Distribution
Category **UC-32**

SAND86-1817
Unlimited Release
Printed September 1987
Revised February 2019

NACHOS II - A FINITE ELEMENT COMPUTER PROGRAM FOR INCOMPRESSIBLE FLOW PROBLEMS

PART II - USER'S MANUAL

D. K. Gartling
Fluid Mechanics and Heat Transfer Division I
Sandia National Laboratories
Albuquerque, New Mexico 87185

ABSTRACT

Complete user instructions are given for the finite element, fluid mechanics and heat transfer code, NACHOS II. The theoretical background and numerical methods used in the program are documented in SAND86-1816. Examples of the use of the code are presented in SAND86-1818.

Preface

Since the first version of NACHOS was released in late 1977, a great deal of research and development has occurred in the application of finite element methods to problems in fluid mechanics. Progress in the development of new elements and formulations for incompressible flows, improved iterative and transient solution algorithms and new matrix solution procedures have combined to make current practice significantly different from the accepted methods of a decade ago. These new methods, in concert with significant changes in computer hardware, have dictated the need for a revised and updated version of the NACHOS program. The present report, along with the revised theoretical report and new example problem report, document the new code, NACHOS II.

In developing NACHOS II a number of new capabilities and features were added to increase the overall utility of the code. The basic system of balance equations (*i.e.*, mass, momentum and energy) has been augmented with the inclusion of two user defined transport equations. This extension allows problems such as double-diffusion, mass transport and some chemically reacting flows to be considered without code modfication. Also, a porous flow model was added to the code that permits fluid saturated porous materials to be included in a simulation. The element library was expanded to include a nine node Lagrange element; a continuous or discontinuous pressure approximation can be used with any element in the library. The discontinuous pressure elements can also be used with a penalty function formulation of the incompressibility constraint. In order to improve the performance of the solution algorithms, all equations in a problem are solved in a fully coupled manner. For steady-state simulations the standard Picard method has been augmented with a full Newton method and a quasi-Newton procedure. Transient analyses are performed with either a backward Euler or trapezoid rule integration procedure. Either method can be run with a fixed time step or a dynamic time step selection procedure.

Minor improvements in the allowed material models, boundary condition types and dependencies, flux computations and the stream function computation have also been incorporated in NACHOS II. The code has been rewritten in standard FORTRAN 77 to increase its portability. A dynamic dimensioning scheme has been devised to optimize machine usage. Finally, new pre- and post-processing file formats have been developed to permit stand-alone mesh generators and graphics programs to be easily integrated with the program.

Unfortunately, the modifications and improvements found in NACHOS II have required some alterations in the input and data syntax for the program. The new version of NACHOS is, therefore, not compatible with the older NACHOS input.

Contents

1 Introduction

The NACHOS II computer code is a general purpose program designed for the solution of two-dimensional, viscous, incompressible fluid dynamics problems. The code is based on the Galerkin form of the finite element method (FEM), a technique that has found increased use and acceptance in fluid mechanics over the last several years. The present version of NACHOS II represents a significantly enhanced edition of the original program which was first released in 1977. However, the original intent of providing both a useful analysis program and an easily modified research program has been maintained in the current code.

The class of problems treated by NACHOS II are those described by the two-dimensional (plane or axisymmetric), incompressible form of the Navier-Stokes equations. An energy transport equation is also included in the formulation for problems in which heat transfer effects are important. Two auxiliary transport equations can be added to the equation system to describe other physical processes, *e.g.* mass transfer, chemical reactions. Among the specific types of flow problems for which the prsesent code is suitable are the following:

a) Isothermal flow problems

b) Forced convection problems

c) Free convection problems

d) Mixed convection problems

e) Conjugate heat transfer problems

f) Flow in saturated porous media with or without heat transfer

g) Inelastic, non-Newtonian flows with or without heat transfer

Other problem classes are also possible depending on the specific definitions applied to the auxiliary transport equations. NACHOS II readily accepts either transient or steady-state forms of the above problems.

A conscious effort has been made during the development of NACHOS II to make the program user-oriented. The basic code structure and many of the programming features found in NACHOS II have been developed and refined from earlier versions of the program. The code has been written using standard FORTRAN 77 in an effort to increase its portability. NACHOS II is one of a series of codes for fluid mechanics and heat transfer problems [1,2], all of which share a common code structure and input format. This feature allows a user ready access to a variety of analysis packages with a minimum of time spent on learning input format conventions.

The present document is intended to provide a detailed description of the input data necessary to access and execute the NACHOS II code. A companion document [3] provides a complete description of the theoretical basis for the code and many of the numerical procedures used in the program. For the serious user of NACHOS II, the theoretical background material is required reading. A detailed description of the use of the code in the solution of various flow problems is provided in [4].

In the next section, a brief description of program capabilities and organization is provided. The following section describes the input data to the program; the penultimate section provides a discussion of programming and installation details. Several short example problems are included in the last section to illustrate typical input data files and use of the mesh generator.

2 Program Overview

The development of a reasonably efficient computer code in an area as extensive as fluid dynamics necessarily demands that the limits of applicability of the program be specifically and carefully defined. The present description is intended only as an overview of the major assumptions, capabilities and features of the program; a more complete outline of code limitations is provided in [3]. A brief section is also included to show the major organizational components of the program.

2.1 Program Features

NACHOS II is restricted to problems for which the fluid of interest may be assumed incompressible within the context of the Boussinesq approximation [3]. Geometrically, the program is limited to treating two-dimensional, plane or axially symmetric problems (boundary shape is arbitrary). These two restrictions on material behavior and geometry serve to define the basic class of problems for which NACHOS II was designed. Within this definition, several other assumptions either explicit or implicit, have been made to further limit the applicability of the code.

The theoretical formulation assumes isotropic behavior for all fluids, solids and porous materials; all fluids are further assumed to obey a Newtonian or generalized Newtonian (inelastic) [3] constitutive law. Flow fields are generally assumed to be laminar, though simple algebraic turbulence models are also allowed. No provision is made for the explicit computation of fluid/fluid interfaces (*i.e.*, free surfaces).

Despite these restrictions, NACHOS II has been successfully used to simulate a wide range of flow problems. Both transient and steady-state analyses of isothermal, forced convection, and free convection problems have been carried out. When considering flows with heat transfer, regions of solid body heat conduction are easily included. Flows in saturated porous layers can also be simulated with both the inclusion or omission of a viscous fluid region. Material properties such as fluid viscosity and thermal conductivity may be arbitrary functions of temperature and other appropriate variables; volumetric source terms may be functions of time, spatial location, temperature or other variables. Hydrodynamic and thermal boundary conditions are quite general and may include specified fluid velocities or tractions, slip or no slip boundaries, specified temperatures or heat fluxes, and convective and radiative boundaries. Boundary conditions may generally be functions of spatial location and time. If needed, NACHOS II allows one or two auxiliary transport equations to be defined and included in the analysis. These equations could be used to describe mass transport in a binary mixture, simple chemical reactions or other such transport processes.

NACHOS II is a self-contained analysis program with its own mesh generator, data analysis and batch plotting package. The mesh generator is based on an isoparametric

mapping procedure and allows quite general regions to be meshed easily and accurately. The data analysis portions of the code allow the stream function, local heat fluxes and fluid stresses to be calculated. The plotting package provides graphic output of element meshes, nodal point locations, contour plots, vector plots, time histories and profiles of any dependent variable. NACHOS II can also be configured to exchange data with stand-alone programs such as external mesh generation codes and post-processing graphics packages.

Two other essential parts of NACHOS II are the element library and the solution procedures for the finite element equations. The elements included in the library consist of isoparametric and subparametric quadrilaterals and triangles, as shown in Figure 1. Within each of these elements, the velocity components and temperature (and auxiliary variables, when present) are approximated using biquadratic basis functions; the pressure is given by a either a continuous, bilinear approximation or a discontinuous, linear approximation. A penalty function approximation to the incompressibility constraint can be employed with the discontinuous pressure elements. For the analysis of steady-state problems, NACHOS II provides the user with a choice of iterative solution procedures – a Picard iteration scheme or a full Newton procedure. Transient problems are analyzed using either of two implicit integration schemes – a first-order Euler method or a second-order trapezoid rule. Both integrators are used in a predictor/corrector mode with a fixed timestep or a dynamic timestep algorithm. Details of these methods and their implementation can be found in [3].

2.2 Program Organization

NACHOS II has been organized in a modular form in an effort to make the operation of the code understandable and to ease the task of code modification. As shown in the schematic of Figure 2, the code consists of a main program plus a number of major, task oriented subroutines. Each of the major subroutines may access one or more subordinate routines plus the utilities attached to the main program.

Input to the program consists of a series of commands and associated data statements. Each command directs the main program to call one of the major subroutines and execute a particular task in the finite element procedure. A list of the major subroutines, their associated input commands, and a brief description of their function is given below.

- MATRL (**MATERIALS**) - reads and stores material property data
- MESH (**MESH**) - reads data and creates nodal point distributions
- ELEMNT (**ELEMENTS**) - reads and stores element and boundary condition data
- FORMKF (**FORMKF**) - generates element level matrices

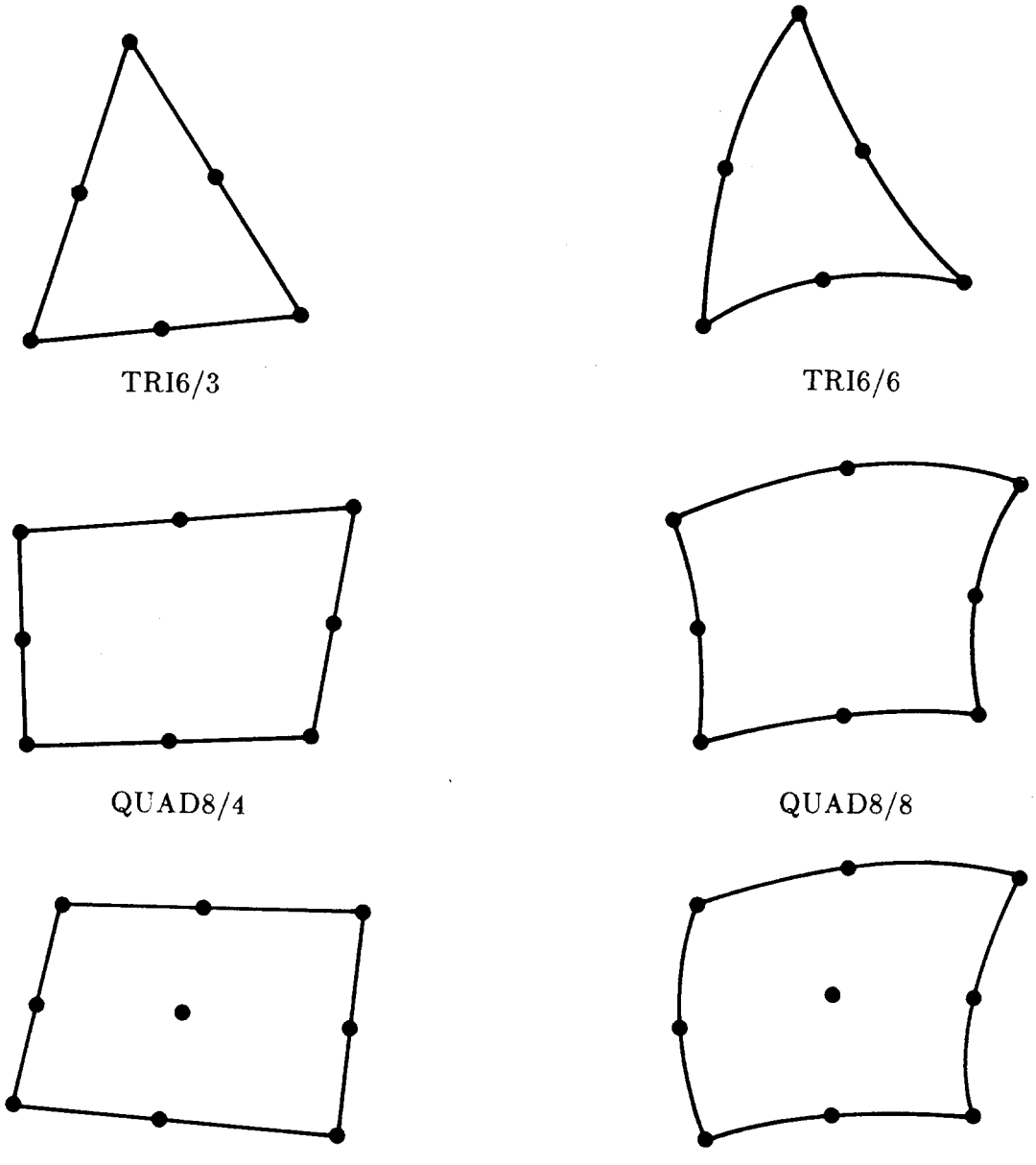


Figure 1: Finite elements in the NACHOS II library.

- PRTLMT (**OUTPUT**) - controls editing of output file
- POINTS (**OUTPUT**) - finds locations for special output points
- SOLVE (**SOLVE**) - controls the steady-state and transient solution procedures
- STREAM (**STREAM**) - generates the stream function field
- FLUXES (**FLUX**) - computes fluid stresses and diffusion fluxes
- XXXPLT (**PLOT**) - generates plots of various types
- GRPHFL (**POST**) - generates post-processing file

The large amount of data needed in a typical finite element analysis is handled in NACHOS II by a combination of disk files and in-core memory. Data that is repeatedly required by several of the major subroutines (*e.g.*, nodal point coordinates, nodal point connectivity and solution vectors) are stored in main memory. The in-core storage scheme makes use of a dynamic dimensioning algorithm that allocates needed array space during program execution. Less frequently needed data (*e.g.*, input data file) and extremely large blocks of data (*e.g.*, element level matrices and matrix solution data) are stored on various disk files. The utilization of the files shown in Figure 2 is discussed in Section 4.4.

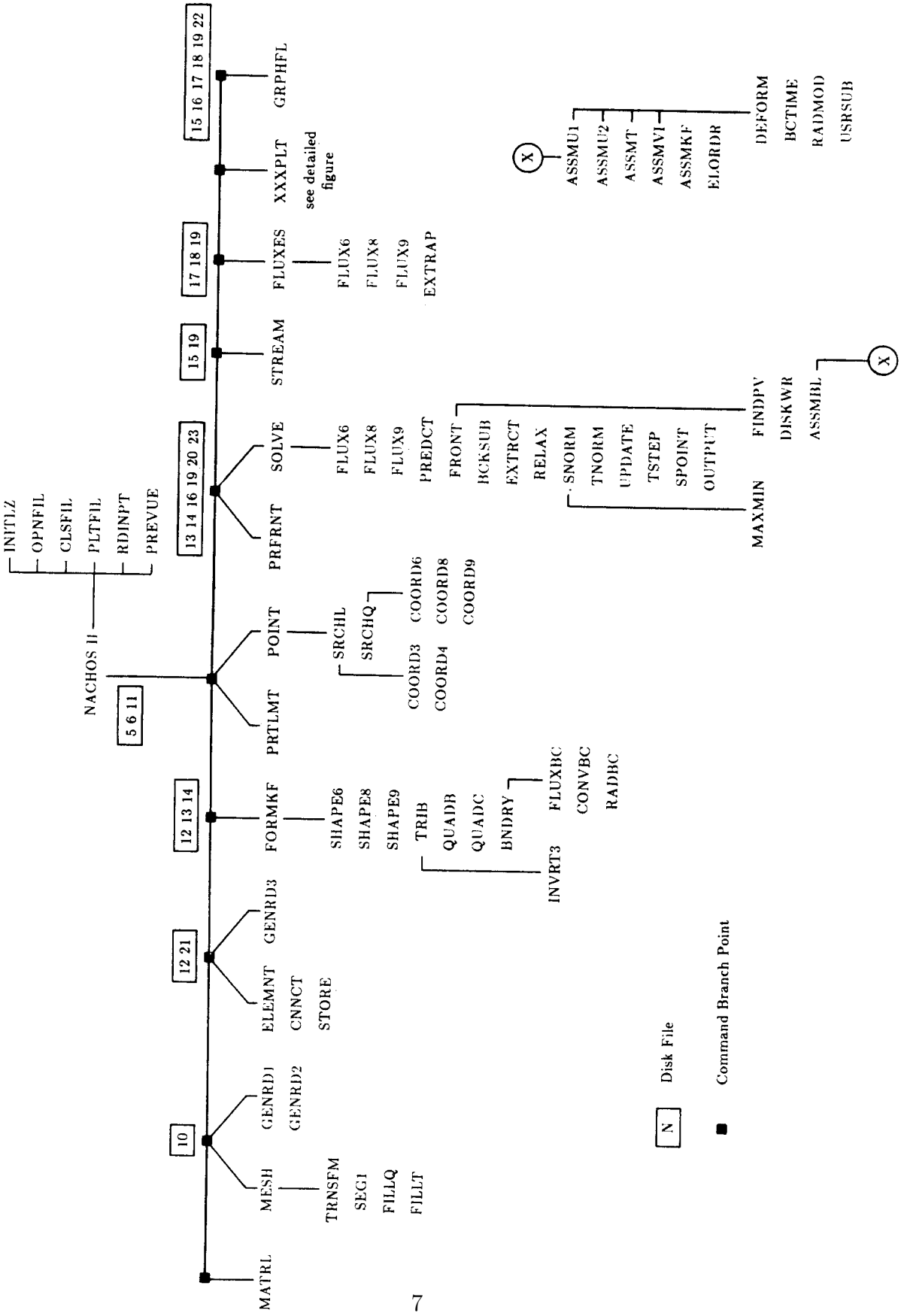


Figure 2: Organization of the NACHOS II code.

.

Figure 2 (Continued) : Organization of the NACHOS II code.

3 Input Guide

The structure of an input data file for the NACHOS II program directly reflects the steps required to formulate, solve and analyze a finite element model for a flow problem. Input to the code consists of four types of cards: a) Commentary cards, b) Command cards, c) Data cards and d) Termination cards. Following a title and optional comment lines, the program is directed by the various command cards to perform such functions as grid generation, element matrix construction, solution of the global matrix equations and calculation of auxiliary data. Data needed to perform these tasks are provided on data cards associated with the particular command card; data card sequences are ended with a termination card. The actual sequence of commands to the program is quite flexible, though there are some obvious limits to the order in which operations can be specified. In the following sections the commands and associated data required by NACHOS II are described in roughly the order in which they would appear in a typical input file. Notice that throughout this manual input data is referred to in terms of "cards"; this is a convenient term that in actuality refers to a record on an input device.

The commands recognized by NACHOS II are listed below in the order in which they are discussed in the subsequent sections. A summary showing the form and content of the indicated commands and associated data cards is provided in Appendix A.

- Title and Comment Cards
- MATERIALS Command Card
- MESH Command Card
- ELEMENTS Command Card
- FORMKF Command Card
- OUTPUT Command Card
- SOLVE Command Card
- STREAM Command Card
- FLUX Command Card
- PLOT Command Cards
- PLOTSET Command Card
- POST Command Card
- Termination Command Cards

3.1 Input Syntax

Most of the NACHOS II commands follow the standard format shown below:

COMMAND WORD, parameter list ...
.
data list ...
.
.
END

In a few cases a command does not require any additional data. When no data are associated with a command, the terminating **END** card is omitted.

In describing the input data, the following conventions have been adopted for the following sections:

(a) Upper case bold face words indicate required command and termination words, *e.g.*, **FORMKF**.

(b) Upper case words indicate a required alphanumeric input value, *e.g.*, BC.

(c) Lower case words and symbols imply that an alphanumeric or numerical value for the specified variable is expected, *e.g.*, xmax.

(d) All input values are specified in a free field format with successive variables separated by commas. All alphanumeric input data is limited to ten characters under this format; all alphanumeric input must be in an upper case format.

(e) The $ character may be used to end an input line. The remaining space on the line can be used for comments.

(f) The * character may be used to continue an input line onto a second data card. The continuation character should follow the last comma on the card to be continued.

(g) *Italics* indicate optional parameters which may be omitted by using successive commas in the input line. If the omitted parameter is not followed by any required parameters, no additional commas need be specified.

(h) ( ) indicates the data type for a particular input variable, *i.e.*, alphanumeric or character (C), real (R) or integer (I).

(i) < > indicates the default value for an optional parameter.

(j) The contents of each input line are indicated by underlining.

(k) All quantities associated with a coordinate direction are expressed in terms of the planar x, y coordinate system. The corresponding quantities for axisymmetric problems are obtained by the association of the radial coordinate, r, with x and the axial coordinate, z, with y.

(l) The global coordinate system used by the code is right-handed with the x (r) axis directed horizontally to the right and the y (z) axis directed vertically upward.

3.2 Title and Comment Cards

The title or header card must be the first card in an input data set for any particular problem. A $ symbol must appear in Column 1; the remaining 79 columns are available for a problem title. The problem title is used to label printed and plotted output. The title card is of the following form:

```
$ THIS IS AN EXAMPLE OF A PROBLEM TITLE
```

Following the title card a number of comment cards may be included to provide a description of the particular problem, modeling assumptions, material models, *etc.* Up to 10 comment lines may be specified; each line begins with a $ leaving the remaining 79 columns available for descriptive text. The comment lines are reproduced at the beginning of the printed output listing. Comment lines have the following form:

```
$ THESE ARE EXAMPLES OF COMMENT LINES
$ UP TO 10 LINES OF PROBLEM DESCRIPTION MAY
$ BE INCLUDED ON THESE CARDS
.
.
.
```

3.3 MATERIALS Command Card

The input of material properties to NACHOS II is accomplished through the **MATERIALS** command and its associated set of data cards. Input of material data is terminated by an **END** card. There are three basic types of material data cards corresponding to fluid/solid materials, porous materials and auxiliary equation data. The **MATERIALS** command is of the following form:

MATERIALS
mat name, mat type, mat number, ρ_0, μ, C, k, β, g_x, g_y, variable properties, *
Q, viscous dissipation, T_0, T_{init}
.
.
mat name, POROUS, mat number, fluid number, κ, μ_e, ϕ, $\hat{c}$, $(\rho_0 C)_e$, k_e, *
variable properties, Q, T_{init}
.
.
mat name, AUXDATA, mat number, C_1, D_1, γ_1, Q_1, c_{1_0}, $c_{1_{init}}$, *
C_2, D_2, γ_2, Q_2, c_{2_0}, $c_{2_{init}}$
.
END

where the parameters for each type of material card are listed below. NACHOS II does not contain any dimensional constants and, therefore, the units for the material properties are free to be chosen by the user. For convenience, a table of consistent units is given in Appendix B.

Fluid/Solid Property Data :

mat name (C) $<>$: is a material name for user reference.

mat type (C) : is a variable to indicate the type of material. The permissable values are NEWTONIAN for a Newtonian fluid, NNEWTONIAN for an inelastic, non-Newtonian fluid and SOLID for a solid (nonflowing) material (see Note 3).

mat number (I) : is an internal reference number for the material. NACHOS II will accept up to 10 materials; the materials must be numbered consecutively from 1 through 10.

ρ_0 (R) : is the material density.

μ (R) : is the material viscosity (omitted for a solid).

C (R) : is the material specific heat.

k (R) : is the material thermal conductivity.

β (R) : is the material thermal volumetric expansion coefficient.

g_x (R or C) : is the x component of the gravitational acceleration. When g_x is set to VARIABLE a variable orientation of the gravitational vector is specified; this option requires the use of a user supplied subroutine (see Notes 1 and 2).

g_y (R) : is the y component of the gravitational acceleration (see Note 1).

variable properties (C) <CONSTANT> : indicates the dependence of the material properties on the independent or dependent variables. When this parameter is omitted or set to CONSTANT, all of the material properties are assumed to have a constant value as specified on this material property card. A parameter value of VARIABLE indicates that one or more of the material properties is nonconstant; this option requires the use of a user supplied subroutine (see Note 3).

Q (R or C) <0.0> : is the volumetric heat source for the material. For no volumetric heating, this parameter is omitted or set to 0.0; for a constant volumetric heating this parameter is set to the heating rate. A variable heating rate is indicated by setting this parameter to VARIABLE; this option requires the use of a user supplied subroutine (see Note 4).

viscous dissipation (C) <NONE> : indicates if viscous dissipation is included in the analysis. If this parameter is omitted or set to NONE, the dissipation terms are omitted from the equations. A parameter value of DISSIPATE causes the dissipation effects to be included in the analysis.

T_0 (R) <0.0> : is the reference temperature for the buoyancy forces, *i.e.*, T_0 is the temperature at which buoyancy forces are zero.

T_{init} (R) <0.0> : specifies the initial temperature of the material.

Porous Media Property Data :

mat name (C) < > : is a material name for user reference.

mat number (I) : is an internal reference number for the material. NACHOS will accept up to 10 materials; the materials must be numbered consecutively from 1 through 10.

fluid number (I) : is the material number of the fluid that is assumed to saturate the porous material.

κ (R) : is the permeability of the porous material.

μ_e (R) : is the effective (Brinkman) viscosity for the fluid-saturated porous material.

ϕ (R) : is the porosity of the material.

$\hat{c}$ (R) : is the inertial coefficient associated with the Forchheimer term in the porous flow momentum equation.

ρC_e (R) : is the effective capacitance for the fluid-saturated porous layer.

k_e (R) : is the effective thermal conductivity for the fluid-saturated porous layer.

variable properties (C) <CONSTANT> : indicates the dependence of the material properties on the independent or dependent variables. When this parameter is omitted or set to CONSTANT, all of the material properties are assumed to have a constant value as specified on this material property card. A parameter value of VARIABLE indicates that one or more of the material properties is nonconstant; this option requires the use of a user supplied subroutine (see Note 5).

Q (R or C) <0.0> : is the volumetric heat source for the material. For no volumetric heating, this parameter is omitted or set to 0.0; for a constant volumetric heating this parameter is set to the heating rate. A variable heating rate is indicated by setting this parameter to VARIABLE; this option requires the use of a user supplied subroutine (see Note 4).

T_{init} (R) <0.0> : specifies the initial temperature of the fluid-saturated porous material.

Auxiliary Property Data (see Note 6):

mat name (C) < > : is a material name for user reference.

mat number (I) : is an internal reference number for the material. This number must be set to the "mat number" of the material for which the auxiliary transport equations are defined.

C_1, C_2 (R) : are the effective capacitance coefficients for the transport equations.

D_1, D_2 (R) : are the effective diffusion coefficients for the transport equations.

γ_1, γ_2 (R) : are the volumetric expansion coefficients for the auxiliary variables.

Q_1, Q_2 (R or C) : are the volumetric sources for the auxiliary transport equations. For no volumetric sources these parameters are omitted or set to zero; for constant volumetric sources these parameters are set to the value of the volumetric generation. Variable source terms are indicated by setting these parameters to VARIABLE; this option requires the use of a user supplied subroutine (see Note 4).

c_{1_0}, c_{2_0} (R) <0.0> : are the reference values for the buoyancy forces induced by the auxiliary variables c.

$c_{1_{init}}, c_{2_{init}}$ (R) <0.0> : specify the initial values of the variables c_1 and c_2.

Notes:

1) The components of the gravity vector are defined such that a positive value of the gravitational acceleration is directed opposite to the coordinate directions. Thus, with the x axis directed horizontally to the right, g_x is positive when directed to the left. The y axis is directed vertically upward and g_y is positive in the downward direction.

2) By setting the g_x parameter to VARIABLE, the orientation and magnitude of the gravity vector can be specified on an element by element basis. Under this option the g_y parameter is omitted and a user supplied subroutine USRGRV must be attached to the main program. The required format for this subroutine is given in Section 3.16.4.

3) When the "variable properties" parameter is set to VARIABLE, the material properties μ, k and β are assumed to be nonconstant. The variation of these properties must be specified to NACHOS II through the user supplied subroutines USRVIS, USRCON and USREXT. The formats for these subroutines are described in Section 3.16.2. When this option is invoked, representative values of μ, k and β should still be supplied on the material data card; these values are used to initialize the iterative solution algorithms. If a non-Newtonian material is defined, the "variable properties" parameter is automatically set to VARIABLE and the appropriate subroutines must be attached to the program (see Note 7).

4) When the volumetric heating is specified as being variable, NACHOS II expects the user supplied subroutine USRVHS to be attached to the program. The specification of a nonconstant volumetric source for one or more of the auxiliary transport equations requires the user subroutine USRVS to be provided. The volumetric source for either or both of the auxiliary equations is defined by the same subroutine. The required formats for all of the volumetric source subroutines are given in Section 3.16.3.

5) When the "variable properties" parameter is set to VARIABLE for a porous material, the material properties μ_e, k_e and β are assumed to be nonconstant. The evaluation of these properties is accomplished with the user subroutines USRVIS, USRCON and USREXT. These are the same subroutines used for the fluid/solid property variations (see Notes 3 and 7).

6) When the auxiliary transport equations are employed in a problem, it is assumed that the defined transport processes apply to *all* the materials in the problem. Therefore, an auxiliary data card must be present for each material defined including any porous materials. In some circumstances it may be possible to effectively eliminate an auxiliary transport process from one or more materials through a judicious choice of property data.

7) The diffusion coefficients, D_i, and/or the expansion coefficients, γ_i, for the auxiliary transport processes, can be specified to be variable properties. To invoke this option the "variable properties" parameter on the appropriate fluid/solid or porous material data card should be set to VARIABLE. NACHOS II then expects these auxiliary parameters to be evaluated from the user supplied subroutines USRDIF and USREX, respectively. The remaining properties, μ, k and β (or μ_e, k_e and β) must also be evaluated through the proper subroutines, even if they have constant values. The required formats for these subroutines are given in Section 3.16.2.

3.4 MESH Command Card

The grid points for the finite element mesh are input through the **MESH** command. The form of this command and its associated data has two forms, depending on the use of the internal mesh generator or the provision of mesh data from an external mesh generation program. The use of the internal mesh generator will be described here. The form of the **MESH** command for use with an external mesh generator is described in Appendix C.

In contrast to many finite element and mesh generation codes, NACHOS II separates the generation of nodal points and the generation of elements into distinct operations. The calculation of nodal point locations is accomplished by use of an isoparametric mapping scheme that considers quadrilateral and triangular parts or regions of the problem domain separately. For each part, a series of coordinates are specified which determine the shape of the region boundary. The node points within each part are identified by an I, J numbering system. The location of points in a region is controlled by user specification of the number of points along a boundary side and a local gradient parameter. Points may also be located along specified arcs or at specific points; previously generated groups of points can be moved to different locations by a copy or reflection operation. An iterative algorithm is also available for generating points in a bounded region. These ideas are more clearly fixed through a description of the specific command and data cards.

The **MESH** command has the following form

MESH, type of generator, imax, jmax, *iprint*
.
.
mesh data
.
.
END

where,

type of generator (C) : is a parameter to specify the type of mesh generator being used. To use the internal mesh generator set this parameter to INTERNAL.

imax, jmax (I) : specify the maximum I and J values used in the mesh.

iprint (I) <2> : specifies the amount of printed output produced during the mesh generation operations. Output increases with the value of *iprint* and ranges from no printout for *iprint* = 1 to a full printout at *iprint* = 4.

The mesh data associated with the above command varies according to the type of operation that is to be performed. Each grid generation operation is performed in sequence as the data is encountered (see Note 1). Reference to Figures 3 through 7 will aid the following descriptions of the different operations and parameters.

Quadrilateral Region (Figure 3) :

To generate grid points for each quadrilateral region or part in the domain, the following (3) input lines are required

QBLOCK, i1, j1, i3, j3, *g1, g2, g3, g4, polar, xpolar, ypolar*
x1, x2, x3, x4, *x5, x6, x7, . . . , x12*
y1, y2, y3, y4, *y5, y6, y7, . . . , y12*

where,

i1, j1, i3, j3 (I) : are the I, J limits for the region being generated as shown in Figure 3a. The difference between the I and J values (*i.e.*, i3-i1+1 and j3-j1+1) determines the number of grid points generated in a particular direction.

g1, g2, g3, g4 (R or I) <1.0> : specify the gradients for the grid point spacing along the four sides of the region. The gradients are defined in Figure 3a as the ratio of grid spacings at the end of each side. The default values of unity give equal grid point spacing along a side; gradients either larger or smaller than unity can be used to bias the point spacing in either direction (see Note 2).

polar (C) < > : specifies the use of polar coordinates for describing the coordinates of the points used to define the region. With this parameter set to POLAR the x's and y's on the second and third data cards are interpreted as polar radii and angles, respectively. If this parameter is omitted the coordinates retain their normal cartesian interpretation (see Note 3).

xpolar, ypolar (R) <0.0> : specify the local origin for use of the polar coordinate option.

x1, x2, . . . ,*x12* (R)

y1, y2, . . . ,*y12* (R) : define the coordinates of the four corner and optional side points for each part. If the region is bounded by straight lines only the four corner coordinates (*i.e.*, x1 to x4 and y1 to y4) need be specified. If any of the region sides are curved, then the appropriate side point, as shown in Figure 3b musyest be specified. If

one side point is defined, a quadratic interpolation is used to define the boundary; specification of two side points allows a cubic interpolation (see Note 4).

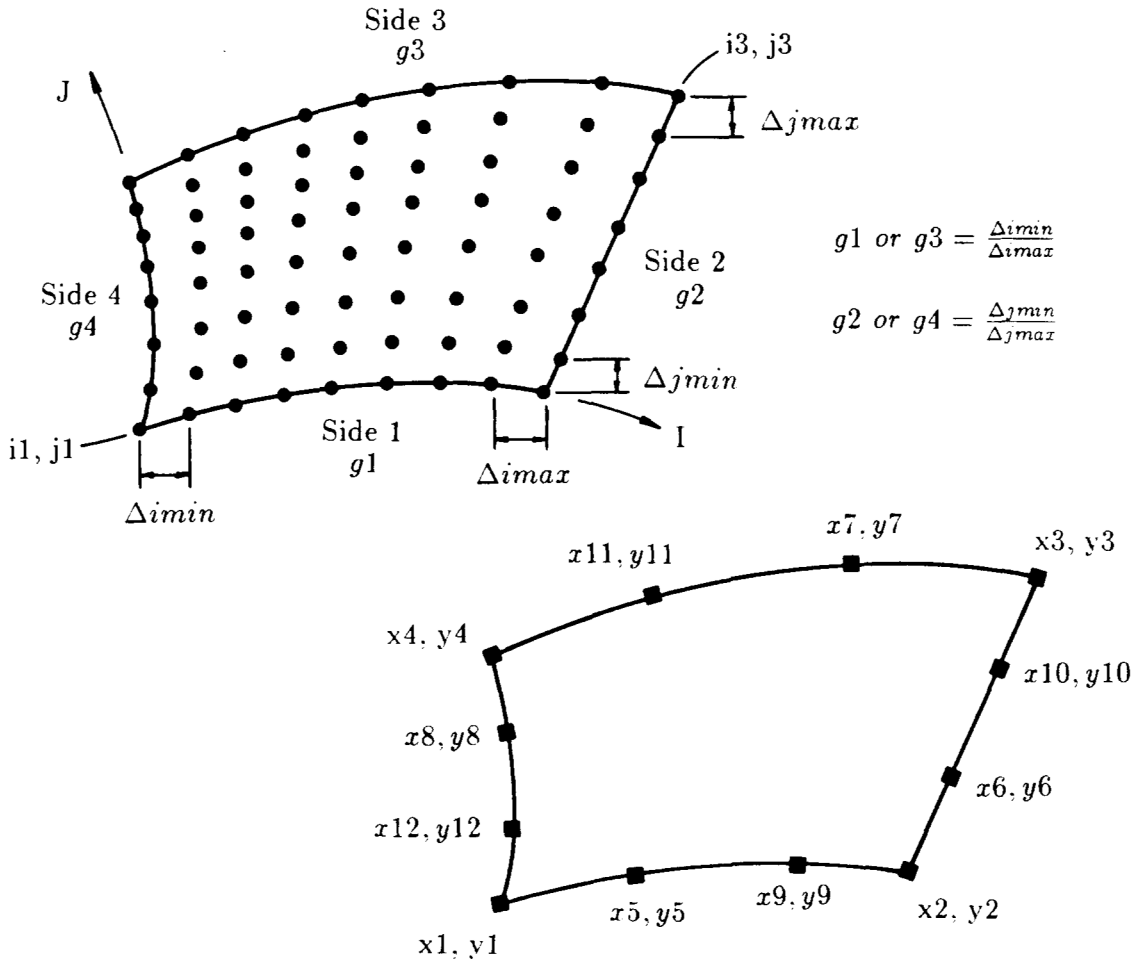


Figure 3: Mesh generation in a quadrilateral region.

Triangular Region (Figure 4) :

To generate grid points for each triangular region or part in the domain, the following (3) input lines are required

TBLOCK, i1, j1, i2, j2, i3, j3, *g1, g3, polar, xpolar, ypolar*
x1, x2, x3, *x4, x5, . . . , x9*
y1, y2, y3, *y4, y5, . . . , y9*

where,

i1, j1, i2, j2, i3, j3 (I) : are the I, J limits for the region being generated as shown in Figure 4a. The difference between i2 and i1 (*i.e.*, i2-i1+1) determine the number of grid points placed along the J = constant side of the region. The difference between j3 and j1 (*i.e.*, j3-j1+1) determine the number of nodes on the I = constant side of the region. The number of grid points on the I and J = constant lines *must* be the same (*i.e.*, i2-i1 = j3-j1) (see Note 5).

g1, g3 (R or I) <1.0> : specify the gradients for the grid point spacing along two sides of the region. The gradients are defined in Figure 4a as the ratio of grid spacings at the end of each side. The default values of unity give equal grid point spacing along a side; gradients either larger or smaller than unity can be used to bias the point spacing in either direction (see Note 2).

polar (C) < > : specifies the use of polar coordinates for describing the coordinates of the points used to define the region. With this parameter set to POLAR the x's and y's on the second and third data cards are interpreted as polar radii and angles. If this parameter is omitted the coordinates retain their normal cartesian interpretation (see Note 3).

xpolar, ypolar (R) <0.0> : specify the local origin for use of the polar coordinate option.

x1, x2, . . . , *x9* (R)

y1, y2, . . . , *y9* (R) : define the coordinates of the three corner and optional side points for each part. If the region is bounded by straight lines only the three corner coordinates (*i.e.*, x1 to x3 and y1 to y3) need be specified. If any of the region sides are curved, then the appropriate side point, as shown in Figure 4b must be specified. If one side point is defined, a quadratic interpolation is used to define the boundary; specification of two side points allows a cubic interpolation (see Note 4).

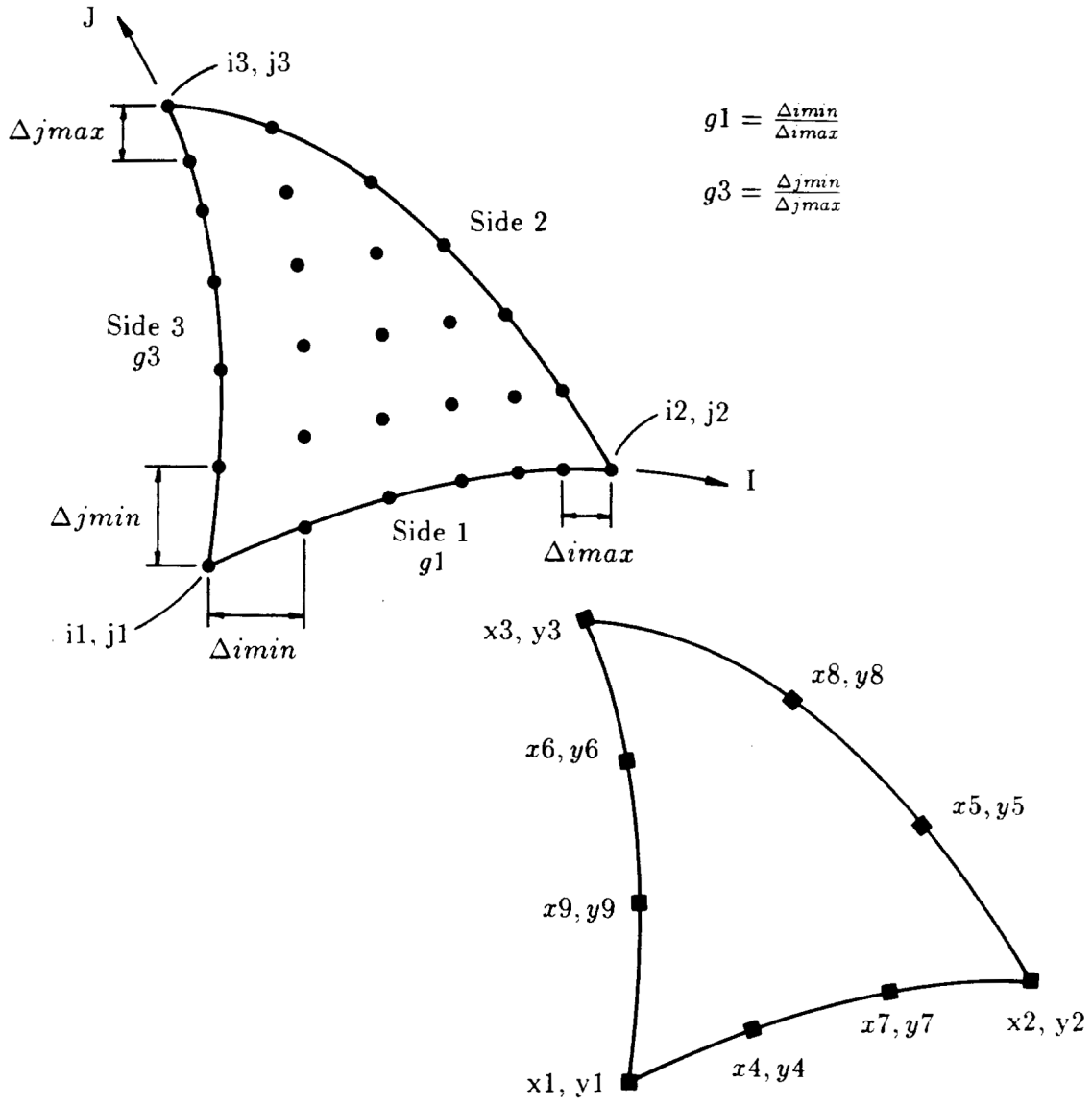


Figure 4: Mesh generation in a triangular region.

Arc (Figure 5) :

Grid points can be distributed along a specified arc by the (3) input lines

ARC, i1, j1, i2, j2, *g1, polar, xpolar, ypolar*
x1, x2, *x3, x4*
y1, y2, *y3, y4*

where,

i1, j1, i2, j2 (I) : are the I, J limits for the line being generated as shown in Figure 5. Since a one-dimensional array of points is being generated, either i1 = i2 or j1 = j2.

g1 (R or I) <1.0> : specifies the gradient for the grid point spacing along the arc. The gradients are defined in Figure 5 as the ratio of grid spacings at the ends of the line. The default value of unity gives a uniform grid point spacing along the line; a gradient either larger or smaller than unity can be used to bias the point spacing in either direction (see Note 2).

polar (C) < > : specifies the use of polar coordinates for describing the coordinates of the points used to define the arc. With this parameter set to POLAR the x's and y's on the second and third data cards are interpreted as polar radii and angles. If this parameter is omitted the coordinates retain their normal cartesian interpretation (see Note 3).

xpolar, ypolar (R) <0.0> : specify the local origin for use of the polar coordinate option.

x1, x2, *x3, x4* (R)

y1, y2, *y3, y4* (R) : define the coordinates of the two end points and optional intermediate points for each arc. If the arc is a straight line only the two end coordinates (*i.e.*, x1, y1 and x2, y2) need be specified. If the arc is curved, then the appropriate intermediate points, as shown in Figure 5 must be specified. If one intermediate point is defined, a quadratic interpolation is used to define the arc; specification of two intermediate points allows a cubic interpolation (see Note 4).

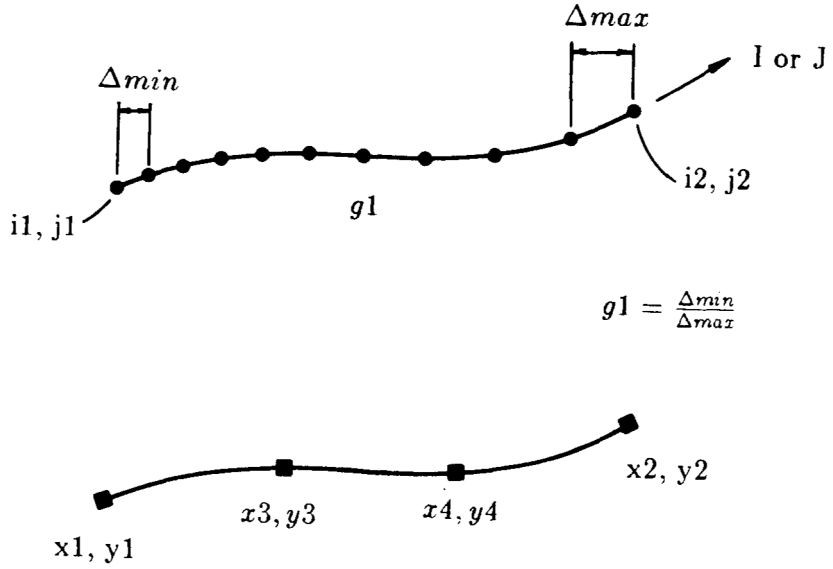


Figure 5: Generation of points along an arc.

Point :

Individual grid points can be specified by the (1) input line

POINT, i1, j1, x1, y1, *polar, xpolar, ypolar*

where,

i1, j1 (I) : is the I, J name for the point being generated.

xl, y1 (R) : are the coordinates for the point

polar (C) : specifies the use of polar coordinates as described previously (see Note 3).

xpolar, ypolar (R) <0.0> : specify the local origin for use of the polar coordinate option.

Copy (Figure 6) :

A distribution of mesh points that have been previously defined may be replicated at another spatial location by use of the Copy operation. This is accomplished by the (2) input lines

COPY, i1, j1, i3, j3, inew1, jnew1, inew3, jnew3
xnew1, ynew1, xnew3, ynew3, *polar, xpolar, ypolar*

where,

i1, j1, i3, j3 (I) : are the I, J limits for the existing mesh points that are to be copied as shown in Figure 6.

inew1, jnew1, inew3, jnew3 (I) : are the I, J limits for the new region that is being created by the copy operation (see Note 6).

xnew1, ynew1, xnew3, ynew3 (R) : are the spatial coordinates that locate the postion of the newly created region.

polar (C) < > : specifies the use of polar coordinates for describing the coordinates of the points used to define the new region. With this parameter set to POLAR the xnew's and ynew's are interpreted as polar radii and angles. If this parameter is omitted the coordinates retain their normal cartesian interpretation (see Note 3).

xpolar, ypolar (R) <0.0> : specify the local origin for use of the polar coordinate option.

Reflect (Figure 7) :

A distribution of mesh points that have been previously defined may be reflected about an arbitrary line via the Reflect operation. This is accomplished by the (2) input lines

REFLECT, i1, j1, i3, j3, inew1, jnew1, inew3, jnew3
xline1, yline1, xline2, yline2, *polar, xpolar, ypolar*

where,

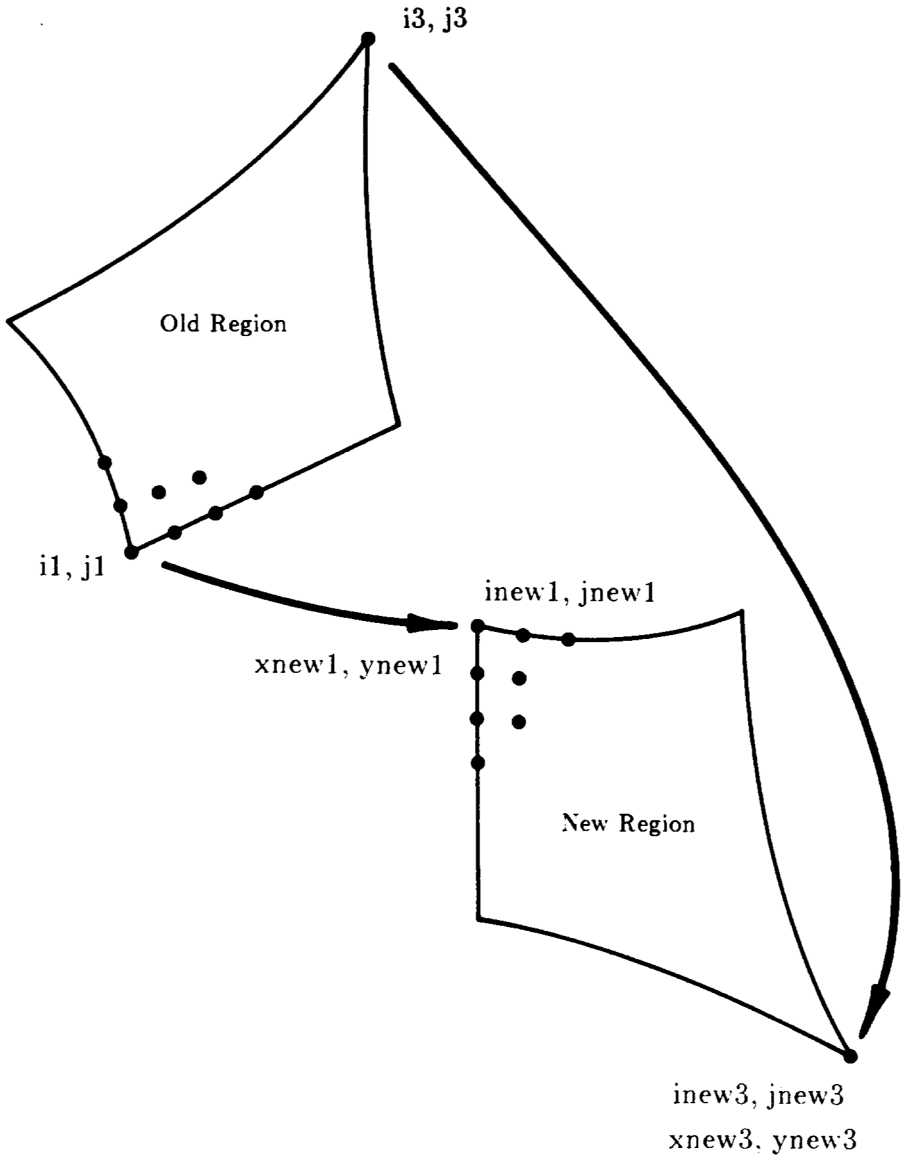


Figure 6: Copying of mesh points to a new region.

i1, j1, i3, j3 (I) : are the I, J limits for the existing mesh points that are to be reflected as shown in Figure 7.

inew1, jnew1, inew3, jnew3 (I) : are the I, J limits for the new region that is being created by the reflection operation (see Note 6).

xline1, yline1, xline2, yline2 (R) : are the spatial coordinates that describe the line about which the reflection is to occur.

polar (C) < > : specifies the use of polar coordinates for describing the coordinates of the points used to define the reflection line (see Note 3).

xpolar, ypolar (R) <0.0> : specify the local origin for use of the polar coordinate option.

Fillin :

An empty quadrilateral space that is bounded by previously defined grid points can be filled through the use of an iterative algorithm. The algorithm uses a weighted average of a standard "Laplace" grid generation scheme and an equal area scheme. All of the nodes along the boundaries of the region must have been previously defined; any previously defined points located in the interior of the region will not be redefined by this procedure. This operation is invoked by the (1) input line

FILLIN, i1, j1, i3, j3, α, ω, *iter, tol*

where,

i1, j1, i3, j3 (I) : are the I, J limits for the region to be filled using the iterative algorithm.

α (R) <0.5> : determines the weighting between the Laplace scheme and the equal area scheme. For α equal to zero, a pure Laplace solution is generated. When α is unity the equal area algorithm is used.

ω (R) <1.86> : is an overrelaxation factor for the iteration process.

iter (I) <100> : specifies the maximum number of iterations allowed before the algorithm is terminated.

tol (R) <0.0001> : is the convergence tolerance for the iteration procedure.

External Definition :

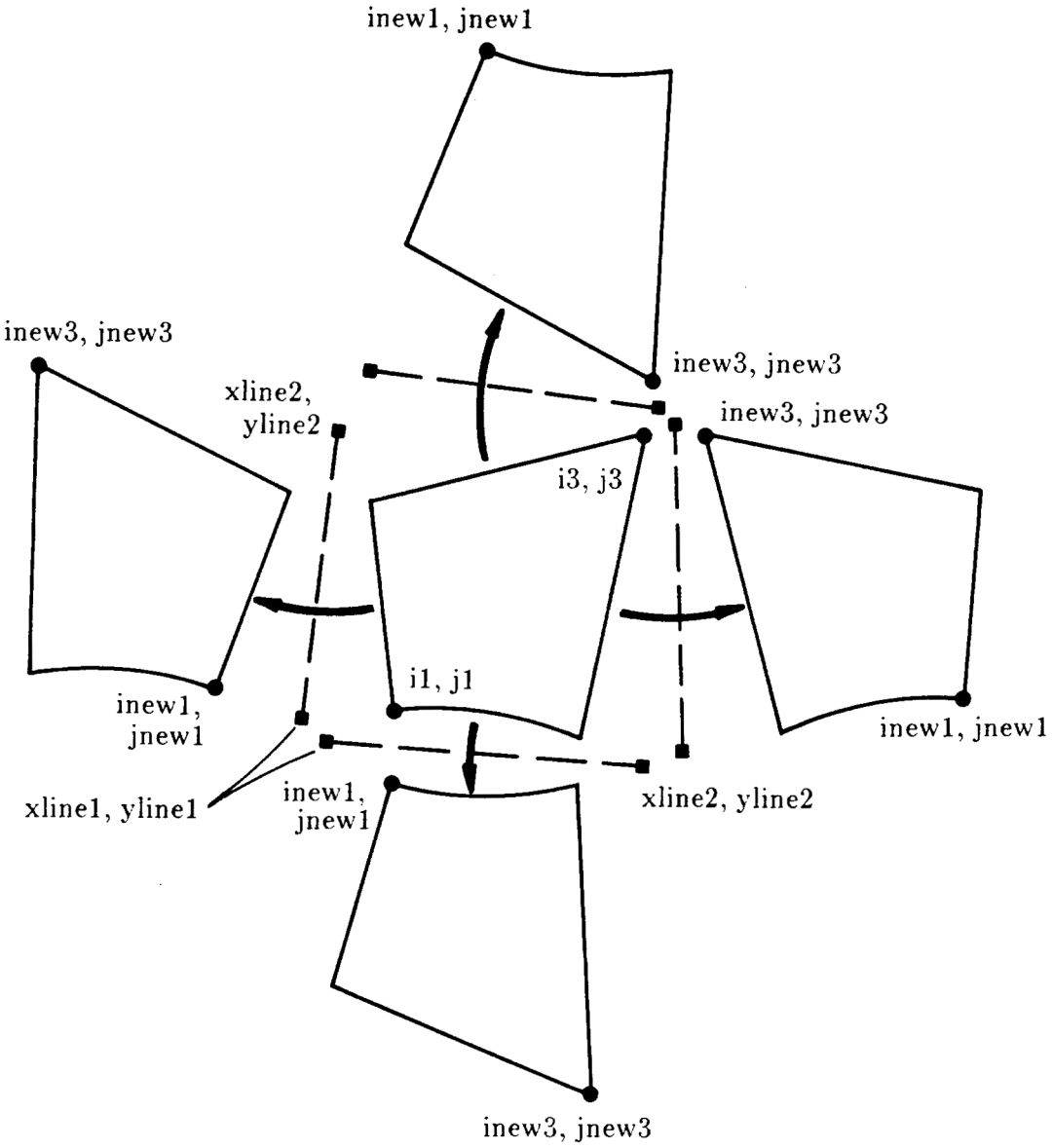


Figure 7: Reflection of mesh points about a line.

Mesh points for all or some of the regions of a problem can be read into NACHOS II from a user supplied subroutine (see Note 7). This is especially useful when it is required to arrange grid points according to an analytic expression, *e.g.*, circular arcs, conic sections, *etc.* This operation is invoked by the (1) input line

EXTDEF

Notes:

1) Each point in the mesh is labeled with an I, J name as it is generated by a particular operation. Points which have been defined will normally be redefined if they are referenced in a subsequent operation; the last operation is the one that prevails.

2) The use of a gradient parameter allows the ratios of mesh point intervals to be specified but does not allow a particular mesh point spacing to be set. Two simple formulas are available to determine the gradient needed to obtain a particular mesh interval at the ends of the boundary lines. The gradient, g_{pt1}, required to set the first mesh point a distance d_1 from the end of the boundary curve is given by

$$g_{pt1} = \frac{d_1(npts-1)}{2L - d_1(npts-1)}$$

where $npts$ is the number of mesh points to be placed along the curve and L is the distance between the end points of the boundary curve. The gradient, g_{pt2}, required to set the second mesh point a distance d_2 from the end of the curve is given by

$$g_{pt2} = \frac{2L - d_2(npts-2)(npts-1)}{6L + d_2(npts-2)(npts-1) - 4L(npts-1)}$$

For the higher-order elements used in NACHOS II, the gradient g_{pt2} would be of most use since it generally corresponds to setting the length of an element side. The g_{pt1} would correspond to the midside node setting for an isoparametric element. Note that midside nodes for subparametric elements are always reset to the midpoint of the element side.

3) When the polar option is specified a local origin is established at *xpolar, ypolar* with the radii then measured from this location. The polar angles are measured from the positive x axis and are positive in a counterclockwise direction. The polar option is strictly a user convenience; all coordinates used internally by NACHOS II are in a cartesian coordinate system.

4) For a quadratic boundary interpolation the intermediate point should be located near the midpoint of the region boundary or arc. A cubic interpolation should have the intermediate points at the one-third points of the boundary line. If the intermediate points deviate significantly from these locations, biasing of the mesh points along the curve may occur even without a gradient parameter being specified. Very large deviations in the intermediate point locations can lead to a singularity in the isoparametric mapping for the curve.

5) The triangular mesh generation option is unusual in that only two sides of the region have I or J = constant; the third side corresponds to a surface with I and J both being nonconstant. Gradients may only be specified on the I and J = constant boundaries. The triangular operation would typically be used to provide a transition between two rectangular blocks such as in an elbow or corner region.

6) In the copy and reflection operations the original quadrilateral block of points defined by i1, j1, i3, j3 is always considered as the standard with respect to mesh point location and I, J numbering. Any discrepancies that occur in the numbering of points in the old and new blocks (*e.g.*, i3-i1 not being equal to inew3-inew1) are resolved in favor of the old block. The coordinates xnew1, ... ynew3 in the copy operation are only used to establish the translational and rotational parts of the mesh transfer operation. These coordinates do not influence the size of the newly created block of points or the location of mesh points within the block. In the reflection operation the inew and jnew points should be located as mirror reflections of the original i and j points as shown in Figure 7. The inew and jnew values determine the i,j sequencing in the reflected part of the mesh. An example of the use of the reflect operation is shown in the example problems section.

7) The external definition option requires that the user subroutine USRMSH be attached to the NACHOS II program. The format for this subroutine is defined in Section 3.16.1.

3.5 ELEMENTS Command Card

The element and boundary condition data are input through the **ELEMENTS** command and its various data sets. The format for the data in this command depends on the use of the internal or an external mesh generation program. The input of data in conjunction with the internal mesh generator is described here. The form of this command for use with an external mesh generator is described in Appendix C.

As noted in the **MESH** command description, the mesh points for the problem geometry are generated independently of the finite elements. This permits great flexibility in constructing elements from a given mesh point arrangement. The construction or description of an element consists of identifying an appropriate group of previously defined mesh points to serve as the corner, midside and center nodes in the element. This concept will become apparent from the form of the element data associated with the present command card. Boundary condition specification is on an element by element basis and is also tied to the mesh point description.

The **ELEMENTS** command has the following form

ELEMENTS, no. elements, *order*, *iprint*
.
.
element data
.
boundary condition data
.
.
END

where,

no. elements (I) : is the number of elements in the mesh. This number need only be an upper bound on the number of elements since it is used only to allocate temporary internal work space.

order (C) < > : determines the numbering of the elements. If *order* is omitted the elements are numbered by increasing I, J values (*e.g.*, (1,1), (2,1), (3,1), ... (1,2), (2,2), ...). When *order* is set to PRESCRIBED, the elements are numbered according to their order in the input list (see Note 1).

iprint (I) <2> : determines the amount of printed output produced during the element operations. Output increases with the value of *iprint* and ranges from no printout for *iprint* = 1 to full printout for *iprint* = 4.

The data associated with this command is of two types. One data set describes the finite elements to be used in the problem while the second data set specifies the boundary conditions for the problem. Though the two types of data may be intermixed in the input deck, they are described here in separate sections.

Element Data :

Elements in the mesh are specified by the element data cards following the **ELEMENTS** command line. Each element is generated by a data card of the form

element type, mat, i1, j1, *i2, j2, i3, . . ., in, jn*

where,

element type (C) : is the name of the type of element. The element types available in NACHOS II are described below.

mat (I) : is the material number for the element. This number should be set to correspond to the "mat number" parameter used on the material property cards.

i1, j1, *i2, j2, . . in, jn* (I) : is the list of I, J values for the node points in the element. The nodes of an element are listed counterclockwise around the element starting with any corner as shown in Figure 8. In some situations, the list of I, J values may be significantly condensed. When only the first node I, J values are specified for a quadrilateral element, the values for the remaining nodes are assumed to be,

i4 = i8 = i1, i5 = i7 = i1 + 1, i2 = i3 = i6 = i1 + 2, i9 = i5 = i7

j2 = j5 = j1, j6 = j8 = j1 + 1, j3 = j4 = j7 = j1 + 2, j9 = j6 = j8

When I, J values are specified for only the corner nodes of any element, the midside I, J values are computed as the average of the corner values (see Notes 2 and 3).

The specification of different types of elements is provided by the "element type" parameter on the previously described data card. The permissible element names for this parameter include the following (see Note 4) :

(a) TRI6/3 – A subparametric triangle with arbitrarily oriented straight sides.

(b) TRI6/6 – A general isoparametric triangle with quadratic interpolation used to define the shape of the element sides.

(c) QUAD8/4 – A subparametric quadrilateral with arbitrarily oriented straight sides.

(d) QUAD8/8 – A general isoparametric quadrilateral with quadratic interpolation used to define the shape of the element sides.

(e) QUAD9/4 – A subparametric quadrilateral with arbitrarily oriented straight sides.

(f) QUAD9/9 – A general isoparametric quadrilateral with quadratic interpolation used to define the shape of the element sides.

In all of the above elements, the primary dependent variables (velocity, temperature and auxiliary variables) are approximated with biquadratic interpolation functions. The pressure is approximated using either a continuous, bilinear approximation or a discontinuous, linear approximation. A penalty function formulation of the incompressibility constraint can be used with the discontinuous pressure form of each element. These options are selected on the **FORMKF** command card described in Section 3.6. Further details on the formulation of these elements are available in [3].

Boundary Condition Data :

The boundary conditions for the problem are specified by element and may appear at any point in the present data section after the element to which they apply has been defined. Boundary conditions are specified to have either a uniform value along an element side or a particular value at a node. The boundary condition data card has the following form

BC, b.c. type, i1, j1, side/node, b.c. data

where,

b.c. type (C) : is the name for the type of boundary condition. The types of boundary conditions available in NACHOS II are described below.

i1, j1 (I) : is the I, J identification of the element (*i.e.*, the first I, J named on the element data card) to which the boundary condition applies.

side/node (I) : identifies the side or node of the element to which the boundary condition is to be applied. The numbering of nodes and sides begins with the identifying node (*i.e.*, the first node named on the element data card) and proceeds counterclockwise as shown in Figure 8.

b.c. data (R or I) : is the numerical value of the applied boundary condition or an integer pointer to a location containing boundary condition data. For a specified boundary condition, this parameter is set equal to the numerical value of the boundary value. For boundary conditions requiring multiple input parameters (*e.g.*, heat transfer coefficient and reference temperature) the "b.c. data" parameter is assigned an integer SET number; the use of the SET data card is described below. Boundary conditions that vary with time or other parameters use the "b.c. data" parameter to specify an integer function number for use with a user supplied subroutine (see Note 5).

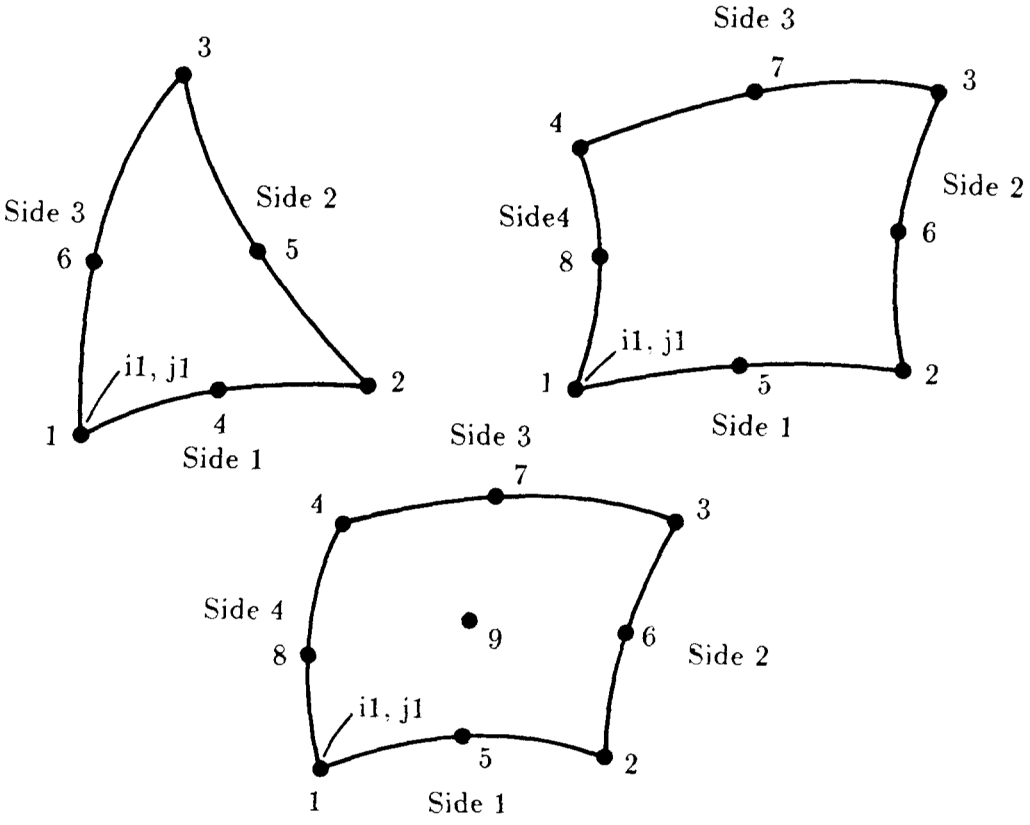


Figure 8: Side and node numbering for elements.

The permissible types of boundary conditions as specified by the "b.c. type" parameter include the following:

(a) U, V – specify the velocity components in the x and y directions at a node, b.c. data = specified velocity components (R).

(b) USIDE, VSIDE – specifies the velocity components to have a constant, uniform value along an element side, b.c. data = specified velocity components (R).

(c) STICK – specifies both velocity components to have a zero value along an element side, b.c. data = specified velocity components (R).

(d) UVARY, VVARY – specifies the velocity components along an element side to vary with time or some other parameters, b.c. data = specified velocity components (R).

(e) P – specifies the pressure at a corner node, b.c. data = specified pressure (R).

(f) TNRMLSIDE – specifies a constant (total) normal stress along an element side, b.c. data = specified stress (R).

(g) TNRMLVARY – specifies the (total) normal stress along an element side to vary with time or some other parameters, b.c. data = function number (I).

(h) TSHRSIDE – specifies the shear stress to have a constant, uniform value along an element side, b.c. data = specified stress (R).

(i) TSHRVARY – specifies the shear stress along an element boundary to vary with time or some other parameters, b.c. data = function number (I).

(j) T – specifies the temperature at a node, b.c. data = specified temperature (R).

(k) TSIDE – specifies the temperature to have a uniform, constant value along an element side, b.c. data = specified temperature (R).

(l) TVARY – specifies the temperature along an element side to vary with time or some other parameters, b.c. data = function number (I).

(m) QSIDE – specifies a uniform, constant heat flux along an element side, b.c. data = specified flux (R).

(n) QVARY – specifies the heat flux on an element side to vary with time or some other parameters, b.c. data = function number (I).

(o) QCONV – specifies a uniform, constant convective heat transfer process along an element side, b.c. data = set number (I).

(p) QRAD – specifies a radiative heat transfer or generalized convective process along an element side, b.c. data = set number (I).

(q) V1, V2 – specifies the auxiliary variables at a node, b.c. data = specified auxiliary variable (R).

(r) V1SIDE, V2SIDE – specifies a uniform, constant value of the auxiliary variables along an element side, b.c. data = specified auxiliary variable (R).

(s) V1VARY, V2VARY – specifies the auxiliary variables on an element side to vary with time or some other parameters, b.c. data = function number (I).

(t) FLUX1SIDE, FLUX2SIDE – specifies a uniform, constant value for the fluxes of an auxiliary variable along an element side, b.c. data = specified flux (R).

(u) FLUX1VARY, FLUX2VARY – specifies the auxiliary variable flux values along an element side to vary with time or some other parameters, b.c. data = function number (I).

All of the above boundary conditions can be employed with any of the previously described elements (see Note 6).

In some of the allowed boundary conditions more than one piece of data is needed to formulate the boundary value. This data is collected and stored using an integer pointer and a special type of data card. The use of the convective and radiative type of boundary conditions (QCONV, QRAD) require the specification of appropriate heat transfer coefficients (h_c, h_r) and reference temperatures (T_c, T_r). These parameters are input through the SET data card which has the following form

SET, b.c. type, set no., param1, param2

where,

b.c. type (C) : is the name for the type of boundary condition for which data is being specified, *i.e.*, either QCONV or QRAD.

set no. (I) : is the number of the particular data set. NACHOS II allows up to twenty data sets for each type of boundary condition. The "set no." is used to identify the appropriate boundary condition set on the BC card.

Param1, param2 (R or C) : are the parameters required for specifying the appropriate boundary condition data. For the convection case (QCONV), param1" is set to the heat transfer coefficient (h_c) and "param2" is set to the convection reference temperature (T_c). For the radiation case (QRAD), "param1" is set to the product of the surface form factor ($\mathcal{F}$, a function of the surface emissivity) and the Stefan-Boltzmann constant (σ) and "param2" is set to the reference temperature for radiation (T_r). If "param1" is set to VARIABLE under this option, a generalized convection/radiation process is employed that requires a user subroutine (see Note 7). The SET data cards may appear anywhere in the present data section; SET cards for both types of boundary condition may appear in the same problem.

Looping Feature :

In order to permit easy specification of elements and boundary conditions which appear in regular patterns in the mesh, a looping feature is incorporated in NACHOS II. This feature allows the definition of ILOOP's and JLOOP's (which are similar to FORTRAN DO loops) for incrementing data in the I and J directions. Nesting of the loops may be in any order but no more than one loop in a given direction may be used at one time. Note that within each loop all the I (or J) values are given the same increment. The looping commands are of the following general form

ILOOP, npass, inc
.
.
element or boundary
condition data
.
.
IEND

or when used in a nested form

JLOOP, npass, inc
ILOOP, npass, inc
.
.
element or boundary
condition data

.
.
IEND
JEND

where,

npass (I) : specifies the number of passes to be made through the loop.

inc (I) : specifies the increment to be added to the I or J values found within the loop. This parameter may be negative.

The looping commands may appear at any point within the element and boundary condition data. The use of the looping feature is illustrated in the example problems section.

Notes:

1) The element ordering should be chosen such that the front width for the global matrix problem is minimized. For a basically rectangular shaped geometry, elements should be numbered across the dimension of the domain containing the fewest elements.

2) Each element is identified by the I, J values for the first node named on the element data card, *i.e.*, i1, j1. All references to a particular element(*e.g.*, in specifying boundary conditions) are made via this I, J name. Since any corner node may be named first, two or more elements may share the same I, J designation or name. This situation must be avoided if any of the elements sharing names are to have boundary conditions imposed on them. A simple reordering of the nodes in the appropriate elements remedies this situation.

3) During the generation of grid points (**MESH** command) there is no requirement that adjacent parts of the grid have a continuous I, J numbering. When elements are constructed along boundaries of such adjacent parts of the grid, it is imperative that common nodes between elements have the same I, J values. Element connectivity is generated from the I, J values for each node and, therefore, common nodes with different I, J labels will not be properly connected. The generation of element meshes using node points with noncontinuous I, J labels is illustrated in the example problem section.

4) In the element name the first number indicates the number of nodes in the element at which the primary dependent variables are defined (*i.e.*, velocity components, temperature and auxiliary variables); the second number defines the number of nodes used to describe the element geometry. Note that in the generation of the subparametric elements, the physical coordinates of the mid-side (and center) nodes are adjusted to lie at the midpoint of the element sides.

5) Boundary conditions that vary with time or some other parameters (*i.e.*, those with a VARY suffix in the "b.c. type" parameter) require that one or more user supplied subroutines USRBCn be attached to the NACHOS II program. The correspondence between the particular boundary condition and the subroutine specifying its variation is established through the integer "function number". On a particular BC data card, the "b.c. data" parameter must be set to the integer "function number", n, which appears in the USRBCn subroutine name. The required formats for these subroutines are given in Section 3.16.5.

6) There are a few limitations on the number of boundary conditions of some types that may be applied to a single element. The TNRMLVARY and TSHRVARY conditions cannot both appear on a single element while the TNRMLSIDE and TSHRSIDE versions of these conditions are not restricted. Also, only one QRAD, QCONV, or other type of VARY condition may be specified for any single element. If two sides of an element require these types of conditions, the element should be split into two elements (*e.g.*, two triangles) with each receiving one of the needed boundary conditions.

7) When "param1" set to VARIABLE under the QRAD boundary condition, NACHOS II requires the user supplied subroutine USRHTC to be attached to the program. This subroutine is designed to evaluate *all* heat transfer coefficients that may be specified in the problem. The different boundary conditions are distinguished by the "set no." parameter. The format for this subroutine is outlined in Section 3.16.5.

3.6 FORMKF Command Card

The formulation and evaluation of the individual element matrices for a particular type of problem is carried out by the **FORMKF** command. This command has the form

FORMKF, *geometry, flow type, f.e. method, penalty parameter*, *
pressure approximation, var1, var2

where,

geometry (C) <PLANAR> : indicates the type of coordinate system used for the problem. A two-dimensional, planar formulation is obtained by omitting this parameter or setting it equal to PLANAR; a parameter value of AXISYM invokes the two-dimensional, axisymmetric formulation.

flow type (C) <ISOTHERMAL> : specifies the type of problem being solved. If *flow type* is omitted or set to ISOTHERMAL, the problem is treated as an isothermal flow problem (no transport equations). Forced convection problems are specified by setting *flow type* to FORCED (transport equations included, no buoyancy forces); free or mixed convection problems are indicated by a parameter setting of FREE (transport equations included, buoyancy forces included).

f.e. method (C) <MIXED> : specifies the type of finite element formulation used in treating the incompressibility constraint. If this parameter is omitted or set equal to MIXED, a standard mixed method is employed in which the pressure is directly approximated and the continuity equation is retained in the matrix problem. If this parameter is set to PENALTY then a penalty function approximation for the incompressibility condition is employed.

penalty parameter (R) <1.0E-8> : is the value of the penalty parameter used when the penalty method is invoked.

pressure approximation (C) <BILINEAR> : specifies the type of pressure approximation used in the element formulation. A value of BILINEAR forces the use of a continuous, bilinear approximation for the pressure within each element; pressure is defined at the corner nodes in an element. The parameter setting LINEAR requests a discontinuous, linear approximation to the pressure; pressure is defined on the interior of an element (see Notes 1 and 3).

var1, var2 (C) < > : specifies the inclusion of the auxiliary transport equations. By setting these parameters to AUXVAR1 and AUXVAR2, respectively, the auxiliary equations are included in the problem formulation (see Note 2).

Notes:

1) A bilinear pressure approximation can be used only with the mixed finite element formulation. The discontinuous, linear pressure representation can be used with either the mixed or penalty methods.

2) The auxiliary transport equations can be invoked singly or as a pair. The auxiliary equations cannot however, be used without the inclusion of the energy equation. If an isothermal flow with an additional transport equation is required, the energy equation should be used to represent the added physical process. The auxiliary transport equations, when used, are defined for *all* materials used in the problem.

3) When solid materials are present in a problem the discontinuous pressure approximation is normally recommended. In NACHOS II solids are represented as fluids with a very large viscosity. A continuous pressure field is not appropriate in this case since "pressures" in the solid are not relevant to the problem and may cause an unrealistic pressure response in adjoining fluid regions.

3.7 OUTPUT Command Card

Editing of the printed output from NACHOS II and the selection of special output points is controlled by the **OUTPUT** command. When used to selectively limit the printed output from NACHOS II this command and its associated data have the following form

OUTPUT, FIELDS
delimiter type, e1, e2, e3, e4, . . . , e50
.
.
END

where,

delimiter type (C) : specifies how the subsequent element numbers are to be interpreted in limiting the printed output. If delimiter type is set to SINGLE, individual element numbers are assumed to be listed in the following data. If delimiter type is set to STRING, the following data is interpreted as pairs of element numbers, with each pair defining the limits on a sequence of elements.

e1,e2,e3, . . e50 (I) : is a list of element numbers indicating which elements are to have data printed during the solution process. A maximum of fifty individual element numbers or twenty-five element pairs may be specified in a single data line (see Note 1).

The **OUTPUT** command can also be used to specify special locations within the problem domain at which values of the field variables are to be found and reported. In general, these locations will not be nodal points of the finite element mesh. The form of the command and data in this case is

OUTPUT, POINTS
x1, y1, x2, y2, . . . , x25, y25
.
.
END

where,

x1, y1, . . .x25, y25 (R) : is a list of the coordinate locations for the special points. A maximum of twenty-five points may be specified on any single data card (see Note 2).

Notes :

1) There is no limit to the number of individual data cards that can be associated with any given **OUTPUT** command; multiple **OUTPUT** commands are also permissable. Data cards with different delimiter types may be mixed within the same command. The **OUTPUT** command must precede the solution command in the input deck. If no **OUTPUT** commands are used in an input deck, the entire solution field is printed at each output interval.

2) NACHOS II allows a maximum of fifty special points to be specified during any particular analysis. The **OUTPUT** command in this form must precede the solution command.

3) **OUTPUT** commands of both forms may be present in the same input deck. The dependent variables at the special output locations are printed at each normal output interval. The variables at these points are also stored for later plotting.

3.8 SOLVE Command Card

The assembly and solution of the global system of matrix equations, for either steady state or transient problems is carried out by the **SOLVE** command. The data cards associated with this command differ according to the type of problem being solved. The general form of the command is

SOLVE, *restart*, *timeplane*, *maxmem*
.
.
steady-state or transient
data cards
.
.
END

where,

restart (C) < > : specifies if a restart from a previous solution file is required. If this parameter is omitted, the initial dependent variable fields are set from the data on the material property card. If this parameter is set to RESTART, the field variables are initialized from an input file (see Note 1).

timeplane (I) <1> : specifies the timeplane on the restart file that is to be used for initializing the solution field.

maxmem (I) <120000> : specifies the maximum in-core memory to be used during the solution process (see Note 2).

The data cards associated with this command describe the solution parameters for the steady-state iterative algorithms and the time-dependent integration procedures.

Steady-State Solutions :

The data card required to access the solution methods for steady-state problems has the following form

STEADY, iterative method, *relax. factor, no. iter, iprint*, *
tol u, tol T, tol c_1, *tol* c_2

where,

iterative method (C) : indicate the type of solution procedure that is to be used. The permissable values include PICARD to invoke a first-order iterative method and NEWTON for a second-order Newton scheme (see Note 3).

relax. factor (R) <0.0> : specifies the relaxation factor that is applied to the solution vector during the iterative procedure (see Note 4).

no.iter (I) <10> : specifies the maximum number of iterations allowed during the solution process.

iprint (I) <1> : indicates when the solution field is printed during the iteration process; *i.e.*, every *iprint* iterations a printout of the solution field is produced (see Note 3).

tol u, tol T, *tol* c_i, (R) <1.0E-5> : specify the convergence tolerances for each variable that may be used to terminate the iteration process (see Note 5).

Transient Solutions :

The data card required to access the integration procedures for transient problems has the following form

TRANSIENT, integration method, *time step option, integration tol*, *
t_{init}, t_{final}, Δt, *no. steps, steady-state tol, iprint*

where,

integration method (C) : indicate the type of integration procedure that is to be used. The permissable values include EULER to invoke a first-order integration method and TRAPEZOID for a second-order integration scheme.

time step option (C) <FIXSTEP> : specifies the method for choosing the integration time step. If this parameter is omitted or set to FIXSTEP then the time step specified as Δt on this card is used throughout the integration interval. A parameter value of AUTOSTEP chooses the automatic time step selection process (see Note 6).

integration tol (R) <0.001> : specifies the integration tolerance that is to be used in selecting the time step. This parameter is omitted if the *time step option* is set to FIXSTEP.

t_{init}, t_{final} (R) : specify the time limits of the integration interval.

Δt (R) : specifies the time step for the integration procedure (see Note 6).

no.steps (I) <1000> : specifies the maximum number of time steps allowed during the integration interval.

steady-state tol, (R) <1.0E-5> : specifies the tolerance that is used to determine when a steady state solution has been reached (see Note 7).

iprint (I) <1> : indicates when the solution field is printed during the integration process; *i.e.*, every *iprint* time steps a printout of the solution field is produced.

Notes :

1) The use of an external file to provide an initial condition for the solution algorithm requires that logical unit 23 be attached to the NACHOS II job through the job control commands. This file must be formatted according to the instructions given in Section 3.17.

2) The dynamic dimensioning scheme in NACHOS II will allocate sufficient memory for the execution of a given solution procedure. However, the matrix solution algorithm functions most efficiently when a large block of in-core memory is available as a buffer area. The *maxmem* parameter allows the user to specify an upper bound on the total computer memory that is made available to the solution procedure. This parameter should be large enough to prevent the program from becoming I/O bound but not so large as to degrade job priority on the computer. User experience is the best guide to the setting of this parameter.

3) Multiple steady-state data cards may be used under a single **SOLVE** command to invoke changes to any of the parameters found on the card. Using combinations of the available iterative algorithms often provides a useful strategy for the solution of difficult nonlinear problems. Typically, the Picard scheme would be used first to get close to the solution; the Newton procedure would follow to provide a rapid convergence to the final result.

4) A relaxation method can be used with any of the iterative procedures. Under this option, the solution at the latest iteration cycle ($\mathbf{V}^{n+1}$) is defined by

$$\mathbf{V}^{n+1} = \alpha\mathbf{V}^{n} + (1-\alpha)\mathbf{V}^{*}$$

where $\mathbf{V}^{n}$ is the solution at the previous iteration, $\mathbf{V}^{*}$ is the last solution to the matrix problem and α is the relaxation factor. Further details on this procedure are given in [3].

5) The iteration process is terminated when the norm on the change in the variable between iterations falls below the specified tolerance. The tolerance on all variables in the problem must be met before convergence is declared and the **SOLVE** command terminated.

6) Automatic timestep selection cannot begin with the first timestep with the integration procedures available in NACHOS II. Under the AUTOSTEP option, the first several steps are computed using the fixed timestep specified by the Δt parameter. This initial timestep should be chosen conservatively to reduce the early time integration error and to prevent the automatic timestep selection process from drastically altering the timestep in order to meet the error criterion.

7) The control of the integration interval is accomplished by any of three methods. Integration will be terminated when the running time exceeds the specified final time (t_{final}) or when the maximum number of steps has been reached. Time integration will also be stopped when a steady state is reached as defined by the "*steady-state tol*" parameter. In this case the norms on the change in the solution between time steps must fall below the indicated tolerance.

3.9 STREAM Command Card

To aid in interpreting the solution obtained from the finite element model, NACHOS II allows the calculation of the stream function field as a user option. The computation of the stream function is activated through a command card of the form

STREAM, *psi base, output, no. timeplanes*

where,

psi base (R) <0.0> : specifies the value of the stream function at the first node of the first element processed (see Note 1).

output (C) <SUMMARY> : indicates if the stream function field is to be printed. If this parameter is omitted or set to SUMMARY, only a summary of the maximum and minimum stream function values are printed. A parameter value of PRINT causes the entire field to be printed for each timeplane in the analysis (see Note 2). A parameter value of NOPRINT suppresses all printout.

no. timeplanes (I or C) <ALL> : specifies the number of timeplanes to be included in the stream function computation. If this parameter is omitted or set to ALL, then all computed timeplanes are considered in the stream function computation. If this parameter is specified as an integer then all timeplanes with numbers less than or equal to the specified value are considered in the calculation (see Note 3).

Notes :

1) The calculation of the stream function is carried out by considering line integrals of the velocity along element boundaries. If the element ordering is not sufficiently continuous (*i.e.*, each successive element processed must have at least one corner node in common with a previously processed element), the entire stream function field cannot be computed. NACHOS II provides a diagnostic message in this situation. This problem can be corrected by changing the element ordering.

2) Printing of the stream function should be used with caution especially for time-dependent problems. The stream function field is printed element by element and can result in a very large amount of printout. The primary use of the **STREAM** command is to create a data file for subsequent plotting.

3) For a time-dependent analysis the computed timeplanes are numbered consecutively from 1 through n, beginning with the initial condition (*i.e.*, the solution at time = Δt is timeplane 2). A steady-state problem has only one timeplane.

3.10 FLUX Command Card

In the analysis of a given flow problem, quantities such as the fluid stresses and boundary heat flux are generally of interest. The computation of such flux quantities is initiated in NACHOS II by a command of the following form

FLUX, element location, *save data*, *mesh location*
.
.
flux type, timeplane no.
.
.
END

where,

element location (C) : specifies where in the finite element the fluxes are to be computed. The permissable values for this parameter are BOUNDARY and GAUSS (see Note 1).

save data (C) < > : specifies if the computed fluxes are to be saved for future plotting or post-processing. If this parameter is omitted the data is not saved. A parameter value of SAVE stores the data on a file for future use (see Note 2).

mesh location (C or I) <FULL> : specifies the elements in the mesh that are to have flux quantities computed. If this parameter is omitted or set to FULL, then the required fluxes are computed for all elements in the mesh. This parameter may also be set to a list of element numbers indicating which elements are to be considered in the flux computation. Up to fifty elements may be specified on any one **FLUX** command card.

flux type (C) : is the name of the type of flux quantity to be calculated. For flux type set to STRESS, the various components of the fluid stress tensor are computed. A value of HEATFLUX causes the heat fluxes from the energy equation to be computed. The fluxes associated with the auxiliary transport equations can be computed with a parameter value of AUXFLUX1 or AUXFLUX2. The computation of all fluxes in a particular problem can be requested with a parameter value of ALLFLUX (see Note 3).

timeplane no. (I) : specifies the timeplane number for which the flux computation is to be carried out. This parameter may be omitted for steady-state problems (see Note 4).

In order to ease the specification of flux data for transient analyses, a looping feature is available for use with the **FLUX** command. The looping command may be used with any of the "flux type" parameters (see Note 5) and has the following form

FLUXLOOP, no. loops, timeplane inc.
flux type, timeplane no.
FLUXEND
.
.
.
END

where,

no. loops (I) : specifies the number of flux computations to be made.

timeplane inc. (I) : indicates the frequency at which a flux computation is made *i.e.*, every "timeplane inc." time steps a flux evaluation is produced beginning with the "timeplane no." indicated on the flux type data card.

Notes :

1) Flux quantities may be computed at either of two locations within an element. Under the BOUNDARY option, the fluxes are evaluated on the element boundaries, midway between nodal points as shown in Figure 9a. This option is most useful when the fluxes are to be integrated along the boundary to produce global quantities, such as the forces on a body or energy transfer from a region. The second option, GAUSS, computes the fluxes at the integration points within the element (2×2 Gauss points in a quadrilateral element and 3 points in a triangular element) as shown in Figure 9b. These interior fluxes are subsequently extrapolated to the nodal points and averaged between adjacent elements to produce a continuous flux field over the domain. The GAUSS option automatically causes the fluxes to be computed for all elements in the mesh (sets the "mesh location" parameter to FULL) so that the nodal averaging procedure can be accomplished. This option is useful when the flux quantities are to be displayed graphically.

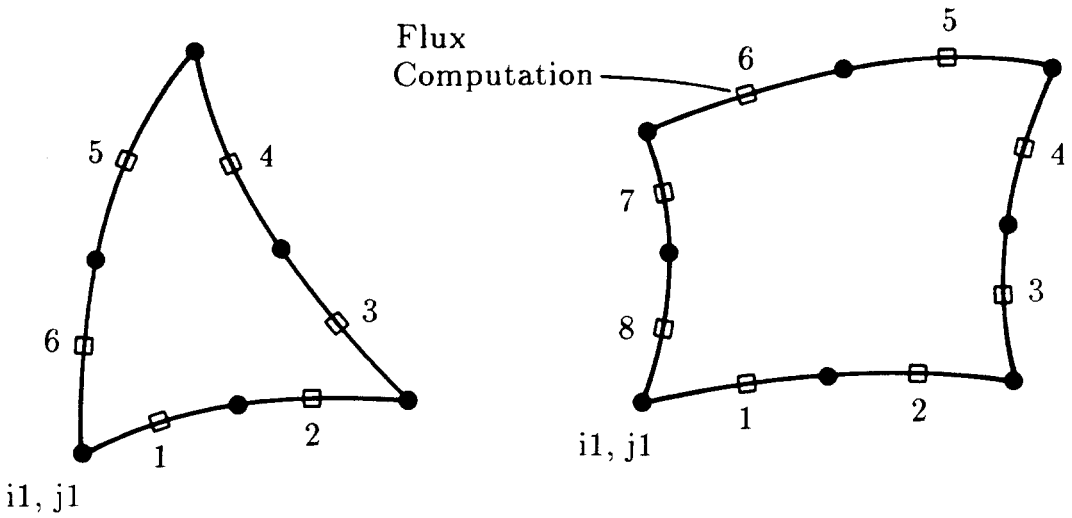
2) When the *save data* parameter is omitted, the flux computations are carried out under the specified options and the results are printed by NACHOS II. No flux

data is saved in this case. When *save data* is set to SAVE, the flux computations are carried out under a set of default options and the resulting fluxes are saved on a disk file for later use. No printed output is generated in this case. Under the SAVE option, fluxes are computed using the GAUSS specification for all elements in the mesh and for all timeplanes. The FLUXLOOP command is disabled under the SAVE option and should not be used.

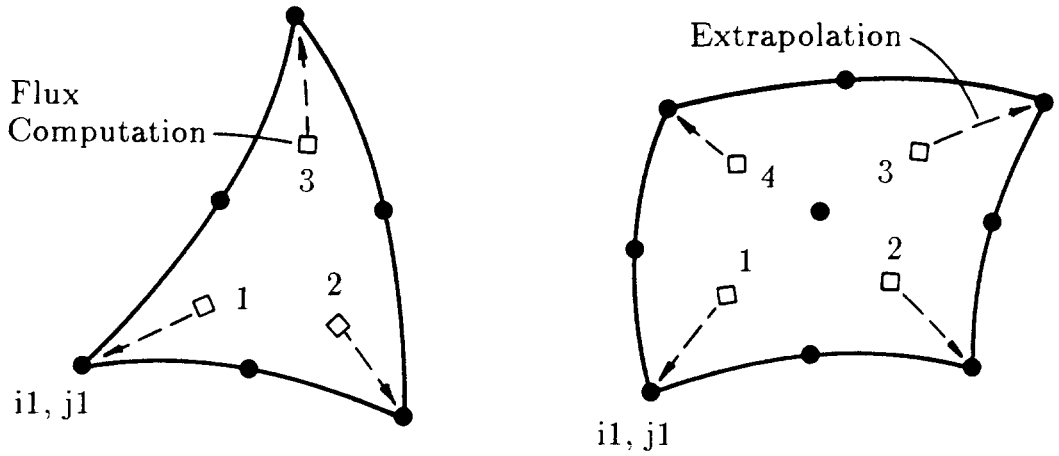
3) With a "flux type" setting of STRESS, the normal stresses (τ_{xx} and τ_{yy}), the shear stress (τ_{xy}) and when applicable, the hoop stress ($\tau_{\theta\theta}$) are computed for each of the specified elements. The values of the vorticity are also computed under this parameter setting. All fluid stresses are computed in terms of the global x, y coordinate system. The HEATFLUX option causes the x and y components of the heat flux vector to be computed; with the BOUNDARY option the heat flux normal to the element boundary and the total energy through each element side is also reported. The AUXFLUX options produce results analogous to the HEATFLUX option.

4) For a time-dependent analysis the computed timeplanes are numbered consecutively from 1 through n, beginning with the initial condition (*i.e.*, the solution at time = Δt is timeplane 2). A steady-state problem has only one timeplane.

5) Each FLUXLOOP command may only contain one "flux type" data card. Multiple FLUXLOOP commands may be used under a single **FLUX** command; FLUXLOOP and individual "flux type" data cards may be mixed under the same **FLUX** command. The looping feature is not used with the SAVE option.



a) Boundary Flux Option



b) Gauss Flux Option

Figure 9: Flux evaluation points within an element.

3.11 PLOT Command Cards

The NACHOS II program contains a batch plotting package to facilitate the interpretation of data obtained from a solution and to aid in setting up element meshes. There are seven basic types of plots which are available from NACHOS II. The command to access the plotting package is of the following form

PLOT, plot type, xmin, ymin, xmax, ymax, *xscale, yscale*

where,

plot type (C) : is a name which specifies the type of plot that is to be constructed. The permissible parameter values are cataloged below.

xmin, ymin, xmax, ymax (R) : specify the range of the coordinates for the plot. For line plots (*e.g.*, element plots, outline plots, contour plots, *etc.*) the xmin, ... ymax parameters define a rectangular window in the global coordinate system; only elements and their associated data that fall entirely within the window will be plotted. For graphs (*e.g.*, time histories or profiles), the xmin, ... ymax parameters may be used to set the maximum and minimum values for the abscissa and ordinate of the plot, respectively. If these parameters are omitted, the axes for the graph are set by NACHOS II.

xscale, yscale (R) <1.0> : specify magnification factors for the x and y coordinates of a line plot. The default values produce a correctly proportioned plot of the largest possible size, consistent with the plotting device (see Note 1).

The permissable types of plots as specified by the "plot type" parameter include the following :

(a) POINTS – generates a plot of the grid points generated by the **MESH** command.

(b) ELEMENTS – generates a plot of the finite element mesh.

(c) OUTLINE – generates an outline plot of the problem domain with the material boundaries indicated.

(d) CONTOUR – generates contour plots of the stream function, pressure, temperature, fluid stresses, vorticity or auxiliary variables.

(e) VECTOR – generates a plot of the velocity, heat flux or auxiliary flux vectors.

(f) HISTORY – generates time history plots of any of the dependent variables.

(g) PROFILE – generates plots of the dependent variables versus spatial location with time as a parameter.

Following some of the **PLOT** command cards a series of data cards is required to indicate the values to be plotted, timeplanes to be considered, *etc.* The list of data cards for each of these **PLOT** commands is terminated with an **END** card. The **PLOT** commands for the POINTS, ELEMENTS and OUTLINE options do not require any further data and therefore do *not* require the use of an **END** card.

Contour Data Cards :

For the CONTOUR plot option, the required data cards are of the following form

contour type, timeplane no., no. contours, *max contour, min contour*,*
c1, c2, ..., c25
.
.
.
END

where,

contour type (C) : is the name indicating the variable to be contoured. This parameter may have any of the following values : TEMP (temperature), PRESS (pressure), AUXVAR1, AUXVAR2 (auxiliary variables), STREAM (stream function), TAUX, TAUY, TAUXY, TAUTHETA (x, y, xy and θ components of the stress), and VORTICITY (vorticity) (see Note 6).

timeplane no. (I) : is the number of the timeplane for which a plot is required. For steady-state problems this parameter may be omitted (see Note 2).

no. contours (I) : specifies the number of contours to be plotted. A maximum of 25 contour lines is allowed on each plot.

max contour, min contour (R) <0.0> : specify the optional maximum and minimum values of the variable to be used in computing the contour values (see Note 3).

c1, c2, ..., c25 (R) <0.0> : are optional values that specify the values of the contours to be plotted (see Note 3).

Any number and/or type of contour data cards may follow a **PLOT**, CONTOUR command; the sequence is terminated by an **END** card. To simplify the plotting of a series of contours for a time-dependent problem, a looping feature is available. The looping command may be used with any contour type (see Note 4) and has the following form

PLOTLOOP, no. plots, timeplane inc.
contour type, timeplane no., no. contours, ...
PLOTEND
.
.
.
END

where,

no. plots (I) : specifies the number of plots to be generated.

timeplane inc. (I) : indicates the frequency at which a plot is generated, *i.e.*, every "timeplane inc." time steps a contour plot is produced beginning with the "timeplane no." indicated on the contour data card (see Note 2).

Vector Data Cards :

For the VECTOR plot option, the required data cards are of the following form

vector type, timeplane no., *vectorscale, vectormax, vectormin*
.
.
.
END

where,

vector type (C) : is the name indicating the variable to be plotted as a vector field. This parameter may have any of the following values : VELOCITY, HEATFLUX, AUXFLUX1 and AUXFLUX2.

timeplane no. (I) : is the number of the timeplane for which a plot is required. For steady-state problems this parameter may be omitted (see Note 2).

vectorscale (R) <1.0> : specifies the scaling factor to be used in plotting the vectors (see Note 5).

vectormax (R) <max vector> : specifies the maximum value of the vectors to be included in the plot. Any vector with a magnitude greater than *vectormax* is not plotted. The default is the largest vector found in the field, *i.e.*, all vectors are plotted.

vectormin (R) <0.0> : specifies the minimum value of the vectors to be included in the plot. Any vector with a magnitude less than *vectormin* is not plotted.

Any number and/or type of vector data cards may follow a **PLOT**, VECTOR command; the sequence is terminated by an **END** card. To simplify the plotting of a series of vector plots for a time-dependent problem, a looping feature is available. The looping command may be used with any vector type (see Note 4) and has the following form

PLOTLOOP, no. plots, timeplane inc.
vector type, timeplane no., ...
PLOTEND
.
.
.
END

where,

no. plots (I) : specifies the number of plots to be generated.

timeplane inc. (I) : indicates the frequency at which a plot is generated, *i.e.*, every "timeplane inc." time steps a vector plot is produced beginning with the "timeplane no." indicated on the vector data card (see Note 2).

History Data Cards :

For the HISTORY plot option, the required data cards are of the following form

plot variable, no. points, timeplane1, timeplane2, e1, n1, e2, n2, ..., e10, n10
.
.
.
END

where,

plot variable (C) : indicates the variable that is to be plotted as a function of time. The permissable values for this parameter include : TEMP (temperature), PRESS (pressure), AUXVAR1, AUXVAR2 (auxiliary variables), UVEL (x velocity component), VVEL (y velocity component), TAUX, TAUY, TAUXY, TAUTHETA (x, y, xy and θ stress components), VORTICITY (vorticity), TIMESTEP (integration time step), TEMP+, PRESS+, AUXVAR1+, AUXVAR2+, UVEL+ and VVEL+ (special point values) (see Note 6).

no. points (I) : specifies the number of time histories to be plotted. A maximum of ten time histories per plot is allowed.

timeplane1, timeplane2 (I) : indicate the timeplane numbers at which the time histories are to begin and end, respectively. The maximum difference allowed between timeplane2 and timeplane1 is 1000 *i.e.*, only 1000 time steps may be represented on a given plot.

e1, n1, e2, n2, ..., e10, n10 (I) : are pairs of numbers indicating the element and node number for which a time history is required. A maximum of ten such pairs per data card (*i.e.*, per plot) is allowed. When plotting histories for special output points (*i.e.*, plot variables with a + suffix), pairs of numbers are not required. In this latter case, the special point numbers are simply listed in sequence.

There is no restriction on the number of data cards that may follow a **PLOT**, HISTORY command card: the sequence is terminated by an **END** card.

Profile Data Cards :

For the PROFILE plot option, the required data cards occur in pairs and are of the following form

TIMEPLANE, no. timeplanes, t1, t2, ..., t10
plot variable, profile spec., no. points, e1, s/n1, e2, s/n2, ..., e50, s/n50
.
.
.
END

where,

no. timeplanes (I) : specifies the number of timeplanes at which the profile is to be plotted. A maximum of ten profiles per plot is allowed.

t1, t2, ..., t10 (I) : specify the timeplane numbers at which the profile is to be plotted. For steady-state problems this list is omitted (see Note 2).

plot variable (C) : indicates the variable that is to be plotted as a function of spatial location. The permissable values for this parameter include : TEMP (temperature), PRESS (pressure), AUXVAR1, AUXVAR2 (auxiliary variables), UVEL (x velocity component), VVEL (y velocity component), SIGMAX, SIGMAY, SIGMAXY, SIGMATHETA (x, y, xy and θ stress components) and VORTICITY (vorticity) (see Note 6).

profile spec. (C) : indicates how the profile geometry is specified. If this parameter is set to SIDES, the profile geometry is specified in terms of element sides. A parameter value of NODES indicates the profile is given by a list of individual nodal points (see Note 7).

no. points (I) : specifies the number of data pairs needed to describe the profile geometry. A maximum of fifty such data pairs is permitted.

e1, s/n1, e2, s/n2, ..., e50, s/n50 (I) : are pairs of numbers specifying the profile geometry in terms of an element number and a side or node number (see Note 7).

There is no restriction on the number of pairs of data cards that may follow a **PLOT**, PROFILE command card: the sequence is terminated by an **END** card.

Notes :

1) The use of the *xscale* and *yscale* parameters produce a plot that is stretched in one of the coordinate directions with unequal axis scaling. Only one of the two scale

factors should be specified for any given plot. The scaled coordinate can only be expanded to the point where it is equal in length to the unscaled dimension, *i.e.*, the maximum allowed scaling will produce a square plot.

2) For a time-dependent analysis the computed timeplanes are numbered consecutively from 1 through n, beginning with the initial condition (*i.e.*, the solution at time = Δt is timeplane 2). A steady-state problem has only one timeplane.

3) The values of the contours to be plotted may be specified by any of three methods. Contour values can be generated automatically at equally spaced intervals by omitting all of the optional parameters on the contour data card, (*i.e.*, *max contour, min contour* and *c1, c2, ...*). In this case the contours are uniformly spaced between the maximum and minimum field values found in the computational domain. The maximum and minimum values used to compute the contour values can also be input by the user by specifying the *max contour* and *min contour* parameters. The contours are also evenly spaced under this option. Individual contour values can be specified through the *c1, c2, ...* parameters. Use of this last option requires that the *max contour* and *min contour* parameters be left unspecified on the data card.

4) There is no limit on the number of PLOTLOOP data sets that may follow a given **PLOT** command; each PLOTLOOP must be closed with a PLOTEND statement. Within any given PLOTLOOP only a single type of data card may be defined. PLOTLOOP data sets and individual data cards may be mixed under a single **PLOT** command.

5) The VECTOR plot option draws an arrow at each corner node of an element, the length of which is proportional to the magnitude of the vector quantity. The *vectorscale* parameter allows the vector magnitudes to be scaled to the appropriate dimensions for the plot. Generally, this parameter will be chosen such that the longest vector in the field will be drawn as some fraction of the axis length on the plot. Thus, for example, if the largest vector in the field is 100.0 units and the plot axis is on the order of 10 units, the *vectorscale* parameter should be on the order of 0.01 for the largest vector to be one tenth of the axis length.

6) Some of the variables to be displayed by the various **PLOT** commands are not directly available from the solution file and must have been generated by special commands prior to plotting. The display of the fluid stresses and vorticity requires that the appropriate **FLUX** command have been previously executed with the SAVE option (see Section 3.10). The **STREAM** command must have been executed prior to the request for contour plots of the stream function. Time histories of variables defined at the special output points require that the special points option have been invoked under the **OUTPUT** command prior to obtaining the solution (see Section 3.7).

7) The geometry for a PROFILE plot is determined by the sequence of nodal points described on the data card. When the "profile spec." parameter is set to SIDES, the profile is taken along element boundaries with the spatial distance measured along straight line segments joining adjacent nodes in the elements. The parameter value of NODES causes the profile to be constructed along straight line segments joining successive nodes without regard to element sides. There is no requirement that the profile have a continuous path, *i.e.*, some elements along a path may be omitted. Also, multiple side/node specifications for an individual element are permitted. When the SIDES option is used, the profile path is directed along element sides with the direction of increasing path length determined by increasing (local) corner node number. Thus, for an eight-node quadrilateral with side 1 specified, the path proceeds from nodes 1 to 5 to 2; with side 3 specified, the path is directed from nodes 3 to 7 to 4 (see Figure 8 for node numbering). If the path direction is inappropriate with this option, the NODES description should be used.

3.12 PLOTSET Command Card

The specification of various formatting options for the batch plotting routines is controlled by the **PLOTSET** command. This command can be used with any of the **PLOT** commands described previously (see Note 1) and has the following form

PLOTSET, *number, special pts., nodes, axes, grid, symbol, font*

where,

number (C) <NONUMBER> : specifies if element numbers are to be placed on an element mesh plot. A parameter setting of NUMBER forces element numbers to be plotted; the omission of this parameter or a setting of NONUMBER causes element numbers to be omitted from the plot.

special pts. (C) <NOSPECIAL> : specifies if the location of any special output points are to located on an element mesh plot. A parameter setting of SPECIAL causes special point locations to be indicated on element mesh plots. If this parameter is omitted or set to NOSPECIAL, then no special point locations are plotted.

nodes (C) <NONODES> : specifies if nodal point locations are to be placed on outline, contour and vector plots. A parameter setting of NODES causes the nodal point locations to be indicated on certain types of plots. If this parameter is omitted or set to NONODES, nodal point locations are not indicated.

axes (C) <AXES> : indicates if coordinate axes are to be plotted. A parameter value of AXES causes an axis system to be placed on a plot: a value of NOAXES suppresses the plotting of coordinate axes.

grid (C) <GRID> : indicates if grid lines are to be included on graph type plots. A parameter value of GRID causes grid lines to be included on the plot; a value of NOGRID suppresses the grid lines.

symbol (C) <SYMBOL> : specifies if symbols are to be used in defining the curves on a graph type plot. The omission of this parameter or a parameter setting of SYMBOL causes symbols to be used to define the curves; a parameter value of NOSYMBOL causes the curves to be defined by lines only.

font (I) <2> : specifies the type of font used for the plot. The types of fonts supported are system dependent; up to five different fonts could be used (see Note 2).

Notes :

1) All of the formatting options specified by the **PLOTSET** command are intialized to their default values at the start of each job. When formatting options are changed by the **PLOTSET** command they remain changed until altered by another **PLOTSET** command. **PLOTSET** commands may appear at any reasonable point in the input file.

2) The types of fonts currently installed in NACHOS II correspond to the following:

   - $font = 1$ Simplex
   - $font = 2$ Small Complex
   - $font = 3$ Duplex
   - $font = 4$ Futura (Shaded)
   - $font = 5$ Swiss Medium (Shaded)

   The designations given above correspond to the use of the DISSPLA graphics package [5].

3.13 POST Command Card

A post-processing data file can be obtained from NACHOS II for use with various stand-alone graphics programs. This option is invoked by the **POST** command. The data associated with this command allows the NACHOS II solution file to be edited such that only selected variables and/or timeplanes are written to the post-processing file. This command and its associated data have the following form

POST
data type, no. var, name1, name2, ..., namen
.
.
.
timeplane data card
END

where,

data type (C) : indicates the type of data that will be written to the post-processing file. The permissable types of data are denoted by NODES, ELEMENTS, HISTORY and GLOBAL. Only one data card of each type is permitted; not all data types need be included (see Note 1).

no. var (I) : specifies the number of different variables that are included under each data type.

name1, name2, ..., namen (C) : specify the names of the variables included under each data type. For the data type labeled NODES the following variables are allowed: UVEL (x velocity component), VVEL (y velocity component) PRESS (pressure), TEMP (temperature), AUXVAR1, AUXVAR2 (auxiliary variables), STREAM (stream function), TAUX, TAUY, TAUXY, TAUT (x, y, xy and θ stress components) and VORT (vorticity). The flux components QX, QY, (heat fluxes) V1X, V1Y, V2X and V2Y (auxiliary variable fluxes) may also be specified (see Note 2).

The timeplane data card associated with the **POST** command, has one of three forms depending on how the timeplane data is specified. If all of the timeplanes computed by NACHOS II are to be included in the post-processing file then the timeplane data card is of the following form

TIMEPLANE, ALL

The above form should be used for steady-state problems. All of the computed timeplanes need not be included in the post-processing file. If a regular pattern of timeplanes are desired in the graphics file then a timeplane data card of the following form may be used

TIMEPLANE, INCREMENT, timeplane1, timeplane2, inc

where,

timeplane1, timeplane2 (I) : specify the starting and ending timeplane numbers that are to be included in the post-processing file (see Note 3).

inc (I) : specifies the increment in the timeplane number to be used in editing the solution file. Beginning with timeplane1, every "inc" timeplane will be included in the post-processing file.

Specific timeplanes from the solution file can be included in the post-processing file by use of the third form of the timeplane card

TIMEPLANE, SPECIFIED, no. timeplanes, t1, t2, ..., tn

where,

no. timeplanes (I) : specifies the number of timeplanes to be written to the post-processing file.

t1, t2, ..., tn (I) : specify the timeplane numbers to be used in editing the solution file. A maximum of 50 timeplanes can be specified under this option (see Note 3).

Notes :

1) At the present time only data corresponding to the NODES parameter is written to the post-processing file. The ELEMENTS, HISTORY and GLOBAL data types have been included for future use or for use with a modified version of the code.

2) Some of the variables that can be written to the post-processing file are not part of the basic solution file. If these variables are to be included in the graphics file, the appropriate commands must have been executed prior to the **POST** command. To include the fluid stresses and/or vorticity, the **FLUX** command must have been executed with the SAVE option specified (see Section 3.10). Inclusion of heat fluxes or fluxes of the auxiliary variables in the post-processing file, also requires that the **FLUX** command have been previously executed with the SAVE option specified. The inclusion of the stream function requires that the **STREAM** command have been previously executed.

3) For a time-dependent analysis the computed timeplanes are numbered consecutively from 1 through n, beginning with the initial condition (*i.e.*, the solution at time = Δt is timeplane 2). A steady-state problem has only one timeplane.

3.14 Termination Command Cards

The termination command cards are of two types. The first type has already been encountered and is used to signal the end of data for a particular command card. This termination card has the form

END

The **END** command is used in conjunction with *all* commands in NACHOS II, *except* for the following: Title and Comments, **FORMKF, STREAM**, and the **PLOT** commands for POINTS, ELEMENTS, and OUTLINE plots.

The termination of all data input to NACHOS II is signaled by the **STOP** command card which is of the form

STOP

This must be the last command in the input file in order to ensure proper termination of the program.

3.15 Input File Structure

The order of the commands to NACHOS II is primarily dependent on the needs of the user. However, some limitations on the command sequence are obvious as some operations are necessary prerequisites to other computations. The following comments provide some guidelines to specific sequencing situations.

(a) The title and comment cards must always be the first entries in the input file.

(b) The **ELEMENTS** command must follow the **MESH** command.

(c) The **PLOT**, POINTS command must be located immediately after the the **MESH** command since the grid point coordinates are not retained during later operations. Typically, a grid point plot is used in setting up a mesh and is not generated during a complete solution sequence.

(d) The **PLOT**, ELEMENTS and OUTLINE commands can be located anywhere after the **ELEMENTS** command.

(e) The remaining **PLOT** commands can occur at any point after the values to be plotted have been computed.

(f) The **OUTPUT** commands must precede the **SOLVE** command.

(g) The **POST** command should normally be the last operation in the input file.

At the beginning of each execution, NACHOS II previews all of the data in the input file and attempts to verify that the requested commands are in an acceptable order. However, not all possibilities can be checked; the user is cautioned to observe the general ordering of commands shown in the input guide.

3.16 User Supplied Subroutines

There are several instances which require the user to supply FORTRAN coded subroutines to NACHOS II. The occurrence of variable material properties or source terms, the definition of certain types of variable boundary conditions, variations in the gravitational vector and the input of special types of mesh distributions all result in the need for one or more user supplied subroutines. The required formats for these subroutines will be described in the following sections.

3.16.1 Mesh Generation

When the EXTDEF data card is encountered in the **MESH** command (Section 3.4) NACHOS II expects a user subroutine (USRMSH) to be supplied. This subroutine allows mesh point locations to be defined by the user in any appropriate manner. The required subroutine must have the following form

```
      SUBROUTINE USRMSH (X, Y, MAXI, MAXJ)
      DIMENSION X(MAXI,*), Y(MAXI,*)
C
C     FORTRAN coding to generate an array of
C      nodal point locations
C
      RETURN
      END
```

where the variables in the subroutine parameter lists are

`X(MAXI, MAXJ)` : a two-dimensional array containing the x coordinates of the mesh points (output).

`Y(MAXI, MAXJ)` : a two-dimensional array containing the y coordinates of the mesh points (output).

`MAXI, MAXJ` : integers specifying the largest I and J values that can be defined in the mesh. These parameters are set on the **MESH** command card (input).

3.16.2 Material Property Subroutines

The variation of material properties with one or more of the dependent variables, requires that a series of material property subroutines be supplied to NACHOS II. When

the "variable properties" parameter on the material data card (Section 3.3) is set to VARIABLE for any material in the problem, subroutines describing the material viscosity (USRVIS), thermal conductivity (USRCON), and volumetric expansion coefficient (USREXT) are generally expected by NACHOS II. For isothermal problems only the viscosity subroutine is required. Forced convection problems must include the thermal conductivity subroutine while free or mixed convection problems also require the volumetric expansion subroutine. In the event that not all of the required material properties are in fact variable, the remaining properties must still be set through the use of the appropriate subroutine, *i.e.*, NACHOS II expects all the subroutines to be present for a particular type of problem. Each subroutine is used to evaluate the appropriate material property for *all* materials labeled as VARIABLE on the material data card.

The required subroutines must have the following forms

```
      SUBROUTINE USRVIS (AMU, T, S, VAR1, VAR2, NNODES, MAT, ICYCLE)
      DIMENSION AMU(*), T(*), S(*), VAR1(*), VAR2(*)
C
C     FORTRAN coding to evaluate the viscosity for each material
C
      RETURN
      END


      SUBROUTINE USRCON (AK, T, S, VAR1, VAR2, NNODES, MAT)
      DIMENSION AK(*), T(*), S(*), VAR1(*), VAR2(*)
C
C     FORTRAN coding to evaluate the thermal conductivity
C      for each material
C
      RETURN
      END


      SUBROUTINE USREXT (BETA, T, VAR1, VAR2, NNODES, MAT)
      DIMENSION BETA(*), T(*), VAR1(*), VAR2(*)
C
C     FORTRAN coding to evaluate the volumetric thermal
C      expansion coefficient for each material
C
      RETURN
      END
```

where the variables in the subroutine parameter lists are

AMU(NNODES) : an array containing the values of the fluid viscosity (output).

AK(NNODES) : an array containing the values of the material thermal conductivity (output).

BETA(NNODES) : an array containing the values of the fluid volumetric thermal expansion coefficient (output).

T(NNODES) : an array containing nodal point values of the temperature (input).

S(NNODES) : an array containing nodal point values of the second invariant of the rate-of-deformation tensor (input).

VAR1(NNODES) : an array containing nodal point values of the first auxiliary variable (input).

VAR2(NNODES) : an array containing nodal point values of the second auxiliary variable (input).

NNODES : an integer specifying the number of nodes at which the material property is to be evaluated (input).

MAT : an integer specifying the material number as set on the material data card (input).

ICYCLE : an integer specifying the number of the current iteration cycle (input).

The above subroutines are also used to evaluate the variable properties for a fluid-saturated porous layer. When the "variable property" parameter on the porous material data card is set to VARIABLE, the above subroutines are used to evaluate the Brinkman viscosity, the effective thermal conductivity and the thermal expansion coefficient, respectively.

When the auxiliary transport equations are included in the problem formulation several additional subroutines are required under the variable properties option. If the basic properties for a material have been indicated as being variable, then the coefficients for the auxiliary equations for that material are also assumed to be variable. Under these conditions the following subroutines are required to evaluate the diffusion coefficients (USRDIF) and volumetric expansion coefficients (USREX) for the auxiliary equations.

```
      SUBROUTINE USRDIF (AD, T, VAR1, VAR2, NNODES, MAT, IEQN)
      DIMENSION AD(*), T(*), VAR1(*), VAR2(*)
C
C     FORTRAN coding to evaluate the diffusion coefficient
C      for each material (auxiliary equation)
C
```

```
      RETURN
      END


      SUBROUTINE USREX (ZETA, T, VAR1, VAR2, NNODES, MAT, IEQN)
      DIMENSION ZETA(*), T(*), VAR1(*), VAR2(*)
C
C     FORTRAN coding to evaluate the volumetric expansion
C      coefficient for each material (auxiliary equation)
C
      RETURN
      END
```

where the variables in the subroutine parameter lists are

`AD(NNODES)` : an array containing the values of the diffusion coefficient for the auxiliary equation (output).

`ZETA(NNODES)` : an array containing the values of the fluid volumetric expansion coefficient due to the auxiliary variable (output).

`T(NNODES)` : an array containing nodal point values of the temperature (input).

`VAR1(NNODES)` : an array containing nodal point values of the first auxiliary variable (input).

`VAR2(NNODES)` : an array containing nodal point values of the second auxiliary variable (input).

`NNODES` : an integer specifying the number of nodes at which the material property is to be evaluated (input).

`MAT` : an integer specifying the material number as set on the material data card (input).

`IEQN` : an integer specifying which of the two auxiliary equations is being evaluated (input).

3.16.3 Volumetric Sources

When the volumetric heating parameter, Q, on a material property data card (Section 3.3) is set equal to VARIABLE, then NACHOS II expects a source subroutine (USRVHS) to be supplied by the user. Also, when either of the source parameters, Q_i, on the auxiliary property data card is set to VARIABLE then a second source subroutine (USRVS) must be supplied to NACHOS II. These subroutines must have the following forms

```
      SUBROUTINE USRVHS (QVALUE, T, VAR1, VAR2, X, Y, NNODES, MAT, NEL,
     1                   TIME, ICYCLE)
      DIMENSION QVALUE(*), T(*), VAR1(*), VAR2(*), X(*), Y(*)
C
C     FORTRAN coding to evaluate the volumetric heat source
C      for each material
C
      RETURN
      END


      SUBROUTINE USRVS (QVALUE, T, VAR1, VAR2, X, Y, NNODES, MAT, NEL,
     1                  TIME, ICYCLE, IEQN)
      DIMENSION QVALUE(*), T(*), VAR1(*), VAR2(*), X(*), Y(*)
C
C     FORTRAN coding to evaluate the volumetric source
C      for each auxiliary equation
C
      RETURN
      END
```

where the variables in the subroutine parameter list are

`QVALUE(NNODES)` : an array containing the values of the volumetric source (output).

`T(NNODES)` : an array containing nodal point values of the temperature (input).

`VAR1(NNODES)` : an array containing nodal point values of the first auxiliary variable (input).

`VAR2(NNODES)` : an array containing nodal point values of the second auxiliary variable (input).

`X(NNODES), Y(NNODES)` : arrays containing nodal point values of the x and y coordinates (input).

`NNODES` : an integer specifying the number of nodes at which the material property is to be evaluated (input).

`MAT` : an integer specifying the material number as set on the material data card (input).

`NEL` : an integer specifying the number of the current element (input).

`TIME` : the value of the current time (input).

ICYCLE : an integer specifying the number of the current iteration cycle (input).

IEQN : an integer specifying which of the two auxiliary equations is being evaluated (input).

3.16.4 Gravity Vector

The orientation of the gravity vector for use in the body force terms of the momentum equation is allowed to vary over the problem domain. If the g_x parameter on a material property card (Section 3.3) is set to VARIABLE, then NACHOS II expects a subroutine (USRGRV) describing the orientation and magnitude of gravity to be supplied by the user. This subroutine must have the following form

```
      SUBROUTINE USRGRV (GX, GY, X, Y, NNODES, NEL)
      DIMENSION X(*), Y(*)
C
C     FORTRAN coding to evaluate the orientation and magnitude
C      of the gravity vector
C
      RETURN
      END
```

where the variables in the subroutine parameter list are

GX : the value of the x component of the gravity vector (output).

GY : the value of the y component of the gravity vector (output).

X(NNODES), Y(NNODES) : arrays containing nodal point values of the x and y coordinates (input).

NNODES : an integer specifying the number of nodes in the element (input).

NEL : an integer specifying the number of the current element (input).

3.16.5 Boundary Conditions

Boundary conditions that vary with time require the use of subroutines (USRBCn) that describe the appropriate time history of the dependent variable or its derivative (flux). These subroutines can also be used to change boundary conditions as a function of the

iteration cycle or to allow some types of boundary conditions to be functions of the dependent variables and/or spatial location. Dependent variables can vary with any of these parameters; for consistency with the finite element formulation, fluxes should be uniform along each element boundary. The correspondence between a particular boundary condition and its variation with time is established through the "function number" parameter, which appears on the BC data card (Section 3.5); this number also appears in the subroutine name that defines the time history. The subroutines for the specification of variable boundary conditions must be of the following form

```
      SUBROUTINE USRBCn (VALUE, UBD, VBD, PBD, TBD, V1BD, V2BD,
     1                   XBD, YBD, NNODES, NEL, TIME, ICYCLE)
      DIMENSION VALUE(*), UBD(*), VBD(*), PBD(*), TBD(*)
      DIMENSION V1BD(*), V2BD(*), XBD(*), YBD(*)
C
C     FORTRAN coding to evaluate a boundary condition
C      as a function of time
C
      RETURN
      END
```

where the variables in the subroutine parameter list are

`n` : an integer specifying the "function number" as set on the BC data card ($1 \leq n \leq 6$).

`VALUE(NNODES)` : an array containing nodal point values of the boundary condition (output).

`UBD(NNODES)` : an array containing nodal point values of the x component of velocity (input).

`VBD(NNODES)` : an array containing nodal point values of the y component of velocity (input).

`PBD(NNODES)` : an array containing nodal point values of the pressure (input).

`TBD(NNODES)` : an array containing nodal point values of the temperature (input).

`V1BD(NNODES)` : an array containing nodal point values of the first auxiliary variable (input).

`V2BD(NNODES)` : an array containing nodal point values of the second auxiliary variable (input).

XBD(NNODES), YBD(NNODES) : arrays containing nodal point values of the x and y coordinates (input).

NNODES : an integer specifying the number of the nodes at which the boundary condition is to be evaluated (input).

NEL : an integer specifying the number of the current element (input).

TIME : the value of the current time (input).

ICYCLE : an integer specifying the number of the current iteration cycle (input).

The use of a generalized convection/radiation boundary condition requires that the variation of the heat transfer coefficient be specified as a function of temperature, spatial location and/or time. When the h parameter on a SET data card (Section 3.5) is specified as VARIABLE, then NACHOS II expects a user subroutine (USRHTC) describing the heat transfer coefficient. This subroutine must be of the following form

```
      SUBROUTINE USRHTC (HT, TSURF, TREF, XSURF, YSURF, TIME, NEL, ISET)
C
C     FORTRAN coding to evaluate the heat transfer coefficient
C      for each set number
C
      RETURN
      END
```

where the variables in the subroutine parameter list are

HT : the value of the heat transfer coefficient (output).

TSURF : the local surface temperature (input).

TREF : the reference temperature of the environment as specified on the SET data card (input).

XSURF, YSURF : the x, y coordinates on the surface (input).

TIME the value of the current time (input).

NEL : an integer specifying the number of the current element (input).

ISET : an integer specifying the "set no." as shown on the BC and SET data cards (input).

Note that if more than one generalized convection/radiation boundary condition occurs in the problem, the above subroutine is used for the evaluation of all such variable heat transfer coefficients. The "set no."parameter (`ISET`) is used to distinguish the different boundary conditions.

3.17 Initial Conditions and Restarts

The analysis of a transient flow problem requires the specification of a set of initial conditions for the all of the dependent variables. For cases where the fluid is initially quiescent and the temperature (and auxiliary variables) may be assumed uniform over each material, the initial conditions may be set through the parameters available on the material data cards (Section 3.3). For problems in which the initial fields are more complex, NACHOS II allows the initial conditions to be input from an external file. In order to be compatible with NACHOS II, the initial conditions must have been written with the following unformatted FORTRAN write statement:

```
WRITE (N) TIME, NUMNOD, NUMVAR, ((SOLN(I,J), I=1,NUMNOD),
          J=1,NUMVAR)
```

where,

`N` : is the logical unit number of the file on which the data is written

`TIME` : is the initial time.

`NUMNOD` : is the number of nodes in the the mesh.

`NUMVAR` : is the number of dependent variables in the problem.

`SOLN(I,J)` : is the two dimensional array containing the dependent variables. The `I` index indicates the node number and the `J` index indicates the particular dependent variable. The dependent variables are listed `1` through `NUMVAR` in the following order: x velocity component, y velocity component, pressure, temperature, auxiliary variable 1, auxiliary variable 2.

When the file containing the initial conditions is attached to a NACHOS II job, the file name must be made to correspond to logical unit 23 (see Section 4.4).

The output file containing the solutions generated by NACHOS II has the same format as described above. Therefore, solutions obtained from NACHOS II may be used directly as initial conditions for subsequent problems. The solution output file from NACHOS II is logical unit 19 (see Section 4.4).

The fact that the input and output files for the solution share a common format means that the code can be easily restarted from a particular point in a previous solution. Such a procedure consists of attaching a previous solution file (unit 23), re-executing the input file and specifying on the **SOLVE** command the timeplane at which the solution is to be restarted. The input file must be re-executed since there is no explicit provision

made for storing all of the data from a previous run. Typically, the computation time for recomputing the mesh, element and boundary condition data and the element matrices is small enough not to warrant the saving of this information for a possible restart operation. Note that when restarts are carried out using the above procedure, the resulting solution file (unit 19) contains the entire time history of the problem. The contents of unit 23 are copied to unit 19 up to and including the restart timeplane; the newly computed timeplanes are added to unit 19 during the NACHOS II execution.

3.18 Error Messages

NACHOS II has been constructed with a number of error checks and tests for bad or inconsistent data, storage problems, *etc.* When an error is encountered, an error message and other pertinent data are printed and the program is terminated with a STOP ERROR. The error citations contain sufficient information to allow the problem to be diagnosed and corrected. For each listed error condition, the subroutine encountering the error is given along with an explanatory message. Also, any relevant integers and/or character data that help to explain the problem are output by the code.

The general format for error messages takes the following form,

```
ERROR FOUND IN - ‘‘SUBROUTINE NAME’’

DESCRIPTION - ‘‘ERROR MESSAGE’’

RELEVANT PARAMETERS -

     LABEL 1 = ‘‘INTEGER DATA’’

     LABEL 2 = ‘‘INTEGER DATA’’

     LABEL 3 = ‘‘CHARACTER DATA’’
```

4 Code Installation and Access

The use of a program such as NACHOS II generally requires some knowledge of how the program is installed and operates on a specific computer system. In particular, details regarding the use of system and utility libraries, file usage, file formats, FORTRAN coding standards and access to the program are important to the successful and efficient use of the code. These topics become especially critical when the code is used for the analysis of very complex problems, when the code is modified for a new application or when the program is transported to a new computer system. In the following sections, all of the above topics are discussed in some detail. The main emphasis will be on the operation of NACHOS II on the Sandia National Laboratories computer systems; the operation of the code on other computers should involve only minor alterations to the code and operating procedures.

4.1 FORTRAN Coding and System Dependencies

The NACHOS II program is written in ANSI standard FORTRAN 77 and should therefore be usable on any computer system that supports such a compiler. Use of the code with older compilers, *e.g.*, FORTRAN IV, would require some code modification to ensure the proper handling of character data with a given machine word length. Accuracy requirements in the code demand that the host computer utilize a "large" word length (*e.g.*, 60 or 64 bit word); computers with a 32 bit word (or less) must employ double precision. The double precision option is most easily implemented by placing the appropriate `IMPLICIT DOUBLE PRECISION` statement at the beginning of each routine in the program. Some changes to intrinsic functions, such as the `SIN` and `COS` functions, may also be required for a double precision implementation.

A dynamic dimensioning procedure is used in NACHOS II to allocate and release computer memory during program execution. Under this algorithm the individual vectors and arrays needed by the program are stored in (noncontiguous) blocks of memory under a single vector name. Pointers indicating the location of individual arrays within memory are maintained by the memory manager as is the allocation of needed storage space. Further details on the memory manager and its operation are given in [6]. Some computer systems may not permit execution time changes in memory allocation; modifications to the code and memory manager to handle this case are outlined in [6].

NACHOS II makes use of a few system dependent utilities. To increase portability of the code most of the system dependencies have been isolated in a few subroutines that are located in a utility library that accompanies the main code. The system dependent graphics routines are not included in this library and are handled separately. The utility library [6] includes such functions as calls to the date and time functions, routines needed for the dynamic dimensioning algorithm and the free field reader used for decoding input

to the code. The library routines are heavily commented to ease the task of converting these utilities to other computers.

The standard version of NACHOS II was developed primarily for use on large mainframe computers, such as the CRAY-1S and CRAY X-MP. In order to improve the efficiency of data transfer on these machines the code makes use of buffered input and output operations. During the matrix solution process the code uses the `BUFFER IN` and `BUFFER OUT` statements to transfer data to a disk file and the `UNIT` function to check for completion of the transfer operation. These routines are not ANSI standard and may have to be replaced with standard unformatted `READ` and `WRITE` statements on some computer systems. The buffered I/O operations take place in subroutines `FRONT, DISKWR` and `BCKSUB`. Comments in the code indicate the changes necessary to implement standard FORTRAN for these operations. When a solid state disk is available with the computer hardware, NACHOS II makes use of this device for the storage of element and matrix data. This hardware feature is activated with a call to a routine `OPNSSD` from within subroutine `RDINPT`; this line of code may be eliminated for machines that do not support such a device. Finally, for efficiency in vector operations the code also makes use of three CRAY library functions called `GATHER, ISRCHEQ` and `SDOT` [7]. FORTRAN versions of these routines have been included in the code in comment form and can be activated for computer systems not having these library functions.

4.2 Graphics Routines

The batch plotting facilities built into NACHOS II make use of the commercial graphics package DISSPLA [5]. For use at Sandia National Laboratories, the DISSPLA package works in conjunction with a series of low-level graphics routines contained in the Sandia Virtual Device Interface [8]. The use of the VDI package makes it possible to direct graphics output to a series of different plot devices [9] by changing only the job control instructions.

To simplify the implementation and conversion of graphics in NACHOS II, all of the system dependencies (*i.e.*, DISSPLA and VDI subroutine calls) have been isolated in a few subroutines within the code. These routines are listed below along with a brief description of their function. The subroutines are heavily commented to ease the task of conversion to a different plotting system.

`SUBROUTINE PLTFIL` - opens, intializes and closes the plot file

`SUBROUTINE ADVPLT` - advances the plot frame and resets the plot parameters

`SUBROUTINE TRMPLT` - ends a plot frame

`SUBROUTINE SIZPLT` - creates scaling for plot device, generates plot axes

`SUBROUTINE TITLES` - writes plot title and identification

`SUBROUTINE PLOTIT` - plots points, symbols and numbers, draws line segments and arrows

`SUBROUTINE GRPHIT` - generates an x,y graph

`SUBROUTINE CHR2HL` - converts character variables to a Hollerith array representation via an internal read

4.3 File Formats

The disk files used internally by NACHOS II are generally sequential access, unformatted files. The specific form of these files is not usually of concern to the user except for the restart (or solution) file, the format of which is described in Section 3.17.

Two files that are of concern to the user are the input file that is written by a stand-alone mesh generation program and the post-processing file written by NACHOS II. The formats for these two files follow a specified standard and must be adhered to for successful communication with the code. Detailed descriptions of these input and output formats are given in references [10,11] and are summarized in Appendices D and E, respectively. When attached to a NACHOS II run, the mesh generation file should be designated unit 10; the post-processing file written by the code is unit 22.

4.4 File Usage

NACHOS II makes use of a series of disk files for the storage of large blocks of element data, working space for the matrix solver, storage of solution data and input and output functions. All of the files used by the program are listed in the following table with a brief description of their function, their internal FORTRAN identification and format type. Specific formats for these files are not given as that information is readily available from a code listing. Previous sections have commented on those file formats that would normally be of concern to the user.

When files are to be attached to a NACHOS II job or saved after processing, the unit numbers used in the job control statements must conform to those indicated in the table. File names are arbitrary but must conform to the syntax rules for the particular operating system. The proper relation between file name and unit number must also be established in the job control statements. A utility program from the SUPES library [6] obtains needed file names from the operating system for use in opening files during program execution. All files are opened in the `OPNFIL` subroutine.

Unit Number	FORTRAN Name	File Usage	File Type
5	NIN	Input file	Formatted
6	NOUT	Output file	Formatted
10	NTP0	Mesh data (external program)	Unformatted
11	NTP1	Free field input	Unformatted
12	NTP2	Element data	Unformatted
13	NTP3	Penalty matrices	Unformatted
14	NTP4	Element matrices	Unformatted
15	NTP5	Stream function data	Unformatted
16	NTP6	Special output point data	Unformatted
17	NTP7	Heat flux data	Unformatted
18	NTP8	Fluid stress data	Unformatted
19	NTP9	Solution data	Unformatted
20	NTP10	Storage for matrix solver	Unformatted
21	NTP11	Scratch storage	Unformatted
22	NTP12	Post-processing data	Unformatted
23	NTP13	Initial conditions (restart)	Unformatted

4.5 Access to the Code

The source version of NACHOS II is maintained on the integrated file system (IFS) of the Central Computer Facility at Sandia National Laboratories. The program may be accessed by the following MASS utility [12] command

```
GET filnam : /FECODES/NACHOS/NACHOS2
```

The `filnam` parameter is the local file name for the code on the host computer. This is the standard version of the program that is designed to run on the Sandia National Laboratories mainframe computers. If NACHOS II is to be run on some other machine (*e.g.*, a VAX or Control Data) minor changes to this version of the program will probably have to be made as outlined in Section 4.1.

A compiled version of the code is also avilable on the IFS and may be obtained with the MASS command

```
GET filnamb : /FECODES/NACHOS/NACHOS2B
```

This binary code corresponds to a compilation of the source code using the latest available compiler on the mainframe computers. The compiled version of the code can be loaded with the needed utility libraries, graphics libraries and optional user subroutines to create a file that is executable on the appropriate machine.

The utility libraries needed for the execution of NACHOS II are available for several computer systems. Access to these routines varies depending on the particular operating system. Instructions for accessing and loading these routines are available in [6]. When user supplied subroutines are used with NACHOS II they must be compiled and then made available to the loader as a library to be searched for unsatisfied external references.

The graphics routines used by NACHOS II consist of the DISSPLA plot package [5] and the Virtual Device Interface (VDI) routines [8,9] available at Sandia National Laboratories. The routines in these packages are available as system libraries and may be accessed through standard options in the loading sequence. A further description of the use of these plotting routines is available in [9].

In order to assist users in the running of NACHOS II, a standard job control file is available on the IFS that allows execution of the code in a batch mode on the mainframe computers. This file may be obtained with the MASS command

```
GET jcifil : /FECODES/NACHOS/BATCHJB
```

This file contains the appropriate commands for accessing the code, getting the utility and plot routines, loading and executing the job. Users may run this file with the appropriate NACHOS II data files via the SENDCTSS utility from a VAX or use the file commands as a guide to interactive execution of the program.

5 Example Problems

A series of three problems have been included in this section to demonstrate the use of the previously described command cards and some of the capabilities of the NACHOS II program. Though the examples were designed to illustrate the basic features of the code, all possible options and subtleties of the program could not be covered. A more extensive demonstration of the use of NACHOS II is given in [4].

5.1 Problem 1 - Grid Generation

The first example consists of three problems which illustrate several points about the relationship between mesh point generation and finite element construction when using the program's internal mesh generator. The L shaped region shown in Figure 10 is to be discretized using a number of quadrilateral elements. The input listings in Figure 11 show three different methods (hereafter denoted A, B and C) by which this task could be accomplished.

In the three cases illustrated, 147 mesh points were generated in two parts; the physical location of the mesh points remained constant for all cases. A plot of the mesh points, as generated by the **PLOT**, POINTS command, is shown in Figure 12.

The input deck for Case A resulted in the element mesh shown in Figure 13a. Note that in generating the elements, only the first I, J pair for the element was specified, with the program assuming default values for the remaining I, J quantities. The mesh generated by input deck B, produced the grid shown in Figure 13b. In this case, the I, J values for all the corner nodes were specified for the elements in the left-hand side of the domain; the I, J values for the midside nodes were allowed to take the default (averaged) values. The difference between Case A and B illustrates that the user is free to choose any meaningful group of mesh points in the construction of an element.

The input deck for Case C shows the use of a discontinuous I, J numbering scheme between adjacent parts of the grid. In the second part of the mesh generation (Figure 11c) the I numbering is started with 23 instead of 13 as in Cases A and B. The mesh points that are common to the two parts of the domain (*i.e.*, share the same physical location as denoted by the X's in Figure 12) thus have two legitimate I, J names (*e.g.* (13,1) = (23,1)). In the construction of elements that use these doubly named mesh points, care must be taken to consistently use only one set of names. This is shown in input deck C where the I, J names from the first part of the grid are used for the common mesh points. The mesh generated by deck C is identical to that of deck A (Figure 13a).

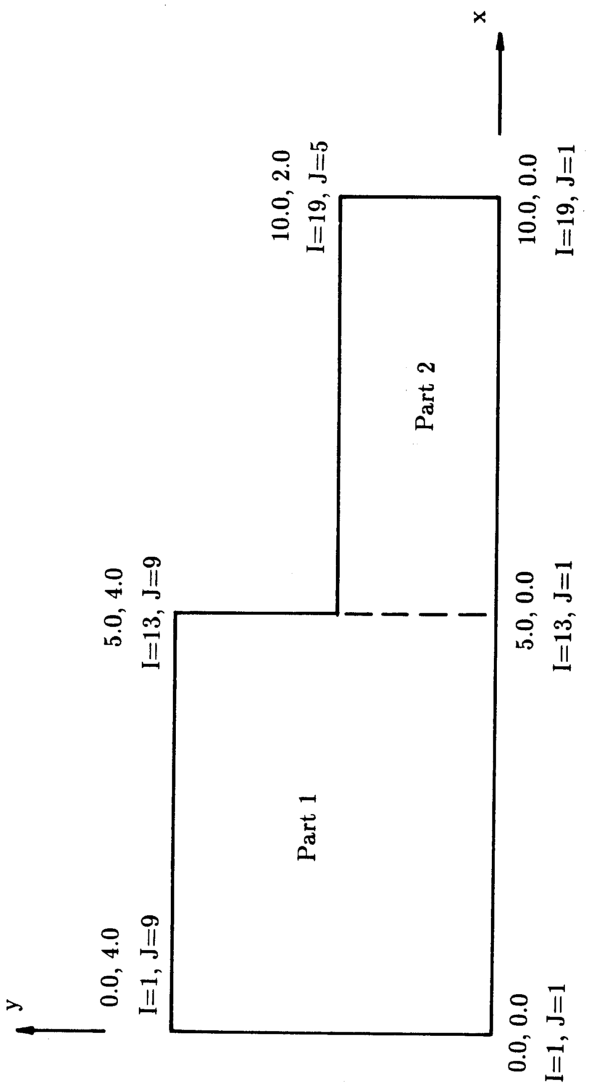


Figure 10: Scematic for Example Problem 1.

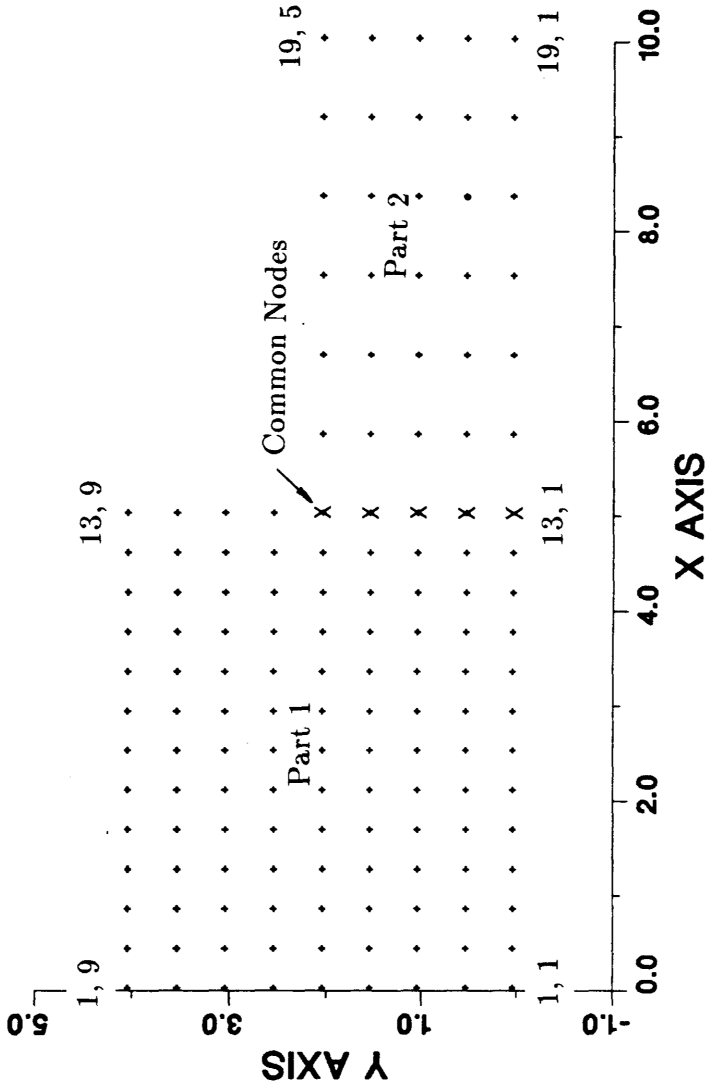


Figure 11: Input listings for Example Problem 1.

.

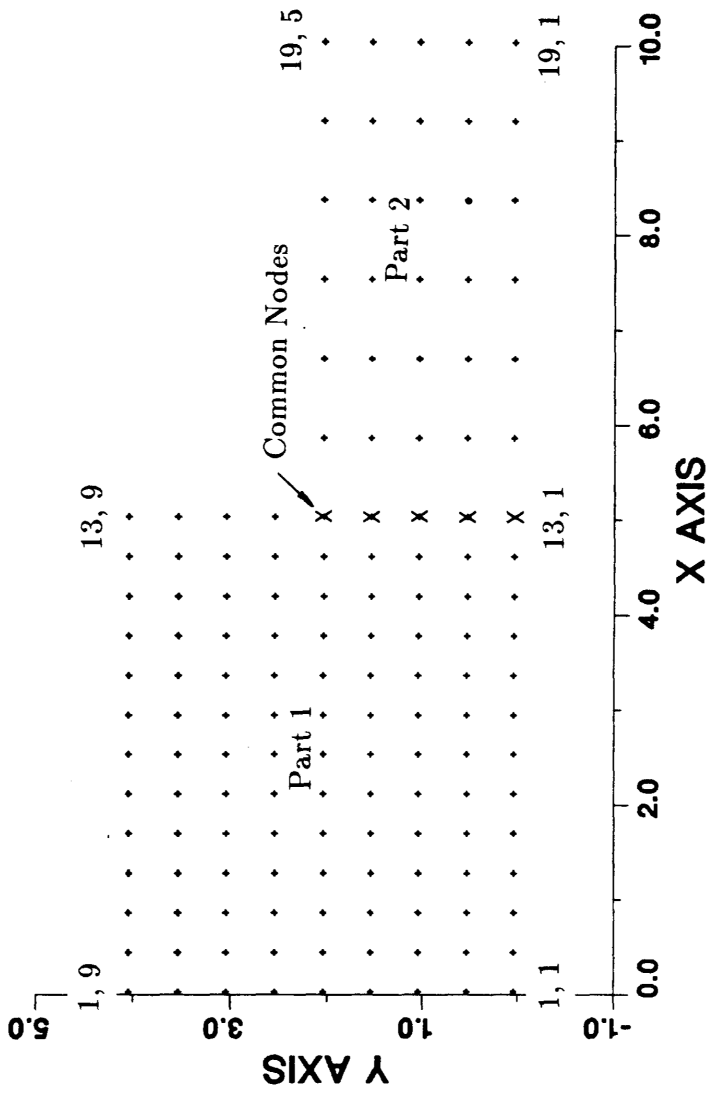
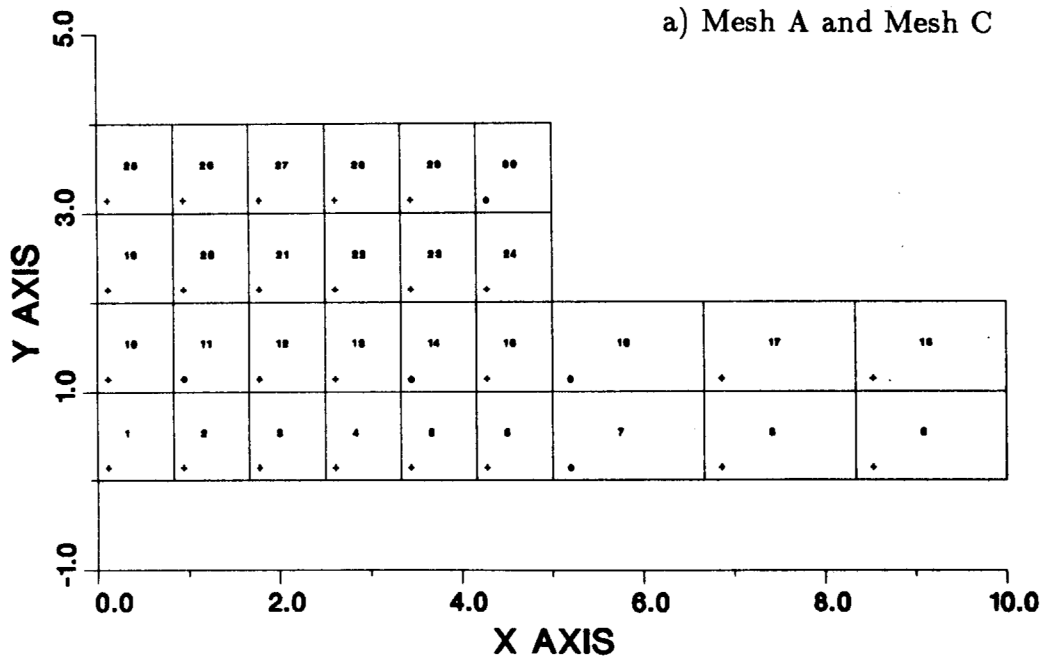


Figure 12: Points plot for Example Problem 1.

.



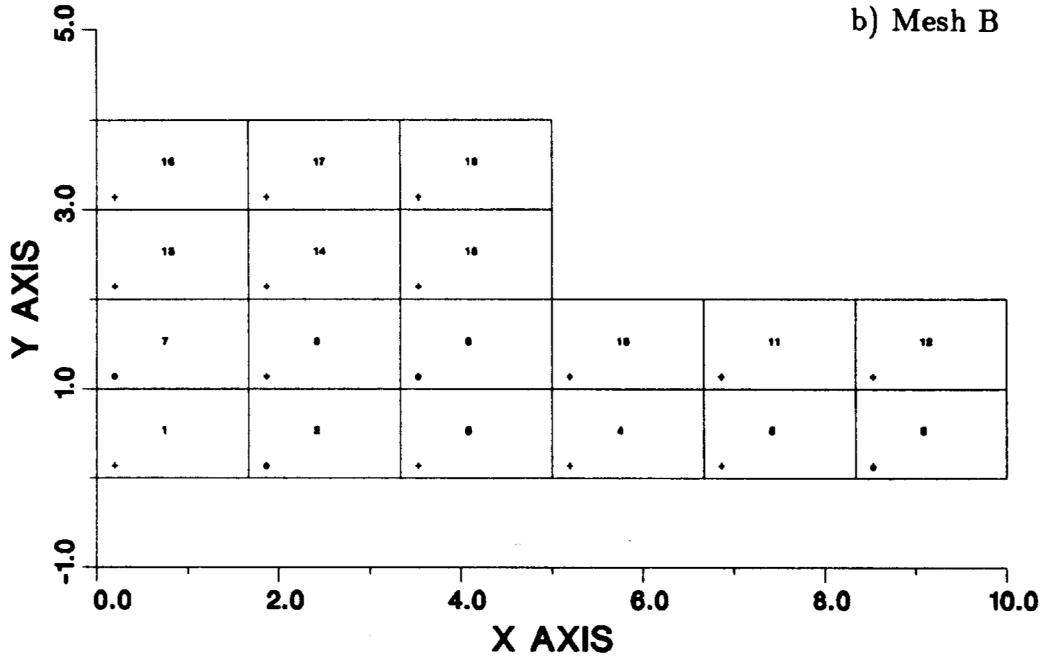


Figure 13: Element plots for Example Problem 1.

5.2 Problem 2 - Flow in a Constricted Tube

The second example has been selected to demonstrate the solution of a simple isothermal, flow problem. The geometry consists of an axisymmetric tube containing a smooth, internal constriction as shown in Figure 14. The shape of the constriction was specified by the equation

$$r = 1.0 - 0.5exp(-4z^2)$$

which yielded a maximum reduction in the tube cross section of 75 % . Fluid flows through the tube under the action of an applied normal stress condition. The Reynolds number for the problem (based on average inlet velocity and tube diameter) is approximately 64. The boundary conditions and material properties for the problem are also shown in Figure 14.

The NACHOS II input deck for this problem is shown in Figure 15. The mesh was generated in six parts; the reflection option was utilized to locate mesh points symmetrically about the constriction. A plot of the mesh generated by NACHOS II is shown in Figure 16. The flow problem was solved using the Newton procedure with convergence to the indicated tolerance obtained in six iterations. Representative plots of the stream function and several velocity profiles across the tube are shown in Figures 17 and 18. Flow separation downstream of the constriction is clearly observed in both the streamline plot and the velocity fields.

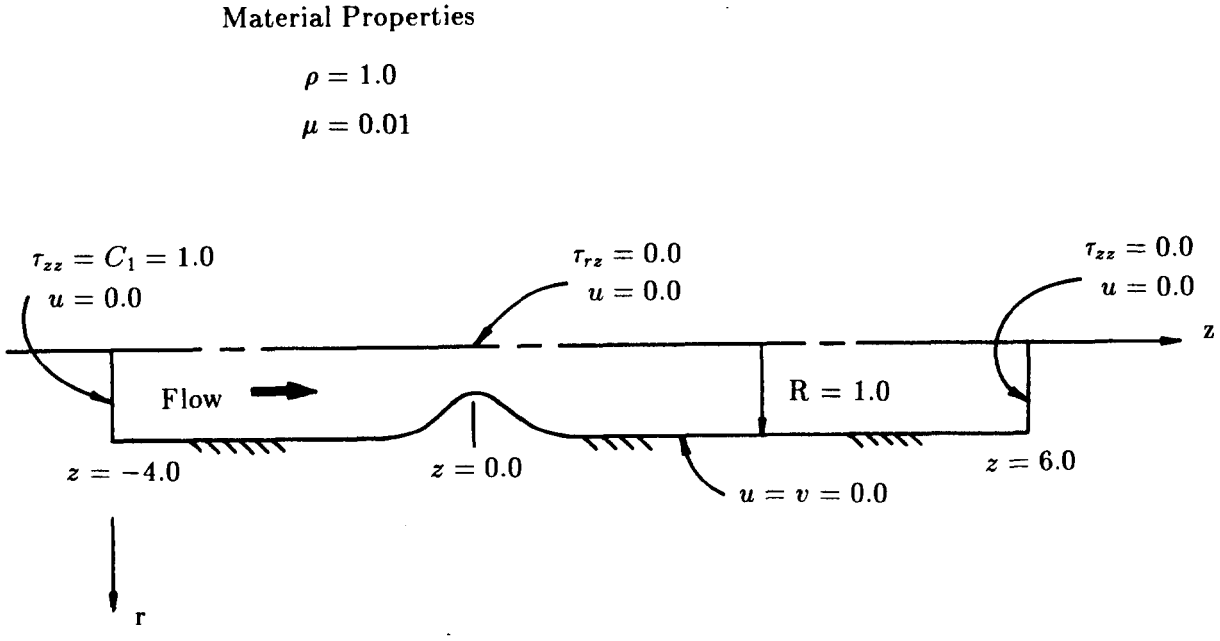


Figure 14: Schematic for Example Problem 2.

.

```
$EXAMPLE PROBLEM TWO
$ISOTHERMAL FLOW IN A CONSTRICTED TUBE
$WALL SHAPE GIVEN BY R = 1.0 - 0.5*EXP(-4*Z**2)
$REYNOLDS NUMBER = 64 BASED ON AVERAGE INLET VELOCITY AND
$TUBE DIAMETER
MATERIALS
WATER,NEWTONIAN,1,1.0,.01
END
MESH,INTERNAL,17,111,2
QBLOCK,1,1,17,31,,4.,,4.
0.,1.0,1.0,0.
-4.0,-4.0,-1.0,-1.0
QBLOCK,1,31,17,35
0.,1.0,.961,0.,,.990
-1.0,-1.0,-0.8,-0.8,,-0.9
QBLOCK,1,35,17,43
0.,.961,.736,0.,,.882
-0.8,-0.8,-0.4,-0.4,,-0.6
QBLOCK,1,43,17,51,,,2.0
0.,.736,.500,0.,,.574
-0.4,-0.4,0.,0.,,-0.2
REFLECT,1,1,17,51,1,101,17,51
0.,0.,.5,0.
QBLOCK,1,101,17,111
0.,1.0,1.0,0.
4.0,4.0,6.0,6.0
END
ELEMENTS,450,PRESCRIBED,1
JLOOP,55,2
ILOOP,8,2
QUAD8/8,1,1,1
IEND
JEND
ILOOP,8,2
BC,USIDE,1,1,1,0.
BC,TNRMLSIDE,1,1,1,1.0
BC,USIDE,1,109,3,0.
BC,TNRMLSIDE,1,109,3,0.
IEND
```

Figure 15: Input list for Example Problem 2.

```
JLOOP,55,2
BC,USIDE,1,1,4,0.
BC,STICK,15,1,2,0.
JEND
END
OUTPUT,FIELDS
STRING,121,280
END
FORMKF,AXISYM
SOLVE
STEADY,NEWTON,,8,8,.005
END
STREAM,0.
POST
NODES,4,UVEL,VVEL,PRESS,STREAM
TIMEPLANE,ALL
END
STOP
```

Figure ????: (continued) Input list for Example Problem 2.

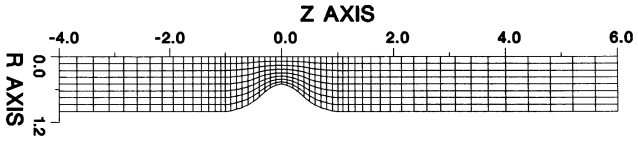


Figure 16: Element plot for Example Problem 2.

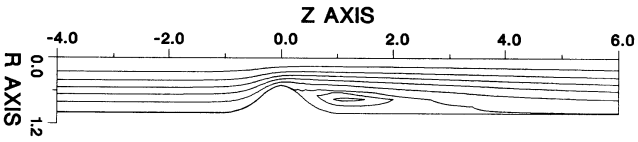


Figure 17: Contour plot for Example Problem 2.

Figure 16: Element plot for Example Problem 2.

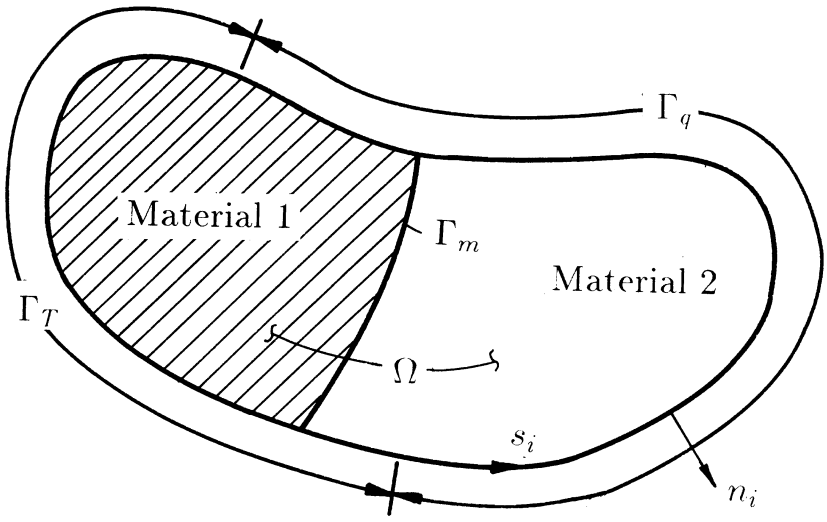


Figure 17: Contour plot for Example Problem 2.

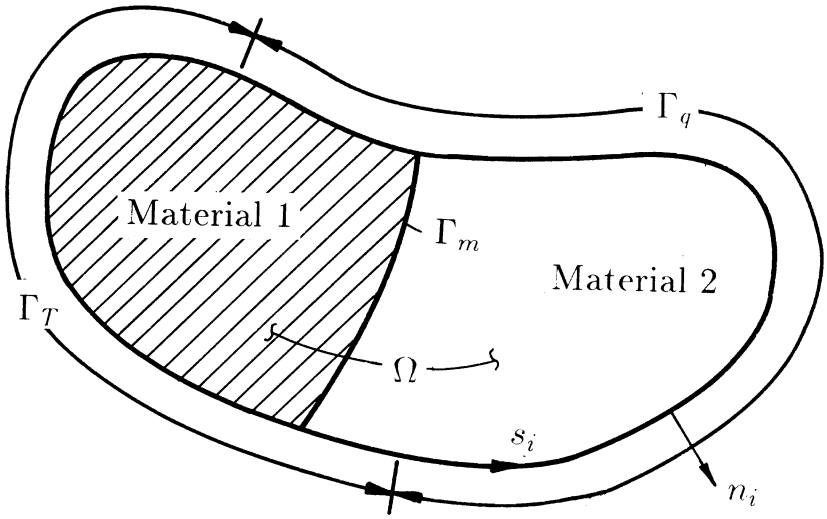


Figure 18: Profile plots for Example Problem 2.

5.3 Problem 3 - Free Convection in an Enclosure

The third example has been selected to illustrate grid generation for a complex geometry and the solution of a buoyancy driven flow. The physical problem consists of a heated, isothermal, circular cylinder, mounted horizontally in an isothermal rectangular enclosure. The space between the cylinder and enclosure is filled with air at atmospheric pressure. The problem geometry is shown in Figure 19 along with the material properties and boundary conditions used in the analysis. The NACHOS II input deck for the problem is presented in Figure 20.

The grid points for this problem were generated in eight parts as indicated in Figure 21. The resulting element plot is shown in Figure 22. A steady-state solution to the problem was requested using the Picard solution procedure with a relaxation parameter. Commands to generate contour plots of the stream function and temperature fields were included in the input deck; a velocity vector plot was also requested. The indicated plots produced by NACHOS II are shown in Figure 23.

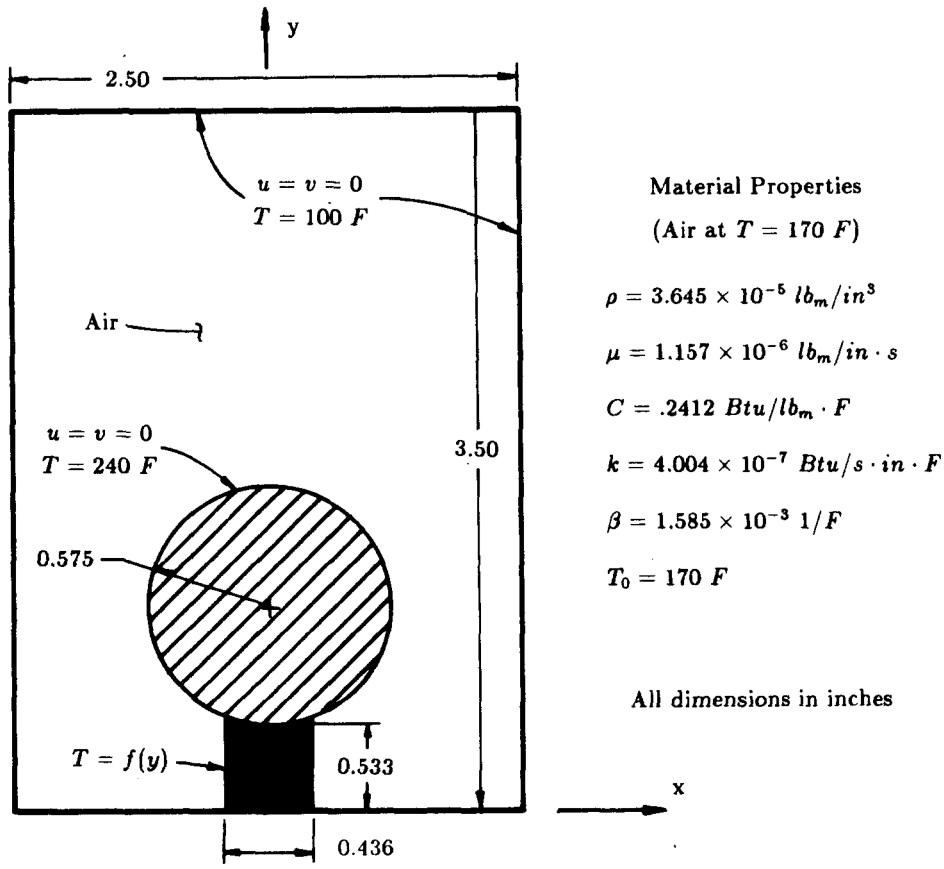


Figure 19: Schematic for Example Problem 3.

.

```
$EXAMPLE THREE
$FREE CONVECTION IN AN ENCLOSURE
$ISOTHERMAL, HORIZONTAL CYLINDER PLACED IN AN
$ISOTHERMAL, RECTANGULAR ENCLOSURE
$PROBLEM DIMENSIONS ARE IN INCHES
$MATERIAL PROPERTIES ARE IN ENGLISH UNITS EVALUATED AT 170 F
MATERIALS
AIR,NEWTONIAN,1,3.645E-5,1.157E-6,.2412,4.004E-7,1.587E-3,,386.4,,,,170.,170.
END
MESH,INTERNAL,42,44,2
QBLOCK,1,1,7,7                          $ PART 1
.218,.580,.580,.218,,,.419
0.,0.,.450,.240,,,.322
QBLOCK,1,26,11,32                       $ PART 2
.580,1.250,1.250,.580
0.,0.,.450,.450
QBLOCK,1,7,7,15                         $ PART 3
.218,.580,.419,.218,.419,,.330
.240,.450,.611,.479,.322,,.538
QBLOCK,7,7,19,15                        $ PART 4
.580,.580,.419,.419,.822,,.594
.450,1.612,1.451,.611,1.031,,1.031
QBLOCK,1,32,11,44                       $ PART 5
.580,1.250,1.250,.580,,,,.822
.450,.450,1.612,1.612,,,,1.031
QBLOCK,19,7,25,15                       $ PART 6
.580,0.,0.,.419,.310,,.222
1.612,1.854,1.626,1.451,1.793,,1.585
QBLOCK,26,1,32,15                       $ PART 7
0.,.580,.580,0.,.310
1.854,1.612,3.500,3.500,1.793
QBLOCK,32,1,42,15                       $ PART 8
.580,1.250,1.250,.580
1.612,1.612,3.500,3.500
END
ELEMENTS,200,PRESCRIBED,2               $ ELEMENT DATA
ILOOP,2,2
JLOOP,3,2
QUAD8/8,1,1,1
JEND
IEND
```

Figure 20: Input list for Example Problem 3.

```
QUAD8/8,1,5,1,1,26,1,28,5,3,6,1,1,27,6,3,5,2
QUAD8/8,1,5,3,1,28,1,30,5,5,6,3,1,29,6,5,5,4
QUAD8/8,1,5,5,1,30,7,7,5,7,6,5,1,31,6,7,5,6
ILOOP,5,2
JLOOP,2,2
QUAD8/8,1,1,26
JEND
IEND
QUAD8/8,1,1,30,3,30,3,32,7,7,2,30,3,31,2,32,1,31
ILOOP,4,2
QUAD8/8,1,3,30
IEND
ILOOP,3,2
JLOOP,4,2
QUAD8/8,1,1,7
JEND
IEND
ILOOP,6,2
JLOOP,4,2
QUAD8/8,1,7,7
JEND
IEND
QUAD8/8,1,7,7,3,32,3,34,9,7,2,32,3,33,2,34,8,7
QUAD8/8,1,9,7,3,34,3,36,11,7,2,34,3,35,2,36,10,7
QUAD8/8,1,11,7,3,36,3,38,13,7,2,36,3,37,2,38,12,7
QUAD8/8,1,13,7,3,38,3,40,15,7,2,38,3,39,2,40,14,7
QUAD8/8,1,15,7,3,40,3,42,17,7,2,40,3,41,2,42,16,7
QUAD8/8,1,17,7,3,42,3,44,19,7,2,42,3,43,2,44,18,7
JLOOP,6,2
ILOOP,4,2
QUAD8/8,1,3,32
IEND
JEND
ILOOP,3,2
JLOOP,4,2
QUAD8/8,1,19,7
JEND
IEND
```

Figure 20: (continued) Input list for Example Problem 3.

```
QUAD8/8,1,25,7,23,7,28,3,26,3,24,7,28,2,27,3,26,2
QUAD8/8,1,23,7,21,7,30,3,28,3,22,7,30,2,29,3,28,2
QUAD8/8,1,21,7,19,7,32,3,30,3,20,7,32,2,31,3,30,2
QUAD8/8,1,19,7,3,44,34,3,32,3,2,44,34,2,33,3,32,2
ILOOP,4,2
QUAD8/8,1,3,44,5,44,36,3,34,3,4,44,36,2,35,3,34,2
IEND
ILOOP,8,2
JLOOP,6,2
QUAD8/8,1,26,3
JEND
IEND
ILOOP,3,2                          $ BOUNDARY CONDITIONS
BC,STICK,1,1,1,0.0
BC,TSIDE,1,1,1,100.
IEND
ILOOP,5,2
BC,STICK,1,26,1,0.0
BC,TSIDE,1,26,1,100.
IEND
JLOOP,7,2
BC,STICK,1,1,4,0.0
JEND
JLOOP,10,2
BC,STICK,9,26,2,0.0
BC,TSIDE,9,26,2,100.
JEND
JLOOP,6,2
BC,STICK,40,3,2,0.0
BC,TSIDE,40,3,2,100.
JEND
{\begin{figure}[ht]
%\begin{footnotesize}
\include{example3.inp}
\begin{verbatim}
ILOOP,8,2
BC,STICK,26,13,3,0.0
BC,TSIDE,26,13,3,100.
IEND
BC,USIDE,25,7,4,0.0
```

Figure 20: (continued) Input list for Example Problem 3.

```
JLOOP,6,2
BC,USIDE,26,3,4,0.0
JEND
JLOOP,4,2
BC,USIDE,23,7,2,0.
JEND
ILOOP,12,2
BC,STICK,1,13,3,0.0
BC,TSIDE,1,13,3,240.
IEND
BC,T,1,1,8,110.
BC,T,1,1,4,120.
BC,T,1,3,8,130.
BC,T,1,3,4,140.
BC,T,1,5,8,150.
BC,T,1,5,4,160.
BC,T,1,7,8,170.
BC,T,1,7,4,180.
BC,T,1,9,8,190.
BC,T,1,9,4,200.
BC,T,1,11,8,210.
BC,T,1,11,4,220.
BC,T,1,13,8,230.
END
FORMKF,,FREE,MIXED
SOLVE,,,600000
STEADY,PICARD,0.50,6,7,.001,.001
STEADY,PICARD,0.25,6,7,.001,.001
STEADY,PICARD,0.00,10,10,.001,.001
END
STREAM,0.
POST
NODES,5,UVEL,VVEL,PRESS,TEMP,STREAM
TIMEPLANE,ALL
END
STOP
```

Figure 20: (continued) Input list for Example Problem 3.

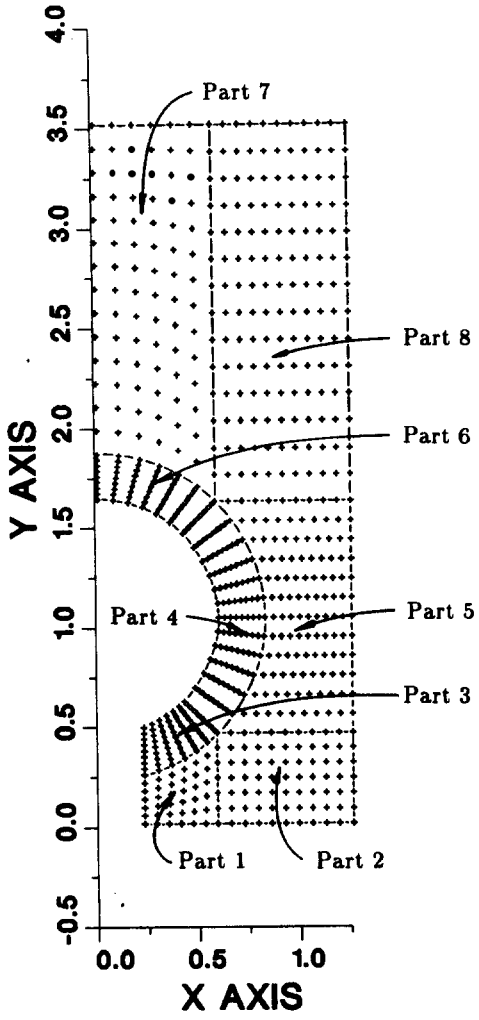


Figure 21: Point plot for Example Problem 3.

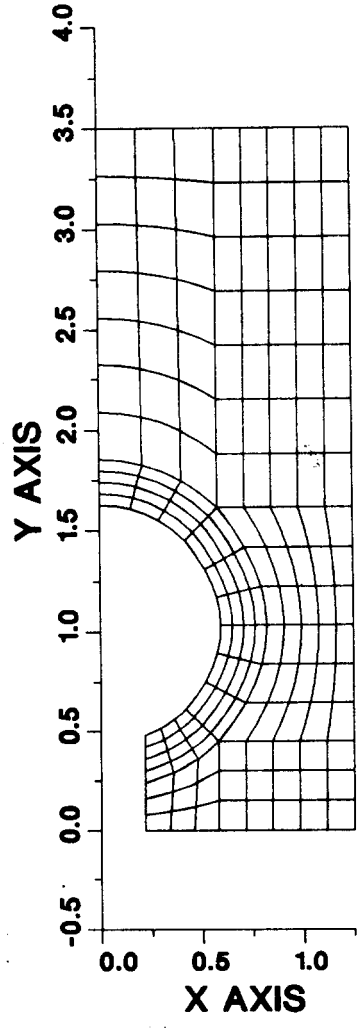


Figure 22: Element plot for Example Problem 3.

.

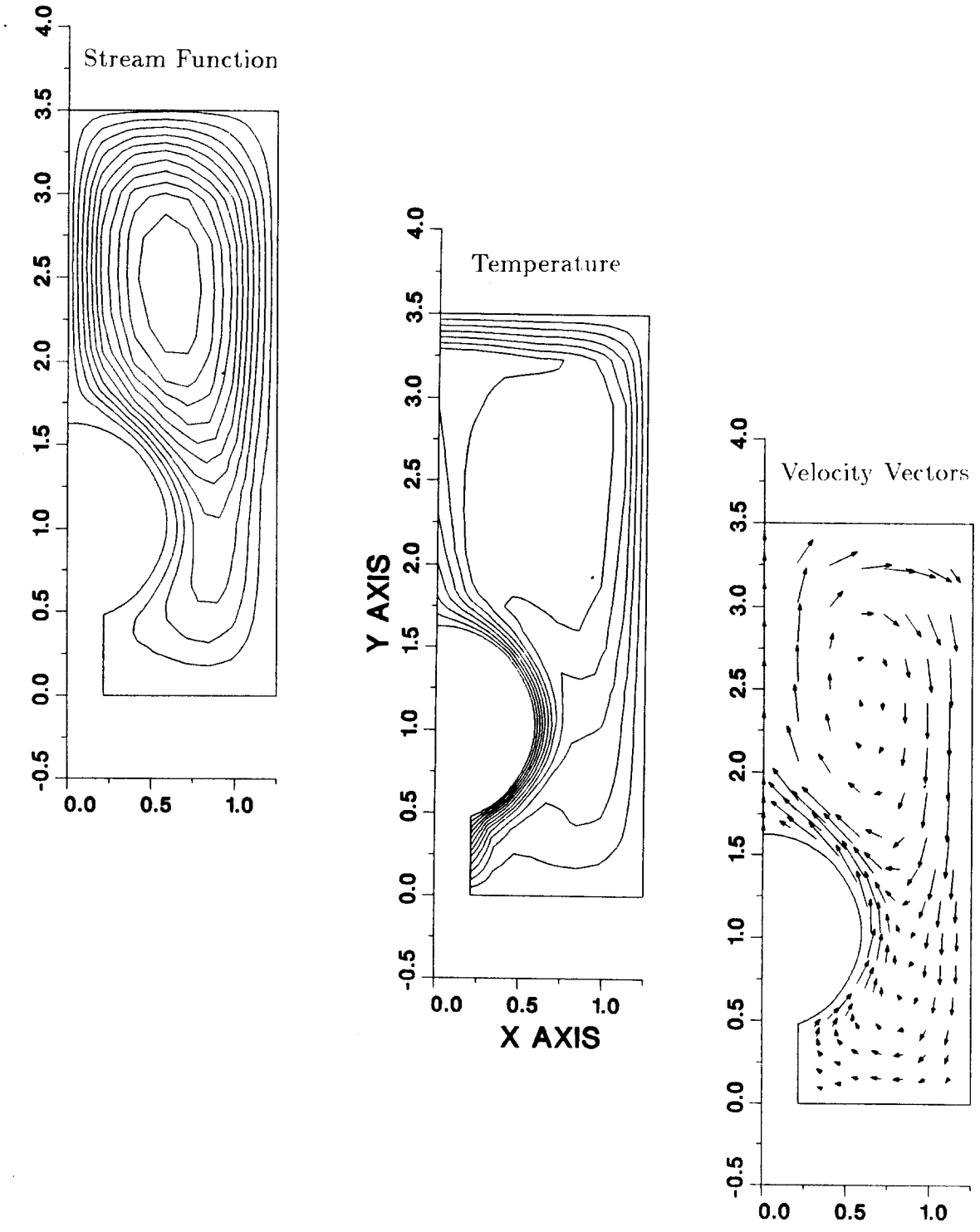


Figure 23: Contour and vector plots for Example Problem 3.

6 References

1. D. K. Gartling, "COYOTE II - A Finite Element Computer Program for Nonlinear Heat Conduction Problems, Part I - Theoretical Background," SAND86-1844, Sandia National Laboratories, Albuquerque, N.M. (1987)

2. D. K. Gartling and C. E. Hickox, "MARIAH - A Finite Element Computer Program for Incompressible Porous Flow Problems," SAND79-1622, Sandia National Laboratories, Albuquerque, N.M. (1979)

3. D. K. Gartling, "NACHOS II - A Finite Element Computer Program for Incompressible Flow Problems, Part I - Theoretical Background," SAND86-1816, Sandia National Laboratories, Albuquerque, N.M. (1987)

4. D. K. Gartling, "NACHOS II - A Finite Element Computer Program for Incompressible Flow Problems, Part III - Example Analyses," SAND86-1818, Sandia National Laboratories, Albuquerque, N.M. (1987)

5. "DISSPLA - User's Manual," Version 10.0, Integrated Software Systems Corporation, San Diego, California (1985)

6. D. P. Flanagan, W. C. Mills-Curran and L. M. Taylor, "SUPES - A Software Utilities Package for the Engineering Sciences," SAND86-0911, Sandia National Laboratories, Albuquerque, N.M. (1986)

7. "CRAY X-MP and CRAY-1 Computer Systems, Library Reference Manual," SR-0014, CRAY Research, Inc., Mendota Heights, Minnesota (1984)

8. K. M. Erickson and R. W. Simons, "Functional Specification of the Sandia Virtual Device Interface," SAND81-1900, Sandia National Laboratories, Albuquerque, N.M. (1982)

9. "VDI - The Virtual Device Interface User's Guide," Sandia National Laboratories, Albuquerque, N.M. (1982)

10. L. M. Taylor, D. P. Flanagan and W. C. Mills-Curran, "The Genesis Finite Element Mesh File Format," SAND86-0910, Sandia National Laboratories, Albuquerque, N.M. (1986)

11. W. C. Mills-Curran, A. P. Gilkey and D. P. Flanagan, "EXODUS: A Finite Element File Format for Pre- and Postprocessing," SAND87-2997, Sandia National Laboratories, Albuquerque, N.M. (1988)

12. "MASS Reference Manual," Version 1.0, Sandia National Laboratories, Albuquerque, N.M. (1983)

13. "PATRAN User's Guide," Release 1.5, PDA Engineering, Santa Ana, California (1984)

Appendix A - Summary of Input Commands

In this section all of the command and data cards recognized by NACHOS II are summarized. No attempt is made to define the parameters on each input line since these descriptions are available in the main text.

Title and Comments :

$ A TITLE LINE
$ A SERIES
$ OF
$ COMMENT LINES
.
.

MATERIALS Command :

MATERIALS
mat name, mat type, mat number, ρ_0, μ, C, k, β, g_x, g_y, variable properties, *
Q, viscous dissipation, T_0, T_{init}
.
.
mat name, POROUS, mat number, fluid number, κ, μ_e, ϕ, $\hat{c}$, $(\rho_0 C)_e$, k_e, *
variable properties, Q, T_{init}
.
.
mat name, AUXDATA, mat number, C_1, D_1, γ_1, Q_1, c_{1_0}, $c_{1_{init}}$, *
C_2, D_2, γ_2, Q_2, c_{2_0}, $c_{2_{init}}$
.
END

MESH Command :

MESH, type of generator, imax, jmax, *iprint*
.
.
QBLOCK, i1, j1, i3, j3, *g1, g2, g3, g4, polar, xpolar, ypolar*
x1, x2, x3, x4, *x5, x6, x7, . . . , x12*
y1, y2, y3, y4, *y5, y6, y7, . . . , y12*
.
.
TBLOCK, i1, j1, i2, j2, i3, j3, *g1, g3, polar, xpolar, ypolar*

x1, x2, x3, *x4, x5, . . . , x9*
y1, y2, y3, *y4, y5, . . . , y9*
.
.
ARC, i1, j1, i2, j2, *g1, polar, xpolar, ypolar*
x1, x2, *x3, x4*
y1, y2, *y3, y4*
.
.
POINT, i1, j1, x1, y1, *polar, xpolar, ypolar*
.
.
COPY, i1, j1, i3, j3, inew1, jnew1, inew3, jnew3
xnew1, ynew1, xnew3, ynew3, *polar, xpolar, ypolar*
.
.
REFLECT, i1, j1, i3, j3, inew1, jnew1, inew3, jnew3
xline1, yline1, xline2, yline2, *polar, xpolar, ypolar*
.
.
FILLIN, i1, j1, i3, j3, α, ω, *iter, tol*
.
.
EXTDEF
.
.
END

ELEMENTS Command :

ELEMENTS, no. elements, *order*, *iprint*
.
.
JLOOP, npass, inc
ILOOP, npass, inc
.
element type, mat, i1, j1, *i2, j2, i3, . . ., in, jn*
.
.
BC, b.c. type, i1, j1, side/node, b.c. data
.
.
IEND

JEND

.

SET, b.c. type, set no., param1, param2

.

.

END

FORMKF Command :

FORMKF, *geometry, flow type, f.e. method, penalty parameter, * pressure approximation, var1, var2*

OUTPUT Command :

OUTPUT, FIELDS
delimiter type, e1, e2, e3, e4, . . . , e50

.

.

END

or

OUTPUT, POINTS
x1, y1, x2, y2, . . . , x25, y25

.

.

END

SOLVE Command :

SOLVE, *restart, timeplane, maxmem*

.

.

STEADY, iterative method, *relax. factor, no. iter, iprint, tol u, tol T*, $tolc_1$, $tolc_2$

.

.

TRANSIENT, integration method, *time step option, integration tol, * t_{init}, t_{final}, Δt, no. steps, steady-state tol, iprint*

.

.

END

STREAM Command :

STREAM, *psi base, output, no. timeplanes*

FLUX Command :

FLUX, element location, *save data, mesh location*
.
.
flux type, timeplane no.
.
END

PLOT Commands :

PLOT, POINTS, xmin, ymin, xmax, ymax, *xscale, yscale*

PLOT, ELEMENTS, xmin, ymin, xmax, ymax, *xscale, yscale*

PLOT, OUTLINE, xmin, ymin, xmax, ymax, *xscale, yscale*

PLOT, CONTOUR, xmin, ymin, xmax, ymax, *xscale, yscale*
contour type, timeplane no., no. contours, *max contour, min contour*,*
c1, c2, . . ., c25
.
.
END

PLOT, VECTOR, xmin, ymin, xmax, ymax, *xscale, yscale*
vector type, timeplane no., *vectorscale, vectormax, vectormin*
.
.
END

PLOT, HISTORY, xmin, ymin, xmax, ymax
plot variable, no. points, timeplane1, timeplane2, e1, n1, e2, n2, . . ., e10, n10
.
.

END

PLOT, PROFILE, xmin, ymin, xmax, ymax
TIMEPLANE, no. timeplanes, t1, t2, ..., t10
plot variable, profile spec., no. points, e1, s/n1, e2, s/n2, ..., e50, s/n50
.
.
END

PLOTSET Command :

PLOTSET, *number, special pts., nodes, axes, grid, symbol, font*

POST Command :

POST
data type, no. var, name1, name2, ..., namen
.
.
TIMEPLANE, ALL
or TIMEPLANE, INCREMENT, timeplane1, timeplane2, inc
or TIMEPLANE, SPECIFIED, no. timeplanes, t1, t2, ..., tn
END

Termination Command :

STOP

Appendix B - Consistent Units

The following list provides examples of consistent units for quantities that may be encountered in the use of NACHOS II.

Quantity	English	Metric	S I
Length	foot (ft)	centimeter (cm)	meter (m)
Time	second (s)	second (s)	second (s)
Mass	lb_m	gram (gm)	kilogram (kg)
Force	lb_m-ft/s^2	gm-cm/s^2	Newton (N)
Energy	Btu	erg	joules (J)
Temperature	Fahrenheit (F) or Rankine (R)	Centigrade (C) or Kelvin (K)	Centigrade (C) or Kelvin (K)
Gravitational Acceleration	ft/s^2	cm/s^2	m/s^2
Density	lb_m/ft^3	gm/cm^3	kg/m^3
Velocity	ft/s	cm/s	m/s
Stress (Pressure)	lb_m/ft-s^2	gm/cm-s^2	pascal (N/m^2)
Viscosity	lb_m/ft-s	gm/cm-s	pascal-s
Specific Heat	Btu/lb_m-F	erg/gm-C	J/kg-K
Power	Btu/s	erg/s	J/s (Watt)
Heat Flux	Btu/ft^2-s	erg/cm^2-s	J/m^2-s
Heat Transfer Coefficient	Btu/ft^2-s-F	erg/cm^2-s-C	J/m^2-s-K
Thermal Conductivity	Btu/ft-s-F	erg/cm-s-C	J/m-s-K
Coefficient of Volumetric Expansion	1/F	1/C	1/K
Volumetric Heat Source	Btu/ft^3-s	erg/cm^3-s	J/m^3-s
Permeability	ft^2	cm^2	m^2
Porosity	-	-	-
g_c	32.174 lb_m-ft/lb_f-s^2	1 gm-cm/dyne-s^2	1 kg-m/N-s^2
σ	4.761×10^{-13} Btu/s-ft^2-R^4	5.669×10^{-5} erg/s-cm^2-K^4	5.670×10^{-8} J/s-m^2-K^4

Appendix C - Commands for Use With an External Mesh Generator

Mesh data may be input to NACHOS II from an external, stand-alone mesh generation program through the use of a standardized file format. This format is documented in Reference [10] and is also outlined in Appendix D. Mesh generation programs such as PATRAN [13] are thus available to provide input to NACHOS II.

When an external program supplies the mesh, element and boundary condition data to NACHOS II, two of the normal input commands to the code have an altered format. The **MESH** command is simplified since no internal mesh generation operations are required. The **ELEMENTS** command is also simplified, since only element and boundary condition types need to be input through this command. The formats for the modified forms of these commands are described below.

MESH Command Card

The basic **MESH** command for use with an external mesh generator is of the form

MESH, type of generator
mesh file format
END

where

type of generator (C) : is a parameter to specify the type of mesh generator being used. To use the external mesh generator set this parameter to EXTERNAL.

mesh file format (C) : is a parameter to specify the type of format used to input the mesh data. The permissable value is GENESIS. Other mesh formats can be included with modifications to the program.

ELEMENTS Command Card

The basic **ELEMENTS** command for use with an external mesh generator is of the form

ELEMENTS, , ,*iprint*

.

.

element data

.

boundary condition data

.

.

END

where,

iprint (I) <2> : determines the amount of printed output produced during the element operations. Output increases with the value of *iprint* and ranges from no printout for *iprint* = 1 to full printout for *iprint* = 4.

The data associated with this command is of two types. One data set describes the types of elements to be used in the problem, while the second data set specifies the types of boundary conditions imposed on the problem. Though the two types of data may be mixed in the input deck, they are described here in separate sections.

Element Data :

Element data from the external mesh generator is provided to NACHOS II in a blocked format. All elements within a particular element block share a common material type (number) and finite element type. The individual element blocks have a unique (integer) identifier that is assigned by the user during the mesh generation operation. The data statements in this section are used to associate specific material numbers and element types with each element block. Reference [10] provides more information on the data structure and blocking of the external mesh file.

For each element block in the mesh, a data card of the following form is required

element type, mat, element block id

where,

element type (C) : is the name of the type of element. The element types available in NACHOS II are described in Section 3.5.

mat (I) : is the material number for the element. This number should be set to correspond to the "mat number" parameter used on the material property cards.

element block id (I) : is the previously assigned identifier for a block of elements in the mesh. The elements in this block are expected to all be the same material and be the same type of finite element (see Note 1).

Boundary Condition Data :

The boundary condition data from the external mesh generator is blocked into groups according to the type of boundary condition that is to be applied to a particular group of nodes or element sides. Each type of boundary condition has a unique (integer) identifier that is assigned by the user during the mesh generation operation. The data in this section serves to associate the boundary condition flags with specific types of boundary conditions and the actual value of the boundary data. Boundary condition data may appear at any point in the present data section.

For each boundary condition identifier in the mesh file, a data card of the following form is required

BC, b.c. type, b.c. id, b.c. data

where,

b.c. type (C) : is the name for the type of boundary condition. The types of boundary conditions available in NACHOS II are described in Section 3.5.

b.c. id (I) : is the previously assigned identifier for a group of boundary conditions. The boundary conditions in this group are all expected to be of the same type and have the same numerical value for the boundary condition (see Note 2).

b.c. data (R or I) : is the numerical value of the applied boundary condition or an integer pointer to a location containing boundary condition data. For a specified boundary condition, this parameter is set equal to the numerical value of the boundary value. For boundary conditions requiring multiple input parameters (*e.g.*, heat transfer coefficient and reference temperature) the "b.c. data" parameter is an integer SET number; the use of the SET data card is described in Section 3.5. Boundary

conditions that vary with time or other parameters use the "b.c. data" parameter to specify an integer "function number" for use with a user supplied subroutine (see Section 3.5).

Notes:

1) The element block identifiers are assigned during the mesh generation operation and must be recalled for use in the element specification. One and only one data card for each element block must appear in the input deck.

2) The boundary condition identifiers are assigned during the mesh generation operation and must be recalled for use in specifying actual boundary data. One and only one data card for each boundary condition group must appear in the input deck.

Appendix D - External Mesh Generator File Format

The file format used by NACHOS II in reading data from an external mesh generator is shown below in a section of FORTRAN code. This is intended only as a summary of the file structure. A detailed description of the GENESIS format is provided in reference [10]. NACHOS II reads data in this format in subroutines `GENRD1, GENRD2` and `GENRD3`.

```
C
C     Heading
C
      CHARACTER*80 HEAD
      READ (NTP0) HEAD
C
C     Problem Sizing Parameters
C
      READ (NTP0) NUMNOD,NDIM,NUMEL,NELBLK,NUMNPS,LNPSNL,NUMESS,
     1            LESSEL,LESSNL,NVERSN
C        NUMNOD - Number of Nodes
C        NDIM   - Number of Coordinates per Node
C        NUMEL  - Total Number of Elements
C        NELBLK - Number of Element Blocks
C        NUMNPS - Number of Nodal Point Sets
C        LNPSNL - Length of the Nodal Point Sets Node List
C        NUMESS - Number of Side Sets
C        LESSEL - Length of the Element Side Sets Element List
C        LESSNL - Length of the Element Side Sets Node List
C        NVERSN - version number of this EXODUS format
C
C     Nodal Point Coordinates
C
      READ (NTP0) ((COORD(I,J),I=1,NUMNOD),J=1,NDIM)
C
C     Optimized Element Order Map
C
      READ (NTP0) (MAP(I),I=1,NUMEL)
C
C     Element Blocks
C
      DO 10 K=1,NELBLK
C
C     Sizing Parameters for this Element Block
C
      READ (NTP0) IDEBLK,NUMELB,NELNOD,NATRIB
```

```
C        IDEBLK - Element block identification (must be unique)
C        NUMELB - Number of elements in this block (the sum of NUMELB
C                 over all blocks must equal NUMEL above)
C        NELNOD - Number of nodes defining the connectivity for an
C                 element of this type
C        NATRIB - Number of element attributes for this element type
C
C     Connectivity for this Element Block
C
      READ (NTP0) ((ICONK(J,I),J=1,NELNOD),I=1,NUMELB)
C
C     Attributes for this Element Block
C
      READ (NTP0) ((ATRIBK(J,I),J=1,NATRIB),I=1,NUMELB)
   10 CONTINUE
C
C     Nodal Point Sets Data
C
C     Nodal Point Set IDs
      READ (NTP0) (IDNPS(I),I=1,NUMNPS)
C     Nodal Point Set Counts
      READ (NTP0) (NNNPS(I),I=1,NUMNPS)
C     Nodal Point Set Pointers
      READ (NTP0) (IPTNPS(I),I=1,NUMNPS)
C     Nodal Point Set Node List
      READ (NTP0) (LSTNPS(I),I=1,LNPSNL)
C     Nodal Point Distribution Factors
      READ (NTP0) (FACNPS(I),I=1,LNPSNL)
C
C     Element Side Sets Data
C
C     Element Side Set IDs
      READ (NTP0) (IDESS(I),I=1,NUMESS)
C     Element Side Set Element Counts
      READ (NTP0) (NEESS(I),I=1,NUMESS)
C     Element Side Set Node Counts
      READ (NTP0) (NNESS(I),I=1,NUMESS)
C     Element Side Set Element Pointers
      READ (NTP0) (IPEESS(I),I=1,NUMESS)
C     Element Side Set Node Pointers
      READ (NTP0) (IPNESS(I),I=1,NUMESS)
C     Element Side Set Element List
      READ (NTP0) (LTEESS(I),I=1,LESSEL)
C     Element Side Set Node List
```

```
      READ (NTP0) (LTNESS(I),I=1,LESSNL)
C     Element Side Set Distribution Factors
      READ (NTP0) (FACESS(I),I=1,LESSNL)
C
```

Appendix E - Post-Processing File Format

The file format used by NACHOS II to write data for an external post-processing file is shown below in a section of FORTRAN code. This is intended only as a summary of the file structure. A detailed description of the EXODUS format is provided in reference [11]. NACHOS II writes ouput data in this format in subroutine GRPHFL. Note that the first section of the EXODUS file is composed of the GENESIS data listed in Appendix D.

```
C
      CHARACTER HEAD*80
C
C     Open the database file
C
      NDB = 11
      OPEN (UNIT=NDB, ..., FORM='UNFORMATTED')
C
C     Header record
C
      WRITE (NDB) HEAD
C
C     Problem sizing parameters
C
      WRITE (NDB) NUMNOD, NDIM, NUMEL, NELBLK, NUMNPS, LNPSNL, NUMESS,
     1            LESSEL, LESSNL, NVERSN
C        NUMNOD - the number of nodes
C        NDIM - the number of dimensions
C        NUMEL - the total number of elements
C        NELBLK - the number of element blocks
C        NUMNPS - the number of node point sets
C        LNPSNL - length of the nodal point sets node list
C        NUMESS - number of side sets
C        LESSEL - length of the element side sets element list
C        LESSNL - length of the element side sets node list
C        NVERSN - version number of this EXODUS format
C
C     Nodal Point Coordinates
C
      WRITE (NDB) ((COORD(I,J),I=1,NUMNOD),J=1,NDIM)
C
C     Optimized Element Order Map
C
      WRITE (NDB) (MAP(I),I=1,NUMEL)
```

```
C
C     Element Blocks
C
      DO 100 K=1,NELBLK
C
C     Sizing Parameters for this Element Block
C
      WRITE (NDB) IDEBLK,NUMELB,NELNOD,NATRIB
C        IDEBLK - Element block identification (must be unique)
C        NUMELB - Number of elements in this block (the sum of NUMELB
C                 over all blocks must equal NUMEL above)
C        NELNOD - Number of nodes defining the connectivity for an
C                 element of this type
C        NATRIB - Number of element attributes for this element type
C
C     Connectivity for this Element Block
C
      WRITE (NDB) ((ICONK(J,I),J=1,NELNOD),I=1,NUMELB)
C
C     Attributes for this Element Block
C
      WRITE (NDB) ((ATRIBK(J,I),J=1,NATRIB),I=1,NUMELB)
C
  100 CONTINUE
C
C     Nodal Point Sets Data
C
C     Nodal Point Set IDs
      WRITE (NDB) (IDNPS(I),I=1,NUMNPS)
C     Nodal Point Set Counts
      WRITE (NDB) (NNNPS(I),I=1,NUMNPS)
C     Nodal Point Set Pointers
      WRITE (NDB) (IPTNPS(I),I=1,NUMNPS)
C     Nodal Point Set Node List
      WRITE (NDB) (LSTNPS(I),I=1,LNPSNL)
C     Point Distribution Factors
      WRITE (NDB) (FACNPS(I),I=1,LNPSNL)
C
C     Element Side Sets Data
C
C     Element Side Set IDs
      WRITE (NDB) (IDESS(I),I=1,NUMESS)
C     Element Side Set Element Counts
      WRITE (NDB) (NEESS(I),I=1,NUMESS)
```

```
C     Element Side Set Node Counts
      WRITE (NDB) (NNESS(I),I=1,NUMESS)
C     Element Side Set Element Pointers
      WRITE (NDB) (IPEESS(I),I=1,NUMESS)
C     Element Side Set Node Pointers
      WRITE (NDB) (IPNESS(I),I=1,NUMESS)
C     Element Side Set Element List
      WRITE (NDB) (LTEESS(I),I=1,LESSEL)
C     Element Side Set Node List
      WRITE (NDB) (LTNESS(I),I=1,LESSNL)
C     Element Side Set Distribution Factors
      WRITE (NDB) (FACESS(I),I=1,LESSNL)
C
C     QA header information
C
      WRITE (NDB) NQAREC
C        NQAREC - the number of QA records (must be at least 1)
C
      DO 110 IQA = 1, NQAREC
      WRITE (NDB) (QATITL(I,IQA), I=1,4)
C        QATITL - the QA title records; each record contains:
C          1) analysis code name (CHARACTER*8)
C          2) analysis code qa descriptor (CHARACTER*8)
C          3) analysis date (CHARACTER*8)
C          4) analysis time (CHARACTER*8)
  110 CONTINUE
C
C     Optional header text
C
      WRITE (NDB) NINFO
C        NINFO - the number of information records
C
      DO 120 I = 1, NINFO
      WRITE (NDB) INFO(I)
C        INFO - extra information records (optional) that contain any
C               supportive documentation that the analysis code
C               developer wishes (CHARACTER*80)
  120 CONTINUE
C
C     Coordinate names
C
      WRITE (NDB) (NAMECO(I), I=1,NDIM)
C        NAMECO - the coordinate names (CHARACTER*8)
C
```

```
C     Element type names
C
      WRITE (NDB) (NAMELB(I), I=1,NELBLK)
C        NAMELB - the element type names (CHARACTER*8)
C
C     History, global, nodal, and element variable information
C
      WRITE (NDB) NVARHI, NVARGL, NVARNP, NVAREL
C        NVARHI - the number of history variable names
C        NVARGL - the number of global variable names
C        NVARNP - the number of nodal variable names
C        NVAREL - the number of element variable names
C
      WRITE (NDB) (NAMEHI(I), I=1,NVARHI), (NAMEGV(I), I=1,NVARGL),
     1            (NAMENV(I), I=1,NVARNP), (NAMEEV(I), I=1,NVAREL)
C        NAMEHI - the history variable names (CHARACTER*8)
C        NAMEGV - the global variable names (CHARACTER*8)
C        NAMENV - the nodal variable names (CHARACTER*8)
C        NAMEEV - the element variable names (CHARACTER*8)
C
      WRITE (NDB) ((ISEVOK(I,J), I=1,NVAREL), J=1,NELBLK)
C        ISEVOK - the name truth table for the element blocks;
C                 ISEVOK(i,j) refers to variable i of element block j;
C                 the value is 0 if and only if data will NOT be output
C                 for variable i for element block j (otherwise the
C                 value is 1)
C
C     Solution data for each time step
C        (time must increase monotonically)
C
      DO 160 ITIME = 1, NSTEP
C
      WRITE (NDB) TIME(ITIME), HIFLAG(ITIME)
C        HIFLAG - the history flag (0 for output of all variables
C                 nonzero for history variables only)
C        TIME - the time step value
C
      WRITE (NDB) (VALHI(IVAR,ITIME), IVAR=1,NVARHI)
C        VALHI - the history values for the time step
C
      IF (HIFLAG(ITIME) .EQ. 0.) THEN
C
      WRITE (NDB) (VALGV(IVAR,ITIME), IVAR=1,NVARGL)
C        VALGV - the global values for the time step
```

```
C
      DO 130 IVAR = 1, NVARNP
      WRITE (NDB) (VALNV(INP,IVAR,ITIME),INP=1,NUMNOD)
C        VALNV - the nodal variables at each node
C                for the current time step
  130 CONTINUE
C
      DO 150 IBLK = 1, NELBLK
      DO 140 IVAR = 1, NVAREL
      IF (ISEVOK(IVAR,IBLK) .NE. 0) THEN
      WRITE (NDB) (VALEV(IEL,IVAR,IBLK,ITIME),IEL=1,NUMELB(IBLK))
C        VALEV - the element variables at each element
C                for the current time step
      END IF
  140 CONTINUE
  150 CONTINUE
C
      END IF
  160 CONTINUE
C
C     End of data is indicated by an end of file
C
      CLOSE (NDB)
C
```