        vbc    (itl)   = 0.0
5080  continue
      do 5090 itl = 1,9
                        del    (1,itl) = 0.0
                        del    (2,itl) = 0.0
5090  continue
      do 5100 itl = 1,maxst
                        ldest (itl  )  = 0
                        kdest (itl  )  = 0
                        nk    (itl  )  = 0
5100  continue
      do 5110 itl = 1,maxfr
                        lhed  (itl  )  = 0
                        khed  (itl  )  = 0
                        kpiv  (itl  )  = 0
                        lpiv  (itl  )  = 0
                        jmod  (itl  )  = 0
                        qq    (itl  )  = 0.0
                        pvkol (itl  )  = 0.0
      do 5110 ill = 1,maxfr
                        eq     (itl,ill)= 0.0
5110  continue

c     ============================================================
c     Title of the program
c     ====================
```



```
      if(.not. eof(51)) read (51,2010) title
      write(60,4010) title

c     Element description data
c     ========================

      if (.not. eof(51)) read (51,2020) ncn ,ngaus
      write(60,4020) ncn ,ngaus

c     Mesh, boundary condition and material parameters
c     ===============================================

      if (.not. eof(51)) read (51,2030) nnp  ,nel  ,nbc  ,nmat
      if (.not. eof(51)) read (51,2040) ntep ,icord
c
      if(icord.eq.0) write(60,4030)
      if(icord.eq.1) write(60,4040)

      write(60,4050)

      if(ntep.eq.0) ntep=1

c     ============================================================
c     icord=0 coordinate system is cartesian   (   planar   )
c     icord=1 coordinate system is cylindrical (axisymmetric)
c
c     if ntep = 1 then computed result after every iteration will
c     be printed ;if you do not need the result of intermediate
c     computations choose your own ntep;the result of first and
c     converged solutions will always be printed.
c     ============================================================

           if(nnp   .eq.0  .or.nnp  .gt.maxnp) then
                                    write(60,4060)
      elseif(nel   .eq.0  .or.nel  .gt.maxel) then
                                    write(60,4060)
      elseif(nbc   .eq.0  .or.nbc  .gt.maxbc) then
                                    write(60,4060)
      elseif(nmat  .eq.0  .or.nmat .gt.maxel) then
                                    write(60,4060)

      print*, "the program is aborted"
      stop

           endif

      write(60,4070) nnp ,nel ,nbc ,nmat

      if (.not. eof(51)) read(51,2060) tolv ,tolp

      maxer=maxel

c     ============================================================================
c     Read input data from data files and prepare arrays for solution process
c     ============================================================================

c       Reads material properties
c       -------------------------
        call getmat(nel,nmat,pmat,51,60,maxel,rtem,rpef)

c       Conversion of GFM file (from GEOSTAR) to MESH.FEM file
c       ------------------------------------------------------
        call gfmfem

c       Read the nodal co-ordinates from MESH.FEM file
c       ----------------------------------------------
        call getnod(nnp,cord,3,60,maxnp,ndim,icord)

c       Read the Nodal Connectivity
c       ---------------------------
        call getelm(nel,ncn,node,51,60,maxer)

c       Read experimental data for catridge filter element ID file
c       ----------------------------------------------------------
        call cartridge (ntd,density,fr,velcartridge,farea,viiisc,
     1                  areabase,xab,yab,xjkl,yjkl,xfgfd,yfgfd,
     2                  xldkdjd,yldkdjd,xadbd,yadbd)


c     ============================================================================
c                    Loop for Different Flow Rates
c     ============================================================================
```

```
      print *, "Total Number of Data sets for Analysis = ", ntd

      do 7001 iiii=1,ntd

      print *, "Loss in area factor = ", areabase(iiii)


c      Flow rate (lit/min) and corresponding loss in area factor
c      ----------------------------------------------------------
       frate    = fr(iiii)
       ndata    =    iiii

c      Corresponding viscosity for the flow rate (Pa.s)
c      ------------------------------------------------

       do 6021 iel   = 1, nel
         pmat(iel,1) = viiisc(iiii)
         pmat(iel,8) = density
6021   continue


c    ========================================================================
c                          Start of the time loop
c    ========================================================================
c
c     Set control parameters (default values are overwritten by input data
c     if specified)
c
c     ncn       number of nodes per element
c     ngaus     number of integration points
c     nter      maximum number of iterations for non-newtonian case
c     ndim      number of space dimensions in the solution domain
c
      nter = 5
      num  = 13
                                      do 5125 iel = 1,maxel
                                      do 5125 lg  = 1,num
                                      sinv(iel,lg)= 0.00
5125                                  continue
                                      do 5140 item= 1,maxnp
                                      temp(item)  =    rtem
5140                                  continue


c    ================================================================================================
============
c     HYDRODYNAMIC SIMULATION 2-D PLEAT USING COMPRESSED PERMEABILITY FOR ESTIMATION OF COMPRES
SION FACTORS
c    ================================================================================================
============

      step = 2

      call pleatbc
     1   (frate,farea,nbc,ibc,jbc,vbc,maxbc,maxnp,ndim,cord,step,
     2    areafactor,xab,yab,xjkl,yjkl,xfgfd,yfgfd,xldkdjd,yldkdjd,
     3    xadbd,yadbd)


      call hydrofilter
     1(nter,ndata,deltat,icho,nnp,ncn,nel,maxel,maxst,maxdf,bc,ncod,
     2    rl,node,ndn,ntrix,mdf,nopp,nbc,ibc,jbc,vbc,maxbc,cord,step,
     3    pmat,vel,press,temp,ldest,kdest,nk,eq,lhed,khed,kpiv,ntep,
     4    lpiv,jmod,qq,pvkol,ngaus,p,del,b,maxfr,ndim,aa,xg,da,num,
     5    icord,rr,alpha,sinv,maxnp,actpress,errov,errop,vet,pet,
     6    comfactor,percomp,pressexp,permstep1,permstep2,velcartridge,
     7     areabase,density,excel)

c    ================================================================================================
c     PRINT ALL THE DESIGN PARAMETERS IN A SEPERATE FILE WHICH CAN BE OPENED IN MICROSOFT EXCEL

c    ================================================================================================

      call excelprint (iiii, excel, fr, pd, viiisc)

7001  end do

      close(unit =   1)
      close(unit =   2)
      close(unit =   3)
      close(unit =  14)
```



```
      close(unit =  15)
      close(unit =  20)
      close(unit =  51)
      close(unit =  61)
      close(unit =  70)
      close(unit = 511)

c    ========================================================================
c                          print statements
c    ========================================================================

1101  format(' ',/)
1111  format (/'  ACFAMP calcualtes percentage compression of the
     1permeable media  ',/' and predicts pressure drop across pleated
     1cartridge filters ',/' including effects of compression and losses
     1 in effective filtration area')
1121  format(' ', 2(/))

1131  format("  ************************************************
     1*****************")

c    ========================================================================
c                          Read statements
c    ========================================================================
c
2000  format(a)
2010  format(80a)
2020  format(2i5)
2030  format(4i5)
2040  format(2i5)
2050  format(2f10.0)
2060  format(3f10.5)
c
c    ========================================================================
c                          Write statements
c    ========================================================================

4010  format(' ',5(/),' ',20x,60('*'),/' ',20x,'*',58x,'*',/
     1' ',20x,'*',' A  two  dimensional finite element model of a  ',
     29x,'*',/' ',20x,'*',' non-newtonian isothermal flow using ',
     320x,'*',/' ',20x,'*',' the UVP method.  ',39x,'*',/' ',20x,'*',
     558x,'*',/' ',20x,60('*')///,' ',20x,80('-'),/' ',20x,80a,/' ',
     620x,80('-'),///)
c
4020  format(' ',20x,3('['),' element description data',10('.'),/
     125x,'no.of nodes per element                    =',i10,/
     225x,'no.of integration points                   =',i10,/
     3//)
c
4030  format('    *** coordinate system is cartesian (planar) ***')
4040  format('*** coordinate system is cylindrical(axisymmetric) ***')
4050  format(' ')
4060  format(' ',10('['),'input data unacceptable',10(']')///)
c
4070  format(' ',20x,3('['),' mesh description data ',10('.'),/
     125x,'no.of nodal points                         =',i10,/
     225x,'no.of elements                             =',i10,/
     325x,'no.of nodal constraints on boundary        =',i10,/
     425x,'no.of different materials                  =',i10,//)
c
4090  format(///'TIME STEP  no.',i5,//)
c4100  format(i5,2e13.4)
c4110  format(5e13.4)
c
c     ========================================================================
      end program
c     ========================================================================
c     ========================================================================
c     e    n    d            o    f            p    r    o    g    r    a    m
c     ========================================================================
c
c
c
c
c
c     --------------------
c     START OF SUBROUINTES
c     --------------------
c     ========================================================================
      subroutine gaussp(ngaus,xg,cg)
c
      implicit double precision(a-h,o-z)
```

```
c
c     x(g)   specifies the coordinates of the Gauss points
c     c(g)   specifies the Gauss weights
c
      dimension xg(3),cg(3)

      if(ngaus.eq.1) then
      xg(1)=0.0
      cg(1)=2.0
      elseif(ngaus.eq.2)  then
      xg(1) = 0.57735026919d00
      xg(2) = -xg(1)
      cg(1) = 1.00
      cg(2) = 1.00
      else
      xg(1) = 0.77459666924d00
      xg(2) = 0.0
      xg(3) = -xg(1)
      cg(1) = 0.555555555556d00
      cg(2) = 0.88888888889d00
      cg(3) = cg(1)
c
      endif
      return
      end
c
c     ================================================================
      subroutine shape ( xi , eta ,  p ,del , ncn )
      implicit double precision (a-h,o-z)
c
      DIMENSION p(9) ,del(2,9)
      if (ncn.eq.4) then
         del(1,1)=-0.25*(1-eta)
         del(1,2)= 0.25*(1-eta)
         del(1,3)= 0.25*(1+eta)
         del(1,4)=-0.25*(1+eta)
C.................................
         del(2,1)=-0.25*(1-xi)
         del(2,2)=-0.25*(1+xi)
         del(2,3)= 0.25*(1+xi)
         del(2,4)= 0.25*(1-xi)
C.................................
         p(1)=0.25*(1-xi)*(1-eta)
         p(2)=0.25*(1+xi)*(1-eta)
         p(3)=0.25*(1+xi)*(1+eta)
         p(4)=0.25*(1-xi)*(1+eta)
      elseif (ncn.eq.8) then
         del(1,1)=0.5*xi-0.5*xi*eta-0.25*eta**2+0.25*eta
         del(1,2)=0.5*xi-0.5*xi*eta+0.25*eta**2-0.25*eta
         del(1,3)=0.5*xi+0.5*xi*eta+0.25*eta**2+0.25*eta
         del(1,4)=0.5*xi+0.5*xi*eta-0.25*eta**2-0.25*eta
         del(1,5)=xi*(-1+eta)
         del(1,6)=0.5-0.5*eta**2
         del(1,7)=-xi*(1+eta)
         del(1,8)=-0.5+0.5*eta**2
C.................................
         del(2,1)=0.5*eta-0.25*xi**2-0.5*xi*eta+0.25*xi
         del(2,2)=0.5*eta-0.25*xi**2+0.5*eta*xi-0.25*xi
         del(2,3)=0.5*eta+0.25*xi**2+0.5*xi*eta+0.25*xi
         del(2,4)=0.5*eta+0.25*xi**2-0.5*xi*eta-0.25*xi
         del(2,5)=-0.5+0.5*xi**2
         del(2,6)=-(1+xi)*eta
         del(2,7)=0.5-0.5*xi**2
         del(2,8)=(-1+xi)*eta
C.................................
         p(1)=0.25*(1-xi)*(1-eta)*(-1-xi*eta)
         p(2)=0.25*(1+xi)*(1-eta)*(-1+xi-eta)
         p(3)=0.25*(1+xi)*(1+eta)*(-1+xi+eta)
         p(4)=0.25*(1-xi)*(1+eta)*(-1-xi+eta)
         p(5)=0.5*(1-xi**2)*(1-eta)
         p(6)=0.5*(1+xi)*(1-eta**2)
         p(7)=0.5*(1-xi**2)*(1+eta)
         p(8)=0.5*(1-xi)*(1-eta**2)
      elseif (ncn.eq.9) then
         del(1,1)=0.25*(2*xi-1)*(eta**2-eta)
         del(1,2)= -xi*(eta**2-eta)
         del(1,3)=0.25*(2*xi+1)*(eta**2-eta)
         del(1,4)=0.5 *(2*xi+1)*(1-eta**2)
         del(1,5)=0.25*(2*xi+1)*(eta**2+eta)
         del(1,6)=-xi *(eta**2+eta)
         del(1,7)=0.25*(2*xi-1)*(eta**2+eta)
         del(1,8)=0.5 *(2*xi-1)*(1-eta**2)
```



```
         del(1,9)=-2  *xi*(1-eta**2)

C.................................
         del(2,1)=0.25*(xi**2-xi)*(2*eta-1)
         del(2,2)=0.5 *(1-xi**2)*(2*eta-1)
         del(2,3)=0.25*(xi**2+xi)*(2*eta-1)
         del(2,4)=-eta*(xi**2+xi)
         del(2,5)=0.25*(xi**2+xi)*(2*eta+1)
         del(2,6)=0.5 *(1-xi**2)*(2*eta+1)
         del(2,7)=0.25*(xi**2-xi)*(2*eta+1)
         del(2,8)=-eta*(xi**2-xi)
         del(2,9)=-2  *eta*(1-xi**2)
C.................................
         p(1)=0.25*(xi**2-xi)*(eta**2-eta)
         p(2)=0.5*(1-xi**2)*(eta**2-eta)
         p(3)=0.25*(xi**2+xi)*(eta**2-eta)
         p(4)=0.5*(xi**2+xi)*(1-eta**2)
         p(5)=0.25*(xi**2+xi)*(eta**2+eta)
         p(6)=0.5*(1-xi**2)*(eta**2+eta)
         p(7)=0.25*(xi**2-xi)*(eta**2+eta)
         p(8)=0.5*(xi**2-xi)*(1-eta**2)
         p(9)=(1-xi**2)*(1-eta**2)

      endif
      return
      end
c     ================================================================
c
      subroutine deriv
     1    (iel  ,ig   ,jg   ,p    ,del  ,b    ,ncn   ,da   ,cg   ,node,
     2     cord ,maxel,maxnp,maxst)
c
      implicit double precision(a-h,o-z)
      dimension p(9),b(2,9),del(2,9),cg(3),cj(2,2),cji(2,2)
      dimension node(maxel,27),cord(maxnp,2)
c
      do 6010 j=1,2
      do 6010 l=1,2
      gash=0.0
      do 6020 k=1,ncn
      a=iabs(node(iel,k))
6020  gash=gash + del(j,k)*cord(a,l)
6010  cj(j,l)=gash
      detj=cj(1,1)*cj(2,2)- cj(1,2)*cj(2,1)
      if(detj.le.0.0) then
      write(60,3010) iel,detj
3010  format(1x ,' Error: Zero or Negative Jacobian. ', i6,g20.5)
      stop
      endif
c
      cji(1,1) = cj(2,2) / detj
      cji(1,2) =-cj(1,2) / detj
      cji(2,1) =-cj(2,1) / detj
      cji(2,2) = cj(1,1) / detj
c
      do 6030 j=1,2
      do 6030 l=1,ncn
      b(j,l)=0.0
      do 6030 k=1,2
6030  b(j,l) = b(j,l) + cji(j,k) * del(k,l)

      da= detj*cg(ig)*cg(jg)
c
      return
      end

c     ================================================================
c
      subroutine front
     1    (aa   ,rr   ,iel  ,nop  ,maxel,maxst,ldest,kdest,nk   ,maxfr,
     2     eq   ,lhed ,khed ,kpiv ,lpiv ,jmod ,qq   ,pvkol,vel  ,rl   ,
     3     ncod ,bc   ,nopp ,mdf  ,ndn  ,maxdf,nel  ,maxte,ntov ,lcol ,
     4     nell ,ntra, press )
c
c     Frontal elimination routine using diagonal pivoting
c
      implicit double precision(a-h,o-z)
      dimension aa   (maxst,maxst) ,rr   (maxst)
      dimension nop  (maxel,maxst)
      dimension ldest(maxst)       ,kdest(maxst)       ,nk   (maxst)
      dimension eq   (maxfr,maxfr) ,lhed (maxfr)       ,khed (maxfr)
      dimension kpiv (maxfr)       ,lpiv (maxfr)
```

```
      dimension jmod (maxfr)      ,qq   (maxfr)      ,pvkol(maxfr)
      dimension vel  (maxte)      ,r1   (maxdf)      ,ncod (maxdf)
      dimension bc   (maxdf)      ,nopp (maxdf)      ,mdf  (maxdf)
      dimension ndn  (maxdf)      ,press(maxdf)
c
c     nlp and ndl are the file specifiers for units 60 and 14 respectively
c     ======================================================================

      nlp=60
      ndl=14
c
c     Prefront
c     ========
      nmax=maxfr
      ncrit=20
      nlarg=maxfr-10
      if(iel.eq.1) nell = 0
      if(iel.eq.1) ntra = 1
      if(ntra.eq.0) goto 6040
      nmax = maxfr
      ntra = 0
      ncrit = 20
      lfron = 0
      nlarg = nmax-10
c
c     Find last appeareance of each node
c     ==================================
      nlast = 0
      do 6010 i = 1,ntov
      do 6020 n = 1,nel
      jdn = ndn(n)
      do 6030 l = 1,jdn
      if(nop(n,l).ne.i)go to 6030
      nlastl = n
      nlast = n
      ll = l
6030  continue
6020  continue
      if(nlast.eq.0) go to 6010
      nop(nlast,ll) = -nop(nlast,ll)
      nlast = 0
6010  continue
      ntrix = jdn
c
c     Assembly
c     ========
6040  continue
      if(iel.gt.1) go to 6060
      lcol = 0
      do 6050 i = 1,nmax
      do 6050 j = 1,nmax
      eq(j,i) = 0.
6050  continue
6060  nell = nell+1
      n = nell
      jdn = ndn(nell)
      kc = 0
      do 6070 j = 1,jdn
      nn = nop(n,j)
      m = iabs(nn)
      k = nopp(m)
      idf = mdf(m)
      r1(m) = rr(j)+r1(m)
      do 6070 l = 1,idf
      kc = kc+1
      ii = k+l-1
      if(nn.lt.0)ii = -ii
      nk(kc) = ii
6070  continue
c
c     Set up heading vectors
c     ======================
c
      do 6080 lk = 1,kc
      node = nk(lk)
      if(lcol.eq.0)goto 6100
      do 6090 l = 1,lcol
      ll = l
      if(iabs(node).eq.iabs(lhed(l)))go to 6110
6090  continue
6100  lcol = lcol+1
      ldest(lk) = lcol
```



```
      lhed(lcol) = node
      go to 6080
6110  ldest(lk) = ll
      lhed(ll) = node
6080  continue
      if(lcol.le.nmax)go to 6130
      nerror = 2
      write(nlp,3010)nerror
      stop
6130  continue
      do 6140 l = 1,kc
      ll = ldest(l)
      do 6140 k = 1,kc
      kk = ldest(k)
      eq(kk,ll) = eq(kk,ll)+aa(k,l)
6140  continue
      if(lcol.lt.ncrit.and.nell.lt.nel) return
c
c     Find out which matrix elements are fully assembeled
c     ===================================================
6150  lc = 0
      ir = 0
      do 6160 l = 1,lcol
      kt = lhed(l)
      if(kt.ge.0)go to 6160
      lc = lc+1
      lpiv(lc) = l
      kro = iabs(kt)
      if(ncod(kro).ne.1)go to 6160
      ir = ir+1
      jmod(ir) = l
      ncod(kro) = 2
      rl(kro) = bc(kro)
6160  continue
c
c     Modify equations with applied boundary conditions
c     =================================================
      if(ir.eq.0)go to 6190
      do 6170 irr = 1,ir
      k = jmod(irr)
      kh = iabs(lhed(k))
      do 6180 l = 1,lcol
      eq(k,l) = 0.
      lh = iabs(lhed(l))
      if(lh.eq.kh)eq(k,l) = 1.
6180  continue
6170  continue
6190  continue
      if(lc.gt.0)go to 6200
      ncrit = ncrit+10
c     write(nlp,3020)ncrit
      if(ncrit.le.nlarg) return
      nerror = 3
      write(nlp,3030)nerror
      stop
6200  continue
c
c     Search for absolute pivot
c     =========================
      pivot = 0.
      do 6210 l = 1,lc
      lpivc = lpiv(l)
      kpivr = lpivc
      piva = eq(kpivr,lpivc)
      if(abs(piva).lt.abs(pivot))go to 6220
      pivot = piva
      lpivco = lpivc
      kpivro = kpivr
6220  continue
6210  continue
      if(pivot.eq.0.0) return
c
c     Normalise pivotal row
c     =====================
      lco = iabs(lhed(lpivco))
      kro = lco

c     if(nit.eq.0.or.npra.eq.0)go to 6230
c6230 continue

      if(abs(pivot).lt.0.1d-28) write(nlp,3050)
      do 6240 l = 1,lcol
```

```
      qq(l) = eq(kpivro,l)/pivot
 6240 continue
      rhs = r1(kro)/pivot
      r1(kro) = rhs
      pvkol(kpivro) = pivot
c
c     Eliminate then delete pivotal row and column
c     ============================================
      if(kpivro.eq.1)go to 6300
      kpivr = kpivro-1
      do 6250 k = 1,kpivr
      krw = iabs(lhed(k))
      fac = eq(k,lpivco)
      pvkol(k) = fac
      if(lpivco.eq.1.or.fac.eq.0.)go to 6270
      lpivc = lpivco-1
      do 6260 l = 1,lpivc
      eq(k,l) = eq(k,l)-fac*qq(l)
 6260 continue
 6270 if(lpivco.eq.lcol)go to 6290
      lpivc = lpivco+1
      do 6280 l = lpivc,lcol
      eq(k,l-1) = eq(k,l)-fac*qq(l)
 6280 continue
 6290 r1(krw) = r1(krw)-fac*rhs
 6250 continue
 6300 if(kpivro.eq.lcol)go to 6360
      kpivr = kpivro+1
      do 6310 k = kpivr,lcol
      krw = iabs(lhed(k))
      fac = eq(k,lpivco)
      pvkol(k) = fac
      if(lpivco.eq.1)go to 6330
      lpivc = lpivco-1
      do 6320 l = 1,lpivc
      eq(k-1,l) = eq(k,l)-fac*qq(l)
 6320 continue
 6330 if(lpivco.eq.lcol)go to 6350
      lpivc = lpivco+1
      do 6340 l = lpivc,lcol
      eq(k-1,l-1) = eq(k,l)-fac*qq(l)
 6340 continue
 6350 r1(krw) = r1(krw)-fac*rhs
 6310 continue
 6360 continue
c
c     Write pivotal equation on disc
c     ==============================
      write(nd1) kro,lcol,lpivco,(lhed(l),qq(l),l = 1,lcol)
      do 6370 l = 1,lcol
      eq(l,lcol) = 0.
      eq(lcol,l) = 0.
 6370 continue
c
c     Rearrange heading vectors
c     =========================
      lcol = lcol-1
      if(lpivco.eq.lcol+1)go to 6390
      do 6380 l = lpivco,lcol
      lhed(l) = lhed(l+1)
 6380 continue
 6390 continue
c
c     Determine whether to assemble,eliminate,or backsubstitute
c     =========================================================
      if(lcol.gt.ncrit)go to 6150
      if(nell.lt.nel) return
      if(lcol.gt.1)go to 6150
      lco = iabs(lhed(1))
      kpivro = 1
      pivot = eq(1,1)
      kro = lco
      lpivco = 1
      qq(1) = 1.

c     if(nit.eq.0.or.npra.eq.0)go to 6400
c     write(nlp,3040)lco,kro,pivot

      if(abs(pivot).lt.1d-28)go to 6410

c6400 continue
```



```
      r1(kro) = r1(kro)/pivot
      write(nd1) kro,lcol,lpivco,lhed(1),qq(1)
c
c     start back-substitution
c     =======================
      call bacsub
     1    (ntov ,ncod ,bc   ,r1   ,vel  ,press, maxfr,qq   ,lhed ,nd1)
c
c     main exit with solution
c     =======================
 6410 continue
c
 3010 format(/'  nerror=',i5//
     1 '  the difference nmax-ncrit is not sufficiently large'
     1/'  to permit the assembly of the next element---'
     1/'  either increase nmax or lower ncrit'
     1/)
c3020 format('  frontwidth value=',i4)
 3030 format(/'  nerror=',i5//
     1 '  there are no more rows fully summed,this may be due to---'
     1/'  (1)incorrect coding of nop or nk arrays'
     1/'  (2)incorrect value of ncrit. increase ncrit to permit'
     1/'     whole front to be assembled'
     1/)

c3040 format(13h pivotal row=,i4,16h pivotal column=,i4,7h pivot=,e20.10
c    1)

 3050 format('  warning-matrix singular or ill conditioned')

      return
      end
c
c     ================================================================
      subroutine bacsub
     1    (ntot1,ifix ,vfix ,rhs  ,soln ,soln1, mfrnt,rwork,iwork,idv2 )
c
c
      implicit double precision(a-h,o-z)
      dimension ifix (ntot1),vfix (ntot1),rhs (ntot1),soln (ntot1)
      dimension rwork(mfrnt) ,iwork(mfrnt) ,soln1(ntot1/3)
c
c
c
                        do 6010 ipos=1,ntot1
                        soln(ipos) =0.0
                        if(ifix(ipos).ne.0) soln(ipos)=vfix(ipos)
6010                    continue



      do 6020 kpos=1,ntot1
c
      backspace idv2
      read(idv2)     ipos,ifrnt,jfrnt,(iwork(k),rwork(k),k=1,ifrnt)
      backspace idv2
c
      if(ifix(ipos).ne.0)  go to 6020
c
      ww           = 0.0
      rwork(jfrnt) = 0.0
c
                        do 6030 k=1,ifrnt
                        jpos=iabs(iwork(k))
                        ww  =ww - rwork(k)*soln(jpos)
6030                    continue
c
      soln (ipos)=rhs(ipos)+ww

6020  continue

      do  6040 ipos  = ((2*ntot1)/3)+1 , ntot1
            j        = ipos -((2*ntot1)/3)
          soln1(j) = soln(ipos)

6040  continue

      return
      end
c
c     ================================================================
      subroutine flow(iit,node,cord ,pmat ,nopp ,mdf ,ndn ,ncod ,bc ,vel
```

```
     1,press, rl, temp,ldest,kdest,nk ,eq ,lhed ,khed ,kpiv ,lpiv,
     2jmod, qq, pvkol, iter ,nel ,ncn , ngaus, p, del, b, ntrix, maxel,
     3maxnp, maxst, maxfr, maxdf ,ndim ,aa ,xg ,da ,ntov ,num, icord,
     4rr, iel, deltat,alpha,idv4,sinv,nnp,step, permx, permy,guessp,
     5comfactor,percomp,permstep1,permstep2)
c
      implicit double precision(a-h,o-z)
c
      dimension node (maxel,maxst),pmat (maxel,   9),cord (maxnp, ndim)
      dimension ncod (maxdf      ),bc   (maxdf     ),sinv (maxel,   13)
      dimension vel  (nnp,   ndim) ,rl  (maxdf     )
      dimension aa   (maxst,maxst),rr   (maxst     ),ldest(maxst      )
      dimension xg   (          3),cg   (         3),kdest(maxst      )
      dimension x    (          2),v    (         2),nk    (maxst     )
      dimension bicn (          2),hh   (         2)
      dimension p    (          9),del  (    2,   9),b     (    2,    9)
      dimension eq   (maxfr,maxfr),nopp (maxdf     )
      dimension ldsc (         22)
      dimension lhed (maxfr      ),khed (maxfr     ),jmod (maxfr      )
      dimension lpiv (maxfr      ),kpiv (maxfr     ),qq   (maxfr      )
      dimension pvkol(maxfr      ),mdf  (maxdf     ),ndn  (maxdf      )
      dimension ppp  (9     ,    9),pp   (9         )
      dimension ak   (maxst,maxst),akf  (maxst     )
      dimension NQ   (9     ,    9),NP   (3    ,   4)
      dimension C    (maxst      ),temp (maxnp     )
      dimension press(maxnp      ),SHAPE1D(3        )
      dimension gdsf (    2,    9)
      dimension dmass(maxst,maxst)
c
                                              rvisc = pmat(iel,1)
                                              rpef  = pmat(iel,2)
                                              power = pmat(iel,3)
                                              rtem  = pmat(iel,4)
                                              tbco  = pmat(iel,5)
                                              taco  = pmat(iel,6)
                                              roden = pmat(iel,8)
                                              gamad = pmat(iel,9)

      velsound = 1150.0

      call perm
     1(step, permx, permy,guessp,comfactor,percomp,permstep1,permstep2)


c     if(iit == 1) then
c         permstep1 = permx
c         permstep2 = permstep1

c     else if (iit.gt.1) then
c         permstep2 = permx

c     end if

c     percomp = (permstep1-permstep2)*100/permstep1

c     if(iel == 1) then
c         print *, "permstep1 = ", permstep1
c         print *, "permstep2 = ", permstep2
c         print *, "permcomp = ", percomp
c     end if


                do 6010 idf= 1,ntrix
                rr (idf)       = 0.0
                akf(idf)       = 0.0
                C  (idf)       = 0.0
                do 6010 jdf= 1,ntrix

                aa   (idf,jdf)=0.0
                dmass(idf,jdf)=0.0
                ak   (idf,jdf)=0.0
6010            continue
c

      call gaussp(ngaus,xg,cg)

            lg=0
      do 6020 ig=1,ngaus
                                    g = xg(ig)
      do 6020 jg=1,ngaus
                                    h = xg(jg)
```



```
                                    lg = lg + 1


      if(iter.eq.1) then

c
      call shape (g,h,p,del,ncn)
c
      call deriv (iel,ig,jg,p,del,b,ncn,da,cg,node,cord,
     1            maxel,maxnp,maxst)

                  iig=ig
                  jjg=jg

                  write(15)   iel ,ig ,jg ,p ,del ,b ,da
      else
      if(.not. EOF(15))read(15) iel,iig,jjg,p ,del ,b , da
c
      endif
c
c     calculation of viscosity based on the constitutive equation.
c
      spress = 0.0
      stemp  = 0.0
c

            do 5333 ip = 1,ncn
            jp = iabs(node(iel,ip))
 5333       continue
            epsii = 1.d-10
            gamad = sinv(iel,lg)
            if(srate.lt.epsii) srate = epsii
c
      call visca
     1(rvisc,power,visc,stemp,rtem,tbco,spress,rpef,taco,gamad)
c

c     prepration of the convective acceleration terms/balancing
c     dissipation is used
c
                  do 6050 idff= 1,2
                  x(idff)     = 0.0
                  v(idff)     = 0.0
                  hh(idff)    = 0.0
6050              continue
          do 6060 icn = 1 ,ncn
                  jcn = iabs(node(iel,icn))
          do 6060 idff= 1 ,  2
          x(idff)     = x(idff) + p(icn)*cord(jcn,idff)
          v(idff)     = v(idff) + p(icn)*vel (jcn,idff)
6060      continue


      if(icord.eq.1) then
c
c     modify da for axisymmetric computations.
c
      da = da * x(1)
      endif


c     column index

      do 6070 i=1,ncn

                  j11= i
                  j12= i + ncn
                  j13= i + 2*ncn


      do 6070 j=1,ncn
                  j21= j
                  j22= j + ncn
                  j23= j + 2*ncn


c     ---------------------------------
c     DISCRETISED FORM OF DARCY EQUATION
c     ---------------------------------
c     Stiffness Matrix of Left Hand Side
c     ==================================
```

```
c     For Transient state (Cartesian co-ordinate system)
c     ==================================================

      aa(j11,j21)=aa(j11,j21) +  p(i)*p(j)*da
     1                        +  (alpha*deltat*visc*p(i)*p(j)*da)
     2                           /(roden*permx)

      aa(j11,j22)=aa(j11,j22) +  0.0

      aa(j11,j23)=aa(j11,j23) -  alpha*deltat*b(1,i)*p(j)*da

      aa(j12,j21)=aa(j12,j21) +  0.0

      aa(j12,j22)=aa(j12,j22) +  p(i)*p(j)*da
     5                        +  (alpha*deltat*visc*p(i)*p(j)*da)
     6                           /(roden*permy)

      aa(j12,j23)=aa(j12,j23) -  alpha*deltat*b(2,i)*p(j)*da

      aa(j13,j21)=aa(j13,j21) +  alpha*deltat*velsound*velsound*
     1                           p(i)*b(1,j)*da

      aa(j13,j22)=aa(j13,j22) +  alpha*deltat*velsound*velsound*
     1                           p(i)*b(2,j)*da

      aa(j13,j23)=aa(j13,j23) +  p(i)*p(j)*da

c     ---------------------------------
c     DISCRETISED FORM OF DARCY EQUATION
c     ---------------------------------
c     Stiffness Matrix of Right Hand Side
c     ===================================
c     For Transient state (Cartesian co-ordinate system)
c     ==================================================

      ak(j11,j21)=ak(j11,j21) +  p(i)*p(j)*da
     1                        -  ((1.0-alpha)*deltat*visc*p(i)*p(j)*da)
     2                           /(roden*permx)

      ak(j11,j22)=ak(j11,j22) +  0.0

      ak(j11,j23)=ak(j11,j23) +  (1.0-alpha)*deltat*b(1,i)*p(j)*da

      ak(j12,j21)=ak(j12,j21) -  0.0

      ak(j12,j22)=ak(j12,j22) +  p(i)*p(j)*da
     1                        -  ((1.0-alpha)*deltat*visc*p(i)*p(j)*da)
     2                           /(roden*permy)

      ak(j12,j23)=ak(j12,j23) +  (1.0-alpha)*deltat*b(2,i)*p(j)*da

      ak(j13,j21)=ak(j13,j21) -  (1.0-alpha)*deltat*velsound*velsound*
     1                           p(i)*b(1,j)*da

      ak(j13,j22)=ak(j13,j22) -  (1.0-alpha)*deltat*velsound*velsound*
     1                           p(i)*b(2,j)*da
```



```
      ak(j13,j23)=ak(j13,j23) +  p(i)*p(j)*da

 6070 continue
 6020 continue

c     For Transient State (Cartesian Co-ordinate System)
c     ==================================================
c     Term one on RHS is evaluated
c     ============================

      do 6080 i=1,ncn

                  j11= i
                  j12= i + ncn
                  j13= i + 2*ncn

      do 6080 j=1,ncn
                  j21= j
                  j22= j + ncn
                  j23= j + 2*ncn

      nn=iabs(node(iel,j))

      akf(j11)=akf(j11) + ak(j11,j21)*vel(nn,1) +
     1                    ak(j11,j22)*vel(nn,2) +
     2                    ak(j11,j23)*press(nn)

      akf(j12)=akf(j12) + ak(j12,j21)*vel(nn,1) +
     1                    ak(j12,j22)*vel(nn,2) +
     2                    ak(j12,j23)*press(nn)

      akf(j13)=akf(j13) + ak(j13,j21)*vel(nn,1) +
     1                    ak(j13,j22)*vel(nn,2) +
     2                    ak(j13,j23)*press(nn)

 6080 continue

c    Evaluation of Elemental Load Vector
c    ===================================

      do 6085 i=1,ncn

                  j11= i
                  j12= i + ncn
                  j13= i + 2*ncn

c     For Transient State  (Cartesian Co-ordinate System)

      rr(j11)=akf(j11)+C(j11)
      rr(j12)=akf(j12)+C(j12)
      rr(j13)=akf(j13)+C(j13)

 6085 continue

      maxte=maxdf

      call front
     1(aa   ,rr   ,iel  ,node ,maxel,maxst,ldest,kdest,nk   ,maxfr
     2,eq   ,lhed ,khed ,kpiv ,lpiv ,jmod ,qq   ,pvkol,vel  ,rl
     3,ncod ,bc   ,nopp ,mdf  ,ndn  ,maxdf,nel  ,maxte,ntov ,lcol
     4,nell ,ntra, press )


c
      return
      end
c
c     ================================================================
      subroutine getnod (nnp  ,cord ,idv11 ,idv2 ,maxnp,ndim,icord)
c
      implicit double precision(a-h,o-z)
c
c     arguments
c     =========
```

```
c     nnp    total number of nodal points in the mesh
c     cord   array for nodal coordinates
c     idv11  input device id. == MESH.FEM file
c     idv2   output device id.
c     ndim   see below
c
      dimension cord(maxnp, ndim)
      rewind (3)
c
      if (.NOT. EOF(3)) read (3,1010) (jnp ,(cord(jnp,idf),idf=1,2)
     1    ,jnp=1,nnp)
      if(icord.eq.0) write(idv2,3010)
      if(icord.eq.1) write(idv2,3020)
      write(idv2,3030) (jnp ,(cord(jnp,idf),idf=1,2) ,jnp=1,nnp)
c
      return
c
1010  format(i5,g20.8,g20.8)
3010  format(' ',///' ',20('*'),' nodal coordinates ',20('*'),//
     1' ',(7x,'id.',13x,'x-coord',13x,'y-coord',20x)/)
3020  format(' ',///' ',20('*'),' nodal coordinates ',20('*'),//
     1' ',2(7x,'id/',7x,'r-coord',7x,'z-coord',20x)/)
3030  format(' ',i10,10x,f10.6,10x,f10.6)
c
      end
c
c     ================================================================
      subroutine getelm (nel  ,ncn  ,node ,idv1 ,idv2 ,maxel)
c
      implicit double precision(a-h,o-z)
c
c     arguments
c     =========
c     ncn    number of nodes per element
c     node   array for element connectivity data
c     idv1   input device id.
c     idv2   output device id.
c     maxel  see below
c
      dimension node (maxel, ncn)
c
              do 6010 iel = 1 ,nel
6010  if (.not. eof(51))read (idv1,1010) iel ,(node(iel,icn),icn=1,ncn)
c     print*, "nodal connectivity array read"
      write(idv2,3010)
               do 6020 jel = 1 ,nel
6020  write(idv2,3020) jel ,(node(jel,icn),icn=1,ncn)
c
      return
c
1010  format(10i8)
3010  format(' ',///,' ',20('*'),' element connectivity ',20('*'),//
     1' ','id.',5x,'n o d a l - p o i n t  e n t r i e s',/)
3020  format(i5,5x,i5,5x,i5,5x,i5,5x,i5,5x,i5,5x,i5,5x,i5,5x,i5,5x,i5,5x,
     1i5)
c
      end
c
c     ================================================================c
      subroutine getbcd (nbc  ,ibc  ,jbc  ,vbc
     1                  ,idv1 ,idv2 ,maxbc)
c
      implicit double precision(a-h,o-z)
c
c     arguments
c     =========
c     nbc    number of nodal constraint data
c     ibc    array for constrained nodal points
c     jbc    array for constrained degree of freedom
c     vbc    array for boundary values
c     idv1   input device id.
c     idv2   output device id.
c     maxbc  see below
c
      dimension ibc  (maxbc) ,jbc  (maxbc),vbc  (maxbc)
c
      if (.not. eof(51))read (idv1,1010) (ibc(ind) ,jbc(ind) ,vbc(ind)
     1    ,ind=1,nbc)
      print*, "boundary conditions array read"
      write(idv2,3010)
      write(idv2,3020) (ibc(ind) ,jbc(ind) ,vbc(ind) ,ind=1,nbc)
c
```



```
      return
c
1010  format(2i5,g20.8)
3010  format(' ',//  /,' ',20('*'),' nodal constraint ',20('*'),//
     1' ',(8x,'id.',7x,'dof',10x,'value',10x)/)
3020  format(5x,i5,5x,i5,f17.4)
c
      end
c
c     ================================================================c
      subroutine putbcv
     1 (nnp  ,nbc ,ibc  ,jbc ,vbc ,ncod ,bc ,maxbc,maxdf, maxel,
     2  node)
c
      implicit double precision(a-h,o-z)
c
c     arguments
c     =========
c     ncod    array for constraint switch defined for every d.o.f.
c     bc      array for storing contraint value
c     maxbc   see below
c     maxdf   see below
c
      dimension ibc  (maxbc) ,jbc  (maxbc) ,vbc  (maxbc)
      dimension ncod (maxdf) ,bc   (maxdf) ,node (maxel,27)
c
      do 6010 ind = 1 ,nbc
      if(jbc(ind)>3) goto 6010
      jnd          = ibc(ind)+(jbc(ind)-1)*nnp
      bc    (jnd) = vbc(ind)
      ncod  (jnd) =  1
6010  continue

c     specifying the stress free condition on node number 84
c     ======================================================
c     iel=16
c     inp=24
c     kc=iabs(node(iel,inp))
c
      return
      end
c
c    ================================================================
      subroutine clean
     1    (ncn  ,nel  ,ndf  ,node ,r1   ,maxel,maxst,maxdf,
     2     bc   ,ncod ,icho )
c
      implicit double precision(a-h,o-z)
c
c     arguments
c     =========
c     all arguments are defined elsewhere.
c
      dimension r1   (maxdf) ,node(maxel,maxst)
      dimension bc   (maxdf) ,ncod(maxdf      )
c
c     function
c     ========
c     cleans the used arrays and makes them ready for  solution
c
           do  6010 i   = 1,maxdf
                 r1(i) = 0.0
                 bc(i) = 0.0
               ncod(i) = 0
6010                 continue
               ntrix   = ndf *ncn
           do  6020 iel = 1,nel
           do  6020 inp = 1,ntrix
           node(iel,inp) = iabs(node(iel,inp))
6020                 continue
           if(icho.ne.1)then
           do 6030 iel = 1,nel
           write(11,3010)iel,(node(iel,j),j=1,ncn)
3010       format(10i5)
6030       continue
c
           endif
c
      return
      end
c
c    ================================================================
```

```
      subroutine setprm
     1     (nnp  ,nel  ,ncn  ,node ,ndf  ,maxel,maxst,ndn  ,ntrix,
     2      maxdf,ntov ,mdf  ,nopp )
c
      implicit double precision(a-h,o-z)
c
c     arguments
c     =========
c     all arguments are defined elsewhere.
c
      dimension node (maxel,maxst), ndn  (maxdf)
      dimension mdf  (maxdf      ), nopp (maxdf)
c
c     function
c     ========
c     Sets the location data for nodal degrees of freedom
c
      do 6010 iel = 1 ,nel
         ndn(iel) = ntrix
      do 6010 icn = 1 ,ncn
              kcn =node(iel,icn)
              jcn =icn+(ndf-2)*ncn
              lcn =kcn+(ndf-2)*nnp

              acn =icn+(ndf-1)*ncn
              bcn =kcn+(ndf-1)*nnp

      node(iel,jcn) = lcn
      node(iel,acn) = bcn
c
6010  continue
      do 6020 idf = 1,ntov
         mdf(idf) = 1
        nopp(idf) = idf
6020  continue
c
      return
      end
c
c     ================================================================
      subroutine getmat (nel ,nmat,pmat, idv1, idv2,maxel,rtem, rpef)
c
      implicit double precision(a-h,o-z)
c
c     arguments
c     =========
c     nmat  number of materials
c     pmat  array for material constants for each element
c     idv1 input device id.
c     idv2  output devide id.
c     maxel see below
c
      dimension pmat (maxel,   9)
c
c     write(idv2,3010)
c
      do 6010 imat = 1 ,nmat
      if (.NOT. EOF(51)) read(idv1,1010) rvisc, power, tref, tbco, taco,
     1                                   dispc, pref, roden, gamad
c     print*, "material properties read"
              ifrom = 1
              ito   = nel

              if(rtem .eq.0.) rtem  = 0.001
              if(rpef .eq.0.) rpef  = 0.001

      do 6020 iel  = ifrom ,ito
         pmat(iel,1) = rvisc
         pmat(iel,2) = pref
         pmat(iel,3) = power
         pmat(iel,4) = tref
         pmat(iel,5) = tbco
         pmat(iel,6) = taco
         pmat(iel,7) = dispc
         pmat(iel,8) = roden
         pmat(iel,9) = gamad
         rtem        = tref
         rpef        = pref

c     roden    density
c     rvisc    mu nought; consistency coefficient
c     pref     reference pressure
```



```
c     power    power law index
c     tref     reference temperature
c     tbco     coefficient b in the power law model
c     taco     coefficient a in the power law model
c     dispc    dispersion coefficient
c     gamad    shear rate
c
6020  continue
c
c      write(idv2,3020) imat ,ifrom ,ito ,rvisc ,power
c      write(idv2,3030)
c      write(idv2,3040) tref ,tbco, pref, taco
c      write(idv2,3050)
c      write(idv2,3060) dispc , roden , gamad

6010  continue
c
      return
c
1010  format(9d10.5)
c
c3010  format(' ',//' ',35('*'),' material properties ',35('*'),//
c     1      ' ',2x,'id.',5x,'eid.(from-to)',3x,'consistency co-efficient'
c     2,5x,'power law index',/)
3020  format(' ',i3,i12,i4,5x,g15.5,15x,g15.5)
c3030  format(/x,' reference temperature   coefficient b
c     1   reference pressure  coefficient a '/)
3040  format(f16.3,f22.4,6x,g10.3,9x,g10.3)
c3050  format(/x,
c     1'Dispersion Coefficient     Density          Shear rate'/)
3060  format(g13.3,15x,g7.1,6x,g16.5)
c
      end
c
c     ================================================================
c
      subroutine contol
     1(vel  ,iter ,ntov ,nnp  ,maxnp,maxdf,errov,errop,vet,
     2 pet, press)

c
      implicit double precision(a-h,o-z)
c
      dimension vel  (maxdf), press(maxnp)
      dimension vet  (maxdf), pet  (maxnp)

      errv  = 0.0
      torv  = 0.0
      errc  = 0.0
      torc  = 0.0
      errp  = 0.0
      torp  = 0.0
c
c     calculate difference between velocities in consecutive iterations
c     =================================================================
c
                  do 6010 icheck = 1,ntov
      if(iter.eq.1) vet(icheck) = 0.0
                  errv = errv +
     1          (vel(icheck)-vet(icheck)) * (vel(icheck)-vet(icheck))
                  torv = torv + vel(icheck)*vel(icheck)
c
      vet(icheck) = vel(icheck)
c
6010              continue
                  errov= errv/torv
c
c     calculate difference between pressures in consecutive  iterations
c     =================================================================

                  do 6030 icheck = 1,nnp
                       if(iter.eq.1)  press(icheck) = 0.0
                       errp = errp +
     1                 (press(icheck)-pet(icheck))*(press(icheck)-pet
     2                 (icheck))
                       torp = torp + press(icheck)*press(icheck)
c
      pet(icheck) = press(icheck)
c
6030              continue
                  errop= errp/torp
```

```
c
      return
      end

c     ======================================================================
      subroutine output
     1     (nnp  ,vel  ,press, maxdf,maxnp,icord,pmat, maxel,actpress)
c
      implicit double precision(a-h,o-z)
c
c     arguments are already defined
c     =============================
      dimension vel (maxdf    ), press(maxnp)
      dimension pmat(maxel,  9), actpress(maxnp)

      write(60,3010)

      if(icord.eq.0) write(60,3020)
      if(icord.eq.1) write(60,3030)


      roden=pmat(1,8)

      do 6010 inp = 1,nnp
                     jnp = inp + nnp
      actpress(inp)=roden*press(inp)

      write(60,3040)inp ,vel(inp),vel(jnp), actpress(inp)

6010  continue

      call minimax
     1( cmax, pmax , vel, press ,  maxnp,  nnp,  nc,   np  ,
     2  nm  ,ncm   , nvxm , nvym,  nvxl  , nvyl  , pmin, cmin ,
     3  vxmax, vxmin, vymax, vymin,  ndim, maxdf, actpress )

      write(60,3045)
      write(60,3050)nvxm,vel(nvxm),nvxl,vel(nvxl)

      write(60,3055)
      write(60,3060)nvym,vel(nnp+nvym),nvyl,vel(nnp+nvyl)

      write(60,3065)
      write(60,3070)np,press(np),nm,press(nm)


3010  format(/' nodal velocities and  pressures  '/)
3020  format('  id.       ux          uy              press'/)
3030  format('  id.       ur          uz              press'/)
3040  format(i5,2e13.4,e22.8)

3045  format('node no.          max ux     node no.          min ux'/)
3050  format(i5,e22.8,i5,e22.8)

3055  format('node no.          max uy     node no.          min uy'/)
3060  format(i5,e22.8,i5,e22.8)

3065  format('node no.          max p      node no.          min p'/)
3070  format(i5,e22.8,i5,e22.8)

      return
      end

c     ======================================================================
      subroutine visca
     1          (rvisc,power,visc,stemp,rtem,tbco,spress,rpef,taco
     2          ,gamad )

      implicit double precision(a-h,o-z)


      visc   = rvisc*((4*gamad)**((power-1.0)/2))


      return
      end
c
c     ======================================================================
      subroutine minimax
     1 ( cmax, pmax , vel, press , maxnp, nnp  , nc  , np ,
```



```
     1   nm  ,ncm   , nvxm   , nvym    , nvxl  , nvyl  , pmin, cmin ,
     1   vxmax, vxmin, vymax, vymin,  ndim, maxdf, actpress )

      implicit real*8 (a-h ,o-z)
      dimension vel  (maxdf)
      dimension press ( maxnp  )
      dimension actpress (maxnp)
c
      vxmax   =   vel(1)
      vxmin   =   vel(1)
      vymax   =   vel(nnp+1)
      vymin   =   vel(nnp+1)

      pmax    =   actpress (1)
      pmin    =   actpress (1)


      np      =   1
      nm      =   1
      nvxm    =   1
      nvym    =   1
      nvxl    =   1
      nvyl    =   1

      do 6020 i=2,nnp

      pm      =   actpress (i)
      pi      =   actpress (i)
      vxmx    =   vel(i)
      vxmn    =   vel(i)
      vymx    =   vel(nnp+i)
      vymn    =   vel(nnp+i)

        if (   pm.gt.pmax  ) then
          pmax=pm
          np  =i
        endif
        if (   pi.lt.pmin  ) then
          pmin = pi
          nm   = i
        endif
        if ( vxmx.gt.vxmax ) then
          vxmax= vxmx
          nvxm  = i
        endif
        if ( vymx.gt.vymax ) then
          vymax= vymx
          nvym  = i
        endif
        if ( vxmn.lt.vxmin ) then
          vxmin= vxmn
          nvxl  = i
        endif
        if ( vymn.lt.vymin ) then
          vymin= vymn
          nvyl  = i
        endif

6020  continue

      return
      end


c     ======================================================================
      subroutine secinv
     1   (nel  ,nnp  ,ncn  ,ngaus,node ,sinv ,cord ,p  ,b ,
     2    del  ,da   ,vel  ,maxnp,maxel,maxst,ndim ,icord,
     3    maxdf,num)
c
      implicit double precision(a-h,o-z)
c
c     function
c     --------
c     calculates the second invariant of rate of deformation
c     tensor at integration points.
c
      dimension vel  (maxdf      ) ,cord (maxnp, ndim)
      dimension node (maxel,maxst) ,sinv (maxel,  num)
      dimension p    (    9      ) ,del  (    2,    9)
      dimension b    (    2,    9)
c
```

```
                                        rewind 15

       do 5000 iel= 1 , nel
               lg = 0
       do 5010 ig = 1 ,ngaus
       do 5010 jg = 1 ,ngaus
               lg = lg+1
c
            if(.not. EOF(15)) read (15) iiel,iig,jjg,p,del,b,da
c
                    x1  = 0.0
                    u1  = 0.0
                    u11 = 0.0
                    u12 = 0.0
                    u21 = 0.0
                    u22 = 0.0
          do 5020 icn = 1 ,ncn
                                    jcn = iabs(node(iel,icn))
                                    mcn = jcn + nnp

c *** components of the rate of deformation tensor
                   u11 = u11 + b(1,icn)*vel(jcn)
                   u12 = u12 + b(2,icn)*vel(jcn)
                   u21 = u21 + b(1,icn)*vel(mcn)
                   u22 = u22 + b(2,icn)*vel(mcn)
 5020     continue
c
c *** second invariant of the rate of deformation tensor
c
      sinv(iel,lg)=0.125*((u11+u11)*(u11+u11)+
     1                    (u12+u21)*(u12+u21)+
     2                    (u21+u12)*(u21+u12)+
     3                    (u22+u22)*(u22+u22))

 5010 continue
 5000 continue

      return
      end
c
c     ======================================================================
      subroutine hydrofilter
     1(nter,ndata,deltat,icho,nnp,ncn,nel,maxel,maxst,maxdf,bc,ncod
     2,r1,node,ndn,ntrix,mdf,nopp,nbc,ibc,jbc,vbc,maxbc,cord,step,
     3pmat,vel,press,temp,ldest,kdest,nk,eq,lhed,khed,kpiv,ntep,
     4lpiv,jmod,qq,pvkol,ngaus,p,del,b,maxfr,ndim,aa,xg,da,num,
     5icord,rr,alpha,sinv,maxnp,actpress,errov,errop,vet,pet,
     6comfactor,percomp,pressexp,permstep1,permstep2, velcartridge,
     7areabase,density,excel)
c
      implicit double precision(a-h,o-z)
c
c     Function
c     --------
c     Calculates the Velocities and Pressures for 2-D Filter Analysis
c
      dimension vel  (maxdf      ) ,cord (maxnp, ndim)
      dimension node (maxel,maxst) ,sinv (maxel,  num)
      dimension p    (    9      ) ,del  (    2,    9)
      dimension b    (    2,    9)
      dimension velcartridge(20  ) ,dpguess(20       )
      dimension press(maxnp      ) ,actpress (maxnp  )
      dimension press_withcomp     (           ndata)
      dimension press_areacomp     (           ndata)
      dimension areabase(ndata)
      dimension excel (6)


c     ================================================================================
c                                 Transient data
c     ================================================================================
c
c     stime       starting time
c
c     deltat      time increment
c
c     nter        maximum number of time steps being employed for finding solution
c     ================================================================================


      if(velcartridge(ndata) .le. 0.0067) then
             dpguess(ndata) = 17014165.2425*velcartridge(ndata)
```



```
      elseif(velcartridge(ndata) .gt. 0.0067  .and.
     1    velcartridge(ndata) .le. 0.0362) then
          dpguess(ndata) = 75818119.5771*(velcartridge(ndata)**1.3099)

      elseif(velcartridge(ndata) .gt. 0.0362) then
             dpguess(ndata) = 30053002.836*velcartridge(ndata)

      end if

      guessp = dpguess(ndata)

      iit = 1

      print *, "Data set = ",ndata
c     print *, "velcartridge = ",velcartridge(ndata)
c     print *, "guessp = ", guessp
c     print *, "inner iteration = ", iit


 9001 do 5150 iter = 1 ,nter


c     TEMPORAL UPWIND SCHEME USED
c     ===========================
      alpha    = 0.75
      deltat   = 0.10
c     ---------------------------

      time = iter*deltat
c     write(60,4090) iter

c     Print *, "Iter = ", Iter
c     ================================================================================
c                          Calculate Nodal Velocities & Pressures
c     ================================================================================
      icho=1
                                              rewind 14
                                              rewind 15

                           ndf   = 3
                           ntov  = ndf * nnp
                           ntrix = ndf * ncn
      call clean
     1    (ncn  ,nel  ,ndf  ,node ,r1   ,maxel,maxst,maxdf,
     2     bc   ,ncod ,icho )

      call setprm
     1    (nnp  ,nel  ,ncn  ,node ,ndf  ,maxel,maxst,ndn  ,ntrix,
     2     maxdf,ntov ,mdf  ,nopp )

      call putbcv
     1  (nnp ,nbc ,ibc ,jbc ,vbc ,ncod ,bc,maxbc,maxdf,maxel, node)


c     idv4 is the file specifier for unit=20
c     ======================================

      idv4=20

      do 5160 iel=1,nel

      call flow (iit, node ,cord ,pmat ,nopp ,mdf ,ndn ,ncod ,bc ,vel
     1,press, r1, temp,ldest,kdest,nk ,eq ,lhed ,khed ,kpiv ,lpiv,
     2jmod, qq, pvkol,iter ,nel ,ncn , ngaus, p,
     3del, b, ntrix, maxel, maxnp, maxst, maxfr, maxdf, ndim ,
     4aa ,xg ,da ,ntov ,num, icord, rr, iel, deltat, alpha,
     5idv4,sinv,nnp,step, permx, permy,guessp,comfactor,percomp,
     6permstep1,permstep2)

 5160 end do

c     iiter=(iter/ntep)*ntep

c     if(iter.eq.1.or.iiter.eq.iter) then

c     call output
c    1   (nnp,vel,press,maxdf,maxnp,icord,pmat,maxel,actpress)
c     endif


 5150 continue
```

```
      do 5170 inp = 1, nnp

          actpress(inp) = press(inp)*density

 5170 end do

 c     if(iit==1) then
 c         press_nocomp = actpress(1741)
 c     end if

 c     press_withcomp(ndata) = actpress(6781)
 c     press_areacomp(ndata) = guessp

       call arealoss (actpress,guessp,areafactor,peraloss,maxnp)

       postarea = peraloss

 c     print *, "postarea = ", postarea

       errarea1 = 0.99*areabase(ndata)
       errarea2 = 1.01*areabase(ndata)

 c     print *, "errarea1 = ", errarea1
 c     print *, "errarea2 = ", errarea2

       if(postarea .le. errarea1) then
           guessp = guessp*1.2
           dpguess(ndata) = guessp
           iit = iit + 1
       goto 9001
       elseif(postarea .ge. errarea2) then
           guessp = guessp*0.8
           dpguess(ndata) = guessp
           iit = iit + 1
       goto 9001
       end if

 c     ====================================================================
 c     If the solutions converge and the simulated peraloss lies within the
 c     range of error bounds assumed for the guessed value of loss in area
 c     factor print the parameters in excel file
 c     ====================================================================
 c
 c     Store the value of pressure drop with compression and no loss in area (BAR)
 c     ---------------------------------------------------------------------------
       excel (2) = (press_withcomp(ndata)/99996.71053)

 c     Store the value of pressure drop with compression and with loss in area (BAR)
 c     ------------------------------------------------------------------------------
       excel (3) = (guessp/99996.71053)

 c     Store the value of percentage compression
 c     -----------------------------------------
       excel (4) =  percomp

 c     Store the value of apparent loss in filtration area
 c     ---------------------------------------------------
       excel (5) = peraloss
       lossinarea= (excel (5))*100

       print *, "pressure drop with compression and no loss in area =
      1", excel(2)
       print *, "pressure drop with compression and with loss in area =
      1", excel(3)
       print *, "% Compression = ", excel(4)
       print *, "% Apparent loss in filtering arean = ", lossinarea
 c
 c     Print the converged velocity and pressure field in output file
 c     --------------------------------------------------------------

       iiter=(iter/ntep)*ntep

       if(iter.eq.1.or.iiter.eq.iter) then

       call output
      1   (nnp,vel,press,maxdf,maxnp,icord,pmat,maxel,actpress)
       endif

 c     Updating viscosity
 c     ------------------
```



```
       call secinv
      1   (nel  ,nnp  ,ncn  ,ngaus,node ,sinv ,cord ,p  ,b,
      2    del  ,da   ,vel  ,maxnp,maxel,maxst,ndim ,icord,
      3    maxdf,num)

 c     Print the output
 c     ================
 c     ====================================================================
 c                            Convergence check
 c     ====================================================================
       call contol(vel ,iter ,ntov ,nnp,maxnp,maxdf,errov, errop
      1,vet ,pet, press)

 c     ====================================================================
 c                             End of time loop
 c     ====================================================================

 4090  format(///'TIME STEP  no.',i5,//)

       return
       end

 c     ====================================================================
       subroutine perm
      1(step,permx,permy,deltap,comfactor,percomp,permstep1,permstep2)

 c
       implicit double precision(a-h,o-z)
 c
 c     Function
 c     --------
 c     Calculates the Permeability of the Fiber Glass Media for Filter Analysis
 c
 c     Convert experimental pressure drop in bar to pascals
 c     ----------------------------------------------------


       if (deltap .le. 1.13E+5) then
       permx       =   1.96735E-12
       permy       =   1.96735E-12
       permstep1   =   1.96735E-12
       permstep2   =   permx
       end if

       if (deltap .gt. 1.13E+5 .and. deltap .le. 10.8E+5) then
       permx       =   (1.628*0.00000000002)*((deltap)**(-0.2397))
       permy       =   (1.628*0.00000000002)*((deltap)**(-0.2397))
       permstep1   =   1.96735E-12
       permstep2   =   permx
       end if

       if (deltap .gt. 10.8E+5) then
       permx       =   1.0435900E-12
       permy       =   1.0435900E-12
       permstep1   =   1.96735E-12
       permstep2   =   permx
       end if

 c      end if

 c     Calculate the compression factor and percentage compression
 c     -----------------------------------------------------------

       comfactor     =    ((permstep1-permstep2)/permstep1)
       percomp       =    (comfactor*100)

       return
       end

 c     ====================================================================
       subroutine arealoss
      1          (actpress,guessp,areafactor,peraloss,maxnp)
 c
       implicit double precision(a-h,o-z)

       dimension actpress(maxnp )

 c
 c     Function
 c     --------
 c     Calculates the Loss in Area Factor and Percentage Loss in Area
 c
```

```
c     Convert experimental pressure drop in bar to pascals
c     ----------------------------------------------------
      deltap      = guessp
      peraloss    = ((deltap-actpress(6781))/deltap)
      areafactor  = 1.0-peraloss

      print*, "Guess value of Pressure in pa  = ",deltap
      print*, "Simulated Pressure drop in pa = ",actpress(6781)
      print*, "Percentage Loss in Area =", peraloss
      print*, "Area factor for Velocity Correction =",areafactor
      return
      end

c     ================================================================
      SUBROUTINE GFMFEM

      IMPLICIT REAL*8 (A-H,O-Z)

      CHARACTER CH(400)*1,SF*4,CC*3,FNAME*30

      DIMENSION NOD(9)

C     FUNCTION
C     --------
C     ---------------------------------------------------
C     TRANSFORMATION OF GEOSTAR GFORM FILE TO MESH.FEM FILE
C     ---------------------------------------------------

      WRITE (*,130 )
130   FORMAT(1X,'Enter GFM file name for the Filter Mesh    ')
      READ (*,135) FNAME
135   FORMAT (A30)
      OPEN (UNIT=1,FILE=FNAME,FORM='FORMATTED')
      OPEN (UNIT=2,STATUS='SCRATCH',FORM='FORMATTED')
      OPEN (UNIT=3,FILE='MESH.FEM',FORM='FORMATTED')
      NBC=0
      IOS=0
      NNM=0
      NEM=0
      DO WHILE ( IOS.EQ.0 )
         READ (1,100,ERR=300,END=300,IOSTAT=IOS) (CH(J),J=1,400)
         CC=CH(1)//CH(2)
         IF (CC.EQ.'ND')THEN
            WRITE (2,100) (CH(K),K=4,300)
            NNM=NNM+1
         ENDIF
         IF (CC.EQ.'EL')THEN
            WRITE (2,100) (CH(K),K=4,300)
            NEM=NEM+1
         ENDIF
100      FORMAT (500A)
      ENDDO
300   REWIND 2
      PRINT *,' '
c     PRINT *,'TOTAL NO. OF NODES    =  ',NNM
c     PRINT *,'TOTAL NO. OF ELEMENTS =  ',NEM
      DO I=1,NNM
         READ (2,*) N,X,Y
         X1 = X*1e-3
         Y1 = Y*1e-3
         WRITE (3,110 ) N,X1,Y1
110      FORMAT (I5,2G20.8)
      ENDDO

c     PRINT *,'File MESH.FEM is created '
      RETURN
      END

c     ======================================================
      SUBROUTINE CARTRIDGE (ntd,density,fr,velcartridge,farea,viiisc,
     1                      areabase,xab,yab,xjkl,yjkl,xfgfd,yfgfd,
     2                      xldkdjd,yldkdjd,xadbd,yadbd)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION fr(20), velcartridge(20), pd(20), viiisc(20),
     1          areabase(20)

c     FUNCTION
c     --------
c     ------------------------------------------------------------------------------------------
c
```



```
c     READS THE FILTERING AREA, FLOW RATE, VISCOSITIES, FLOW RATES & BC CO-ORDINATES FOR FILTER
 MESH
c     ------------------------------------------------------------------------------------------
c
-----

c     Opening of file to be cartridge filter element
c     ==============================================
c
      print*,'Enter the Reference ID of the FILTER ELEMENT'
      read(*,100) cartid

      open(unit=511,File=cartid,access='sequential',form='formatted',
     1     status="unknown",iostat=ios)

c     Read the total number of sets of data
c     -------------------------------------
      read(511,2001) ntd

c     Read the total filtering area and density of the fluid
c     ------------------------------------------------------
      read(511,2005) farea, density

c     Read the Flow Rate in (lit/min), Pressure Drop in (bar) and Viscosity in (Pa.s)
c     -------------------------------------------------------------------------------
      do i=1,ntd
      jj=i
      read(511,2002) fr(jj),areabase(jj),viiisc(jj)

      end do

c     Calculate velocity (m/s) corresponding to experimental flow rate

      do i=1,ntd
      jj=i
      velcartridge(jj) = fr(jj)*0.001/(farea*60)
      end do

c     Read the co-ordinates in (mm) to be used for the calculation of boundary conditions
c     -----------------------------------------------------------------------------------
      read(511,2003) xab,yab
      read(511,2003) xjkl,yjkl
      read(511,2003) xfgfd,yfgfd
      read(511,2003) xldkdjd,yldkdjd
      read(511,2003) xadbd,yadbd

      xab         = xab*1e-3
      yab         = yab*1e-3
      xjkl        = xjkl*1e-3
      yjkl        = yjkl*1e-3
      xfgfd       = xfgfd*1e-3
      yfgfd       = yfgfd*1e-3
      xldkdjd     = xldkdjd*1e-3
      yldkdjd     = yldkdjd*1e-3
      xadbd       = xadbd*1e-3
      yadbd       = yadbd*1e-3

c     print*, xab,yab
c     print*, xjkl,yjkl
c     print*, xfgfd,yfgfd
c     print*, xldkdjd,yldkdjd
c     print*, xadbd,yadbd


c     Read the Geometrical Parameters of the Filter Element
c     -----------------------------------------------------
      read(511,2004) enp    ! Number of Pleats
c     print*, enp

      read(511,2004) hp     ! Height of Filter Element (mm)
c     print*, hp

      read(511,2004) el     ! Length of Pleat          (mm)
c     print*, el

      read(511,2004) di     ! Diameter of the Core     (mm)
c     print*, di

c     Writing all the cartridge description in file 61 used for excel
c     ---------------------------------------------------------------
      write (61,4020) farea,enp,hp,el,di,density

  100 format(80a)
```

```
 2001 format(i5)
 2002 format(10x,d10.5,10x,d10.5,10x,d10.5)
 2003 format(5x,f22.12,f22.12)
 2004 format(10x,f10.5)
 2005 format(10x,d10.5,10x,d10.5)

 4020 format(' ',20x,'-------------------------------------',/
     1' ',20x,' CARTRIDGE FILTER DESCRIPTION DATA',/
     2' ',20x,'-------------------------------------',////
     325x,'Filtering area of cartridge    (m^2)       =',f10.4,/
     225x,'Number of pleats               =',f10.4,/
     325x,'Height of the Filter Element   (mm)        =',f10.4,/
     425x,'Length of the pleat        (mm)        =',f10.4,/
     525x,'Diameter of the cartridge  (mm)        =',f10.4,/
     725x,'Hydraulic fluid density    (kg/m^3)    =',f10.4,/
     7//)

      write (61,4030)


 4030 format(/'     Flow Rate(lit/min)        Viscosity
     1    (Pa.s)      Pressure com(Bar)       Pressure com + area(Bar)
     3    % Compression          %Loss in area   '/)

      RETURN
      END

c     ================================================================
      subroutine pleatbc
     1      (frate,farea,nbc,ibc,jbc,vbc,maxbc,maxnp,ndim,cord,step,
     2       areafactor,xab,yab,xjkl,yjkl,xfgfd,yfgfd,xldkdjd,yldkdjd,
     3       xadbd,yadbd)

      implicit real*8 (a-h,o-z)

      dimension x    (10101)
      dimension y    (10101)
      dimension vx   (10101)
      dimension vy   (10101)
      dimension ibc  (maxbc) ,jbc  (maxbc),vbc  (maxbc)
      dimension cord (maxnp  ,ndim)

      parameter  (pi=3.14159265358979323 8D0)

c     --------
c     FUNCTION
c     --------
C     ---------------------------------------------------------------------
C     CALCULATION OF BOUNDARY CONDITIONS FOR THE POROUS FLOW SIMULATION
C     ---------------------------------------------------------------------
      open (unit=70,file='velbcstep1', form='formatted', status= 'new')
      rewind (70)

c
c     Conversion of volumetric flow rate in litres/minute to average velocity in metres/second
c     ============================================================================================
      frate = (frate)/(farea*60000.0)

c     Read the co-ordinates and store these values in arrays
c     ======================================================

      do i=1,10101
      x(i)=cord(i,1)
      y(i)=cord(i,2)
      end do

c     ------------------------------------------------
c     CALCULATIONS OF THE BOUNDARY CONDITIONS STARTS
c     ------------------------------------------------
c     These are the calculations for the curved part ab
c     ================================================

      do i=1,40

      slope=(y(i+1)-yab)/(x(i+1)-xab)
      theta=atan(slope)
      theta=-theta
      vx(i+1)= frate*cos(theta)
      vy(i+1)=-frate*sin(theta)

      end do
```



```
c     These are the calculations for the straight part bc
c     ===================================================

      slope =(y(901)-y(862))/(x(901)-x(862))
      slopen=(-1.0/slope)
      theta=atan(slopen)
      theta=-theta
      do i=1,40
      vx(i+861)= frate*cos(theta)
      vy(i+861)=-frate*sin(theta)
      end do


c     These are the calculations for the curved part cd
c     =================================================

      do i=1,40

      slope=(y(i+1701)-yjkl)/(x(i+1701)-xjkl)
      theta=atan(slope)
      theta=-theta
      vx(i+1701)= frate*cos(theta)
      vy(i+1701)=-frate*sin(theta)

      end do

c     These are the calculations for the curved part de
c     =================================================

      do i=1,40

      slope=(y(i+2541)-yjkl)/(x(i+2541)-xjkl)
      theta=atan(slope)
      vx(i+2541)=-frate*cos(theta)
      vy(i+2541)=-frate*sin(theta)

      end do

c     These are the calculations for the straight part ef
c     ===================================================

      slope =(y(3382)-y(3421))/(x(3382)-x(3421))
      slopen=(-1.0/slope)

      do i=1,40
      vx(i+3381)=-frate*cos(theta)
      vy(i+3381)=-frate*sin(theta)
      end do


c     These are the calculations for the curved part fg
c     =================================================

      do i=1,40

      slope=(y(i+4221)-yfgfd)/(x(i+4221)-xfgfd)
      theta=atan(slope)

      vx(i+4221)=-frate*cos(theta)
      vy(i+4221)=-frate*sin(theta)

      end do


c     These are the calculations for the curved part gf'
c     ==================================================

      do i=1,40

      slope=(y(i+5061)-yfgfd)/(x(i+5061)-xfgfd)
      theta=atan(slope)
      theta=-theta
      vx(i+5061)= frate*cos(theta)
      vy(i+5061)=-frate*sin(theta)

      end do

c     These are the calculations for the straight part f'e'
c     =====================================================

      slope =(y(5941)-y(5902))/(x(5941)-x(5902))
      slopen=(-1.0/slope)
```

```
      theta=atan(slopen)
      theta=-theta
      do i=1,40
      vx(i+5901)= frate*cos(theta)
      vy(i+5901)=-frate*sin(theta)
      end do


c     These are the calculations for the curved part e'd'
c     ===================================================

      do i=1,40

      slope=(y(i+6741)-yldkdjd)/(x(i+6741)-xldkdjd)
      theta=atan(slope)
      theta=-theta
      vx(i+6741)= frate*cos(theta)
      vy(i+6741)=-frate*sin(theta)

      end do



c     These are the calculations for the curved part d'c'
c     ===================================================

      do i=1,40

      slope=(y(i+7581)-yldkdjd)/(x(i+7581)-xldkdjd)
      theta=atan(slope)
      vx(i+7581)= -frate*cos(theta)
      vy(i+7581)= -frate*sin(theta)

      end do

c     These are the calculations for the straight part c'b'
c     ===================================================

      slope =(y(8422)-y(8461))/(x(8422)-x(8461))
      slopen=(-1.0/slope)

      do i=1,40
      vx(i+8421)=-frate*cos(theta)
      vy(i+8421)=-frate*sin(theta)
      end do


c     These are the calculations for the curved part b'a'
c     ===================================================


      do i=1,39

      slope=(y(i+9261)-yadbd)/(x(i+9261)-xadbd)
      theta=atan(slope)

      vx(i+9261)=-frate*cos(theta)
      vy(i+9261)=-frate*sin(theta)

      end do

c     ============================================================
c     STEP 1  :   WRITE THE BOUNDARY CONDITIONS IN FILE NUMBER 70
c     ============================================================

c      if (step .eq. 1) then

      ncount=0

c     Writing the boundary conditions in the output file
c     ==================================================

c     vx for curve ab
      do i=1,40
      j=i+1
      ncod =1
      value=vx(i+1)

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
```



```
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     vx for line bc
      do i=1,40
      j=i+861
      ncod =1
      value=vx(861+i)

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve cd
      do i=1,40
      j=i+1701
      ncod =1
      value=vx(1701+i)

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do


c     vx for curve de
      do i=1,40
      j=i+2541
      ncod=1
      value=vx(2541+i)

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     vx for line ef
      do i=1,40
      j=i+3381
      ncod=1
      value=vx(i+3381)

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve fg
      do i=1,40
      j=i+4221
      ncod=1
      value=vx(4221+i)

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve gf'
```

```
      do i=1,40
      j=i+5061
      ncod =1
      value=vx(i+5061)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vx for line f'e'
      do i=1,40
      j=i+5901
      ncod =1
      value=vx(5901+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve e'd'
      do i=1,40
      j=i+6741
      ncod =1
      value=vx(6741+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve d'c'
      do i=1,40
      j=i+7581
      ncod=1
      value=vx(7581+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vx for line c'b'
      do i=1,40
      j=i+8421
      ncod=1
      value=vx(i+8421)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve b'a'
      do i=1,40
      j=i+9261
      ncod=1
      value=vx(9261+i)

      ncount = ncount+1
```



```
      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve ah
      do i=0,20
      j=1+(41*i)
      ncod=1
      value=0.0

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vx for curve a'h'
      do i=0,20
      j=9301 +(40*i)
      ncod=1
      value=0.0

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for curve ab
      do i=1,40
      j=i+1
      ncod =2
      value=vy(i+1)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for line bc
      do i=1,40
      j=i+861
      ncod =2
      value=vy(861+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for curve cd
      do i=1,40
      j=i+1701
      ncod =2
      value=vy(1701+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do
```

```
c     vy for curve de
      do i=1,40
      j=i+2541
      ncod=2
      value=vy(2541+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for line ef
      do i=1,40
      j=i+3381
      ncod=2
      value=vy(i+3381)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for curve fg
      do i=1,40
      j=i+4221
      ncod=2
      value=vy(4221+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for curve gf'
      do i=1,40
      j=i+5061
      ncod =2
      value=vy(i+5061)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for line f'e'
      do i=1,40
      j=i+5901
      ncod =2
      value=vy(5901+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for curve e'd'
      do i=1,40
      j=i+6741
      ncod =2
      value=vy(6741+i)
```



```
      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for curve d'c'
      do i=1,40
      j=i+7581
      ncod=2
      value=vy(7581+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for line c'b'
      do i=1,40
      j=i+8421
      ncod=2
      value=vy(i+8421)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     vy for curve b'a'
      do i=1,39
      j=i+9261
      ncod=2
      value=vy(9261+i)

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for line hi
      do i=1,40
         j=821+ i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for curve ij
      do i=1,40
         j=1661 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)   = j
      jbc(ncount)   = ncod
      vbc(ncount)   = value

      write (70,2000) j, ncod, value
```

```
      end do

c     Pressure for curve jk
      do i=1,40
         j=2501 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for curve kl
      do i=1,40
         j=3341 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for line lm
      do i=1,40
         j=4181 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for curve mn
      do i=1,40
         j=5021 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do


c     Pressure for curve nm'
      do i=1,40
         j=5861 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for  line m'l'
      do i=1,40
         j=6701 + i
      ncod =3
```



```
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for  curve l'k'
      do i=1,40
         j=7541 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for curve k'j'
      do i=1,40
         j=8381 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for line j'i'
      do i=1,40
         j=9221 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Pressure for curve i'h'
      do i=1,39
         j=10061 + i
      ncod =3
      value=0.0

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do


c     Concentration  for curve ab
      do i=1,40
      j=i+1
      ncod =4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
```

```
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration  for line bc
      do i=1,40
      j=i+861
      ncod =4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for curve cd
      do i=1,40
      j=i+1701
      ncod =4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do


c     Concentration for curve de
      do i=1,40
      j=i+2541
      ncod=4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for line ef
      do i=1,40
      j=i+3381
      ncod=4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for curve fg
      do i=1,40
      j=i+4221
      ncod=4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for curve gf'
```



```
      do i=1,40
      j=i+5061
      ncod =4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for line f'e'
      do i=1,40
      j=i+5901
      ncod =4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for curve e'd'
      do i=1,40
      j=i+6741
      ncod =4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for curve d'c'
      do i=1,40
      j=i+7581
      ncod=4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for line c'b'
      do i=1,40
      j=i+8421
      ncod=4
      value=0.2

      ncount = ncount+1

      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do

c     Concentration for curve b'a'
      do i=1,39
      j=i+9261
      ncod=4
      value=0.2

      ncount = ncount+1
```

```
      ibc(ncount)    = j
      jbc(ncount)    = ncod
      vbc(ncount)    = value

      write (70,2000) j, ncod, value
      end do


1000  format (i5,2G20.12)
2000  format (2i5,g20.8)

c      rewind(70)

c     if (.not. eof(70))read (70,2000) (ibc(ind) ,jbc(ind) ,vbc(ind)
c    1    ,ind=1,nbc)
c     print*, "boundary conditions array read"
      write(60,3010)
      write(60,3020) (ibc(ind) ,jbc(ind) ,vbc(ind) ,ind=1,nbc)
c
c      end if



      return
c
1010  format(2i5,g20.8)
3010  format(' ',//  /,' ',20('*'),' nodal constraint ',20('*'),//
     1' ',(8x,'id.',7x,'dof',10x,'value',10x)/)
3020  format(5x,i5,5x,i5,f17.4)
c
      close(70)
      return
      end

c     ================================================================================
      subroutine excelprint(iiii, excel, fr, pd, viiisc)

      implicit real*8 (A-H,O-Z)

      dimension excel (6), fr(20), pd(20), viiisc(20)

c     Function
c     --------
c     Prints the flow rate, viscosity, simulated pressure drop
c     values, compression factors and the percentage loss in filtration area
c

      excel(5) = excel(5)*100

      write(61,100) fr(iiii), viiisc(iiii), excel(2),
     1              excel(3),  excel(4)   , excel(5)

  100 format(6(e15.4, 8x))

      return
      end

c     ================================================================================
c     e    n    d           o    f           s   u   b   r   o   u   t   i   n   e   s
c     ================================================================================
```

```
c   ************************************************************
c
c                  Auxilliary Program for ACFAMP
c
c       Developed by  : Navraj S Hanspal & Atul N Waghode
c
c       Supervised by : Prof V.Nassehi & Prof. R.J.Wakeman
c
c                 Latest Version: 23rd May, 2005
c
c   ************************************************************
c   -------------------
c   Program Description
c   -------------------
c
c   This program generates the "Reference cartridge ID file"
c   containing the cartridge characteristic data to be used
c   in main execution of ACFAMP
c
c   INPUT TO BE SUPPLIED BY THE USER
c   --------------------------------
c   1. Reference ID of the FILTER ELEMENT
c   2. Height of pleat in (mm)
c   3. Height of element in (mm)
c   4. Filtering area in (m^2)
c   5. Diameter of the inner core in (mm)
c   6. Centre coordinates of the repetitive cartridge filter domain
c   7. Density of fluid (kg/m^3)
c   8. Total number of data sets to be used in analysis
c   9. Flow rate (lit/min) corresponding to each data set
c  10. Viscosity (cSt) corresponding to each data set
c  11. Assumed percentage filtration area loss for each data set
c
c  ================================================================

      Program FilterData

      implicit real*8 (a-h,o-z)

      dimension fr(20), pd(20), visc(20)

      print *, "
     1    Program Developers: N.S Hanspal and A.N. Waghode"
      print *, "
     1  Project Advisors: Prof. R.J. Wakeman and Prof. V. Nassehi"
      print *, "
     1          Advanced Separation Technologies Group"
      print *, "
     1        Department of Chemical Engineering"
      print *, "
     1              Loughborough University"
      print *, "
     1             Loughborough, Leicestershire"
      print *, "
     1              LE11 3TU, United Kingdom"


      print *, "========================="
      print *, "========================="
      print *, "ACFAMP AUXILLIARY PROGRAM"
      print *, "========================="
      print *, "========================="

      write(*, 1101)

      write(*, 1111)

      write(*, 1121)

      write(*, 1131)

c
c     Opening of file to be used for the FILTER ANALYSIS code
c     =======================================================
c
      print*,'Enter the Reference ID of the FILTER ELEMENT'
      read(*,100) filnam

      open(unit=51,File=filnam,access='sequential',form='formatted',
     1     status="unknown",iostat=ios)

c     Input the Geometrical Parameters of the Filter Element from the Excel Sheets
```



```
c     ========================================================================

      print*, "Enter the height of pleat in (mm) ="
      read*,   hp

      print*, "Enter the number of pleats ="
      read*,   np

      print*, "Enter the height of element in (mm) ="
      read*,   el

      print*, "Enter the filtering area in (m^2) ="
      read*,   fa

      print*, "Enter the diameter of the inner core in (mm) = "
      read*,   di

c     Input the co-ordinates to be used for evaluation of Boundary Conditions
c     ========================================================================

      print*, "Read the 1st set of co-ordinate : x,y"
      read*,   xab,yab

      print*, "Read the 2nd set of co-ordinate : x,y"
      read*,   xjkl,yjkl

      print*, "Read the 1st set of co-ordinate : x,y"
      read*,   xfgfd,yfgfd

      print*, "Read the 1st set of co-ordinate : x,y"
      read*,   xldkdjd,yldkdjd

      print*, "Read the 1st set of co-ordinate : x,y"
      read*,   xadbd,yadbd

c     Input the Density of Fluid
c     ==========================

      print*, "Enter the density in (kg/m^3) ="
      read*,   density

c     Total number of date sets i.e. Flow Rate, Pressure Drop and Viscosity Data for the Filter
 Element
c     ================================================================================================
========

      print*, "Enter the total number of data sets ="
      read*,   ntd

c     Input the Flow Rate, Pressure Drop and Viscosity Data the Filter Element from the Excel S
heets
c     ================================================================================================
=====

      do i=1,ntd
      jj=i

      print*,"Enter the flow rate in (lit/min), percentage loss in area
     1        and viscosity in (cSt)"

      read*, fr(jj),pd(jj),visc(jj)
      pd(jj)=pd(jj)/100.0
      end do


c     Writing in the data file to be used by Filter Analyis Code
c     ==========================================================

c     Write the total number of sets of data
c     --------------------------------------
      write(51,2001) ntd

c     Write the total filtering area and density
c     ------------------------------------------
      write(51,2005) fa, density

c     Write the Flow Rate in (lit/min), Pressure Drop in (bar) and Viscosity in (Pa.s)
c     --------------------------------------------------------------------------------
      do i=1,ntd
      jj=i
      visc(jj)=visc(jj)*0.000001*density
      write(51,2002) fr(jj),pd(jj),visc(jj)
```

```
      end do

c     Write the co-ordinates in (m) to be used for the calculation of boundary conditions
c     ------------------------------------------------------------------------------------
      write(51,2003) xab,yab
      write(51,2003) xjkl,yjkl
      write(51,2003) xfgfd,yfgfd
      write(51,2003) xldkdjd,yldkdjd
      write(51,2003) xadbd,yadbd

c     Write the Geometrical Parameters of the Filter Element
c     ------------------------------------------------------

      write(51,2004) np    ! Number of Pleats
      write(51,2004) hp    ! Height of Filter Element  (mm)
      write(51,2004) el    ! Length of Pleat           (mm)
      write(51,2004) di    ! Height of Filter Element  (mm)

  100 format(80a)
 2001 format(i5)
 2002 format(10x,d10.5,10x,d10.5,10x,d10.5)
 2003 format(5x,g22.12,g22.12)
 2004 format(10x,f10.5)
 2005 format(10x,d10.5,10x,d10.5)


1101  format(' ',/)
1111  format (/'  This auxilliary program generates the Reference
     1cartridge ID file ',/' to be used in execution of ACFAMP.
     1It will now ask the user to input ',/' relevant data to be
     1used in cartridge filter analysis')
1121  format(' ', 2(/))

1131  format("  ************************************************
     1*****************")


      close (51)
      end
```