SAND--77-1332-Rev.

DE83 003822

SAND77-1332 (Revised)
Unlimited Release

COYOTE, A FINITE-ELEMENT COMPUTER PROGRAM FOR NONLINEAR HEAT-CONDUCTION PROBLEMS*

David K. Gartling
Fluid Mechanics and Heat Transfer Division 1511
Sandia National Laboratories**
Albuquerque, New Mexico 87185

October 1982

ABSTRACT

COYOTE is a finite element computer program designed for the solution of two-dimensional, nonlinear heat conduction problems. The theoretical and mathematical basis used to develop the code is described. Program capabilities and complete user instructions are presented. Several example problems are described in detail to demonstrate the use of the program.

*This work was sponsored by the U.S. Department of Energy under contract DE-AC04-76DP00789.

**A U.S. Department of Energy facility.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont)

LIST OF FIGURES

LIST OF FIGURES (cont)

PREFACE

At the time of release of the first version of COYOTE in mid-1978, it was not anticipated that the code would receive the heavy usage that it currently enjoys. In response to user needs, COYOTE has undergone several minor updates in the past several years. However, continued requests for additional capabilities combined with significant changes in computer hardware dictated the need for a major upgrading of the code. The present report describes this latest version of the COYOTE program.

In an effort to make the program more flexible and, therefore, more usable a number of capabilities and improvements have been added to COYOTE. The quadratic, quadrilateral, and triangular finite elements have been supplemented by the addition of linear quadrilaterals and triangles to the element library. These elements are computationally inexpensive and are particularly useful in simulating boundary value problems that contain discontinuous or nearly discontinuous behavior, e.g., change of phase. The time integration scheme in COYOTE has been augmented by the addition of a second family of implicit integration methods, the so-called generalized Crank-Nicolson method. Improved flexibility in the modeling of problems has been achieved by expanding the parameter lists on user supplied subroutines. This allows both material properties and boundary conditions to depend on a wider variety of variables. For user convenience commands have been added to permit editing of the printed output as well as the definition of special computation/output points within the mesh. The plot capabilities of the code have also been expanded. Finally, the COYOTE program has been made operational on the Cray Research S-1 computer. This version of COYOTE allows problems of substantially larger size to be analyzed.

Unfortunately, the additions to COYOTE necessitated some small changes in the input and data syntax for the code. The new version of COYOTE is, therefore, not compatible with all of the old COYOTE input. For experienced users of COYOTE, this inconvenience will be slight and of temporary concern.

INTRODUCTION

[illegible] need for the engineering analysis [illegible] is which the transport of thermal energy occurs primarily through a conduction process is a common situation. For all but the simplest geometries and boundary conditions, analytic solutions to heat conduction problems are unavailable, thus forcing the analyst to fall upon some type of approximate numerical procedure. A wide variety of these numerical packages currently exist, ranging in sophistication from the large general purpose code such as CINDA[1] to codes written by individuals for specific problem applications.

The general purpose heat conduction codes, such as CINDA, provide powerful analysis tools for the engineer. However, inherent in their generality is a complexity that often translates into long learning times for potential code users and large man-hour investments in data preparation for problem analysis. These drawbacks are especially acute for the occasional user of the code, as substantial amounts of time may be spent reviewing code operation prior to each use.

The purpose for developing the code described here, COYOTE, was to bridge the gap between the general purpose heat transfer code and the specific problem type code. The COYOTE code is capable of treating a wide variety of transient or steady, linear or non-linear heat conduction problems though it certainly does not have the generality of codes like CINDA. COYOTE is a user-oriented code that has an input organization and format that is easily learned and remembered. Since the numerical method in COYOTE is based on the finite element method, an interface with numerous existing finite element structural mechanics codes is quite straightforward.[2] Finally, the basic input for COYOTE is virtually identical to the formats for the incompressible flow code NACHOS,[3] and the

porous flow code, MARIAH,[1] allowing users to greatly increase their analysis capability by learning a single data input convention.

The basic program organization and many of the user oriented programming features of the present code are derived from the solid mechanics finite element code TEXGAP,[5] developed by Becker and Dunham. Among the programming features developed in the TEXGAP code and incorporated in the present program are:

(1) Command mode input which allows the user to specify the sequence of operations required for each problem.

(2) Free field input that eliminates the need to recall field width and format specification for input variables.

(3) An isoparametric mesh generation scheme that allows complex boundary shapes to be modeled easily and accurately.

(4) A frontal solution technique for processing elements that allows great flexibility in element choice.

(5) An automated coarse to fine grid rezone procedure that allows specific areas of a solution field to be examined in detail without an undue cost in solution time.

In the following sections of this report, a brief description is given of the theory and computational methods used in the COYOTE program. Also, an input guide for use of the program is provided along with several illustrative example problems.

FORMULATION OF THE BASIC EQUATIONS

The development of the finite element equations for transient heat conduction problems is well documented [1-3] and will only be briefly reviewed here. The present formulation is restricted to two-dimensional geometries (i.e., plane or axisymmetric). To simplify the derivation of the equations in the following sections, only the plane two-dimensional problem will be treated in detail. The derivation of the axisymmetric equations follow in a straightforward manner.

Continuum Equations

The appropriate mathematical description of the heat conduction process in a material region Ω is given by

$$\rho C_p \frac{\partial T}{\partial t} - \frac{\partial}{\partial x_i}\left(k_{ij} \frac{\partial T}{\partial x_j}\right) - Q = 0 \quad , \tag{1}$$

where ρ is the material density, C_p the heat capacity, k_{ij} the conductivity tensor, Q the volumetric heat source, t the time, x_i the spatial coordinates, and T the temperature. For the present work, each material is assumed to be homogeneous and either isotropic (i.e., $k_{ij} = k$) or orthotropic (i.e., $k_{ij} = 0$ where k_{ij} is written in terms of the principle material directions). In general, the material properties may be functions of time, spatial location, and temperature; the heat source Q may also depend on time, spatial location, and temperature.

The boundary of the region Ω is defined by $\Gamma = \Gamma_T + \Gamma_q$, where Γ_T and Γ_q are parts of the boundary for which the temperature and heat flux are specified. The relevant boundary conditions for Equation (1) may then be expressed as

$$T = T_b \qquad \text{on} \qquad \Gamma_T \quad , \tag{2}$$

and,

$$q_i n_i + \left(k_{ij} \frac{\partial T}{\partial x_j}\right) n_i + q_c + q_r = 0 \qquad \text{on} \qquad \Gamma_q \quad , \tag{3}$$

where T_b is an applied boundary temperature, q_i is the applied heat flux vector, n_i the unit outward normal to the boundary Γ_q, q_c the heat flux due to convection, and q_r the heat flux due to radiation. Typically, the convective and radiative heat fluxes are given by,

$$q_c = h_c(T - T_c) \quad , \tag{4}$$

$$q_r = h_r(T - T_r) \quad , \tag{5}$$

where h_c and h_r are convective and radiative heat transfer coefficients and T_c and T_r are equilibrium temperatures for which no convection or radiation occurs. The radiation coefficient is given by,

$$h_r = \varepsilon\sigma(T^2 + T_r^2)(T + T_r) \quad , \tag{6}$$

in which ε is the emissivity and σ is the Stefan-Boltzmann constant. Note that the boundary conditions given in Equations (2) and (3) may again be functions of time, spatial location, and temperature.

Equations (1) through (6) provide a complete description of the boundary value problem for the temperature, T. When considering the transient heat conduction problem, a suitable set of initial conditions describing the initial spatial distribution of T is also required. An approximate solution to this class of problems may be obtained by discretization of the continuum problem through use of a numerical procedure such as the finite element method.

Finite Element Equations

The spatial discretization of the above boundary value problem by use of finite elements may be approached by either of two methods. Historically, the first and most popular approach consists of rewriting the boundary value problem

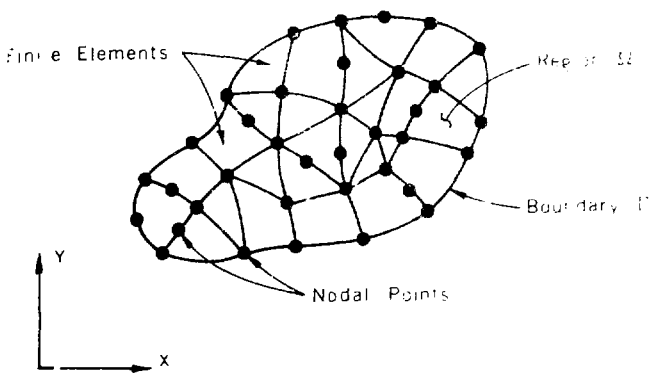


FIGURE 1. Finite Element Discretization of a [illegible]

Assume that within each finite element, the temperature [illegible] approximated by,

$$T(x_i,t) = \sum_{n=1}^{N} \psi_n(x_i)\theta_n(t) \quad ,$$

or in matrix notation,

$$T(x_i,t) = \underset{\sim}{\phi}^T(x_i)\underset{\sim}{\theta}(t) \quad .$$

In Equation (7), ϕ_n is an N dimensional vector of interpolation (shape) functions, θ_n is a vector of nodal point unknowns, superscript T denotes a vector transpose, and N is the number of nodal points in an element. Substitution of Equation (7) into the partial differential equation, Equation (1), and boundary conditions, Equation (3), yields a set of residual equations, due to the approximate nature of Equation (7). The Galerkin method guarantees the orthogonality of the residual vectors to the space spanned by the interpolation functions. This orthogonality is expressed by the inner product,

$$\left\langle \underset{\sim}{\phi}, R \right\rangle = \int_{\Omega_e} \underset{\sim}{\phi} R \, d\Omega = 0 \quad , \tag{8}$$

where R is the residual for the differential equation and Ω_e is the region enclosed by the element.

Carrying out the above operations explicitly for Equations (1), (3), and (7) yields the equation,

$$\int_{\Omega_e} \underset{\sim}{\phi} \left\{ \rho C_p \underset{\sim}{\phi}^T \frac{\partial \underset{\sim}{\theta}}{\partial t} - \frac{\partial}{\partial x_i}\left(k_{ij} \frac{\partial \underset{\sim}{\phi}^T}{\partial x_j} \underset{\sim}{\theta} \right) - Q \right\} d\Omega$$

$$+ \int_{\Gamma_e} \underset{\sim}{\phi} \left(q_i n_i + k_{ij} \frac{\partial \underset{\sim}{\phi}^T}{\partial x_j} \underset{\sim}{\theta} n_i + q_c + q_r \right) d\Gamma = 0 \quad . \tag{9}$$

Equation (9) may be rewritten using Green's theorem (basically an integration by parts on the second order derivative term) to give the equation,

$$\int_{\Omega_e} \rho C_p \underset{\sim}{\phi}\underset{\sim}{\phi}^T \frac{\partial \underset{\sim}{\theta}}{\partial t} \, d\Omega + \int_{\Omega_e} \frac{\partial \underset{\sim}{\phi}}{\partial x_i} k_{ij} \frac{\partial \underset{\sim}{\phi}^T}{\partial x_j} \underset{\sim}{\theta} \, d\Omega$$

$$= \int_{\Omega_e} \underset{\sim}{\phi} Q \, d\Omega - \int_{\Gamma_e} \underset{\sim}{\phi} \left(q_i n_i + q_c + q_r \right) d\Gamma \quad . \tag{10}$$

Several terms in Equation (10) contain material properties which are generally assumed to be functions of time, spatial location, and/or temperature. To accommodate such arbitrary variations within each element, it is convenient to approximate these material functions through a set of interpolation functions. Thus, let ρC_p and k_{ij} be approximated by

$$\begin{aligned} \rho C_p &= \underset{\sim}{\eta}^T \underset{\sim}{\rho C}_p \\ k_{ij} &= \underset{\sim}{\eta}^T \underset{\sim}{k}_{ij} \end{aligned} \tag{11}$$

where $\underset{\sim}{\eta}$ is a vector of interpolation functions and $\underset{\sim}{\rho C}_p$ and $\underset{\sim}{k}_{ij}$ are vectors of nodal point heat capacities and conductivities, respectively. A similar technique may be used to allow the volumetric heat source within an element to have an arbitrary functional dependence. Thus, let

$$Q = \underset{\sim}{\eta}^T \underset{\sim}{Q} \tag{12}$$

where $\underset{\sim}{Q}$ is a vector of nodal point volumetric sources. Substitution of Equations (11) and (12) into Equation (10) produces the following equation

$$\int_{\Omega_e} \underset{\sim}{\eta}^T \underset{\sim}{\rho C}_p \underset{\sim}{\Phi}\underset{\sim}{\Phi}^T \, d\Omega \, \frac{\partial \underset{\sim}{\theta}}{\partial t} + \int_{\Omega_e} \frac{\partial \underset{\sim}{\Phi}}{\partial x_i} \underset{\sim}{\eta}^T \underset{\sim}{k}_{ij} \frac{\partial \underset{\sim}{\Phi}^T}{\partial x_j} \, d\Omega \, \underset{\sim}{\theta}$$

$$= \int_{\Omega_e} \underset{\sim}{\Phi}\underset{\sim}{\eta}^T \underset{\sim}{Q} \, d\Omega - \int_{\Gamma_e} \underset{\sim}{\Phi}(q_i n_i + q_c + q_r) \, d\Gamma \tag{13}$$

Once the form of the interpolation functions, $\underset{\sim}{\Phi}$ and $\underset{\sim}{\eta}$ are specified for an element, the integrals in Equation (13) may be evaluated. Such an evaluation leads to a matrix equation for each element of the following form,

$$\underset{\sim}{M}\dot{\underset{\sim}{\theta}} + \underset{\sim}{K}\underset{\sim}{\theta} = \underset{\sim}{F}_Q + \underset{\sim}{F} \quad , \tag{14}$$

where,

$$\underset{\approx}{M} = \int_{\Omega_e} \underset{\sim}{\eta}^T \rho C_p \underset{\sim}{\phi} \underset{\sim}{\phi}^T \, d\Omega$$

$$\underset{\approx}{K} = \int_{\Omega_e} \frac{\partial \underset{\sim}{\phi}}{\partial x_i} \underset{\sim}{\eta}^T k_{ij} \frac{\partial \underset{\sim}{\phi}^T}{\partial x_j} \, d\Omega$$

$$\underset{\sim}{F}_Q = \int_{\Omega_e} \underset{\sim}{\phi} \underset{\sim}{\eta}^T Q \, d\Omega$$

$$\underset{\sim}{F} = -\int_{\Gamma_e} \underset{\sim}{\phi}(q_n + q_c + q_r) \, d\Gamma \quad .$$

The previous discussion was directed toward the derivation of the equations for a single element. The finite element model for the entire region Ω is obtained through assembly of the element matrices by imposing appropriate inter-element continuity requirements on the dependent variable. Such an assembly (the so-called "direct stiffness" method[9]) yields a matrix equation of the form given in Equation (14).

Discrete Boundary Conditions

As noted previously, boundary conditions for heat conduction problems may be of several types. The discrete form of a specified temperature condition is straightforward. For a temperature specified at a nodal point, the equation for that nodal point is replaced by a constraint condition enforcing the boundary value.

The specification of boundary conditions in terms of various heat fluxes requires slightly more computation. In Equation (14), the boundary fluxes to an element appear in the vector $\underset{\sim}{F}$ as an integral taken along the element boundary (only element boundaries coinciding with Γ need be considered as contributions from interior boundaries are cancelled by adjoining elements). In order to understand the procedure for computation of these boundary integrals, reference must be made to Figure 2 which shows a typical finite element boundary.

Considering first the case of an applied normal heat flux to the element, the contribution to $\underset{\sim}{F}$ is expressed by,

$$\underset{\sim}{F}_n = -\int_{\Gamma_e} \underset{\sim}{\Phi} q_n \, d\Gamma \quad , \tag{15}$$

where the $\underset{\sim}{\Phi}$ functions are restricted to the boundary. If the coordinate along the boundary is s, then

$$d\Gamma = \left[\left(\frac{\partial x_1}{\partial s}\right)^2 + \left(\frac{\partial x_2}{\partial s}\right)^2\right]^{\frac{1}{2}} ds = \Delta \, ds \quad ,$$

and Equation (15) becomes,

$$\underset{\sim}{F}_n = -\int_{-1}^{1} \underset{\sim}{\Phi}(s) q_n(s) \left[\left(\frac{\partial x_1}{\partial s}\right)^2 + \left(\frac{\partial x_2}{\partial s}\right)^2\right]^{\frac{1}{2}} ds \quad . \tag{16}$$

Once the distribution of q_n along the boundary and the shape of the element boundary $x_i(s)$ are known, the computation in Equation (16) is straightforward.

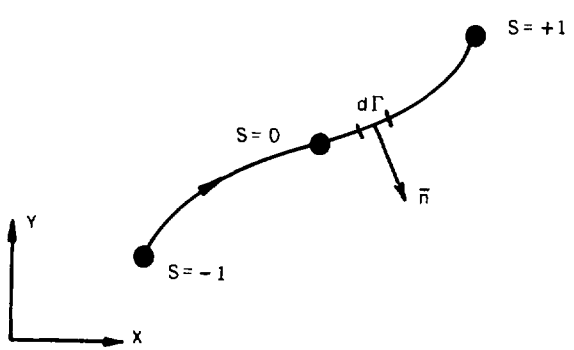


FIGURE 2. Notation for Boundary Integral Analysis

The contribution to $\underset{\sim}{F}$ due to convection is,

$$\underset{\sim}{F}_c = -\int_{\Gamma_e} \underset{\sim}{\Phi} q_c \, d\Gamma = -\int_{\Gamma_e} \underset{\sim}{\Phi} h_c (T - T_c) \, d\Gamma \quad , \tag{17}$$

where the definition in Equation (4) was employed. The temperature along the element boundary is given by,

$$T(s) = \underset{\sim}{\Phi}^T \underset{\sim}{\theta} \quad ,$$

where again $\underset{\sim}{\Phi}$ is a boundary or edge function. Also, since the heat transfer coefficient can be variable, it is convenient to represent it by,

$$h_c(s) = \underset{\sim}{\Phi}^T \underset{\sim}{h}_c \quad .$$

Substituting these relations into Equation (17) and using the definition for $d\Gamma$ produces

$$\underset{\sim}{F}_c = -\int_{-1}^{1} \underset{\sim}{\Phi}^T \underset{\sim}{h}_c \underset{\sim}{\Phi} \underset{\sim}{\Phi}^T \Delta \, ds \, \underset{\sim}{\theta} + \int_{-1}^{1} \underset{\sim}{\Phi}^T \underset{\sim}{h}_c \underset{\sim}{\Phi} T_c \Delta \, ds \quad ,$$

or,

$$\underset{\sim}{F}_c = -\underset{\approx}{C}\underset{\sim}{\theta} + \underset{\sim}{F}_{hc} \quad , \tag{18}$$

where,

$$\underset{\approx}{C} = \int_{-1}^{1} \underset{\sim}{\Phi}^T \underset{\sim}{h}_c \underset{\sim}{\Phi} \underset{\sim}{\Phi}^T \Delta \, ds$$

$$\underset{\sim}{F}_{hc} = \int_{-1}^{1} \underset{\sim}{\Phi}^T \underset{\sim}{h}_c \underset{\sim}{\Phi} T_c \Delta \, ds \quad .$$

Note that in Equation (18), the term $\underset{\approx}{C}\underset{\sim}{\theta}$ contains unknown nodal point temperatures and will thus be moved to the left hand side of the matrix equation in Equation (14).

A computation similar to the one above may be carried out for the radiative flux boundary condition to yield,

$$\underset{\sim}{F}_r = -\underset{\approx}{R}\theta + \underset{\sim}{F}_{hr} \quad , \tag{19}$$

where,

$$\underset{\approx}{R} = \int_{-1}^{1} \underset{\sim}{\Phi}^T h_r \underset{\sim}{\Phi}\underset{\sim}{\Phi}^T \Delta \, ds$$

$$\underset{\sim}{F}_{hr} = \int_{-1}^{1} \underset{\sim}{\Phi}^T h_r \underset{\sim}{\Phi}^T T_r \Delta \, ds \quad .$$

Again, the $\underset{\approx}{R}\theta$ term will be moved to the left hand side of Equation (14) while $\underset{\sim}{F}_{hr}$ is retained on the right hand side. Equation (19) is complicated by the fact that both $\underset{\approx}{R}$ and $\underset{\sim}{F}_{hr}$ are functions of θ since the radiative heat transfer coefficient, h_r (Equation (6)), is an explicit function of temperature.

To summarize the modifications to the basic matrix equation, Equation (1), due to the application of flux type boundary conditions, Equations (16), (18), and (19) may be substituted into Equation (14) to yield,

$$\underset{\approx}{M}\dot{\theta} + \underset{\approx}{K}\theta = \underset{\sim}{F} + \underset{\sim}{F}_Q = \underset{\sim}{F}_n + \underset{\sim}{F}_c + \underset{\sim}{F}_r + \underset{\sim}{F}_Q \quad ,$$

or,

$$\underset{\approx}{M}\dot{\theta} + \underset{\approx}{K}\theta = \underset{\sim}{F}_Q + \underset{\sim}{F}_n - \underset{\approx}{C}\theta + \underset{\sim}{F}_{hc} - \underset{\approx}{R}\theta + \underset{\sim}{F}_{hr} \quad . \tag{20}$$

Rearranging Equation (20) allows the final form of the discrete equation to be written as,

$$\underset{\approx}{M}\dot{\theta} + \underset{\approx}{K}^*\theta = \underset{\sim}{F}^* \quad , \tag{21}$$

with,

$$\underset{\approx}{K}^* = \underset{\approx}{K} + \underset{\approx}{C} + \underset{\approx}{R}$$

$$\underset{\sim}{F}^* = \underset{\sim}{F}_Q + \underset{\sim}{F}_n + \underset{\sim}{F}_{hc} + \underset{\sim}{F}_{hr} \quad .$$

In general, $\underset{\approx}{M}$, $\underset{\approx}{K}^*$, and $\underset{\sim}{F}^*$ may all be functions of $\underset{\sim}{\theta}$, and/or time.

SOLUTION OF THE FINITE ELEMENT EQUATIONS

The discussion of the solution procedures for the matrix equation, Equation (21), naturally divides itself into two sections; transient problems and steady state problems.

Transient Analysis

A large body of literature[9,10,11] is available on possible time integration schemes for equations of the heat conduction type. Both implicit and explicit methods, as well as mode superposition, have been used successfully. In order to efficiently apply typical explicit integration schemes to the heat conduction equation, the capacity matrix $\underset{\sim}{M}$ defined in Equation (21) is replaced with an "equivalent" matrix $\underset{\sim}{M}_D$ which has non-zero coefficients only on the diagonal. This lumping procedure, which expedites the inversion of $\underset{\sim}{M}$ as required in explicit schemes, has been widely studied for use with lower order finite elements (e.g., bilinear quadrilaterals and triangles). Unfortunately, no generally valid procedure is available for constructing diagonal capacity matrices for higher order elements such as the biquadratic quadrilateral. Mode superposition schemes can be very accurate and computationally efficient for linear problems. However, modal analysis requires at least some of the eigenvalues and eigenvectors of Equation (21) to be computed which complicates code structure. Also, for nonlinear analyses, the modal method loses a great deal of its computational efficiency.

The problems listed above were circumvented in the COYOTE program by implementing a family of implicit time integration schemes. Implicit methods in general have the advantage of increased numerical stability which allows the use of a wide range of permissible time steps. This increased stability is obtained at the expense of repetitive solutions of a large matrix problem.

Two basic integration procedures were selected for use in COYOTE, the generalized Crank-Nicolson procedure, which is in fact a family of methods, and a modified Crank-Nicolson algorithm. The generalized Crank-Nicolson scheme when applied to Equation (21) is expressed by[12]

$$\left[\frac{1}{\Delta t}\,\underset{\approx}{M}(\underset{\sim}{\theta}^{o}) + \lambda\underset{\approx}{K}^{*}(\underset{\sim}{\theta}^{o})\right]\underset{\sim}{\theta}^{n+1} = \frac{1}{\Delta t}\,\underset{\approx}{M}(\underset{\sim}{\theta}^{o})\underset{\sim}{\theta}^{n} - (1-\lambda)\underset{\approx}{K}^{*}(\underset{\sim}{\theta}^{o})\underset{\sim}{\theta}^{n} + \lambda\underset{\sim}{F}^{*n+1} + (1-\lambda)\underset{\sim}{F}^{*n} \tag{22}$$

where,

$$\underset{\sim}{\theta}^{o} = \lambda\underset{\sim}{\theta}^{n+1} + (1-\lambda)\underset{\sim}{\theta}^{n} \quad , \quad 0 \leq \lambda \leq 1$$

In writing Equation (22), the general functional dependencies of each matrix and vector have been indicated. The superscript n indicates the time level, Δt is the time step, and λ is a parameter that determines where in the time interval t^n to $t^n+\Delta t = t^{n+1}$ the equation will be evaluated. With specific choices for λ a number of familiar integration schemes are produced: a) $\lambda = 1$ produces a forward Euler or fully implicit method, b) $\lambda = 1/2$ produces the standard Crank-Nicolson method, c) $\lambda = 0$ produces the backward Euler or explicit method, and d) $\lambda = 2/3$ produces a Galerkin method[9] (also known as Donea's method).

When the $\underset{\approx}{M}$ and $\underset{\approx}{K}^{*}$ matrices depend on temperature and $\lambda > 0$, then Equation (22) produces a nonlinear set of algebraic equations that must be iteratively solved at each time step. Various predictor-corrector or extrapolation methods could be used to reduce the work involved in solving Equation (22) at a given step. In the present program, a quasi-linearization scheme is used that avoids altogether the need for iteration during a time step. Setting $\underset{\sim}{\theta}^{o} = \underset{\sim}{\theta}^{n}$ allows all the matrices to be evaluated explicitly and the solution advanced to $\underset{\sim}{\theta}^{n+1}$ in a single step. This assumption is quite reasonable for many conduction problems with thermal shock and some radiation problems being notable exceptions.

The second integration procedure used in COYOTE is a central difference method that is related to the standard Crank-Nicolson scheme. The derivation

of the algorithm is given elsewhere.[12,13] Based on Equation (21), the integration procedure is expressed by,

$$\left[\frac{2}{\Delta t}\,\underset{\approx}{M}(\underset{\sim}{\theta}^{a}) + \underset{\approx}{K}^{*}(\underset{\sim}{\theta}^{a})\right]\underset{\sim}{\theta}^{a} = \underset{\sim}{F}^{*}(\underset{\sim}{\theta}^{a}) + \frac{2}{\Delta t}\,M(\underset{\sim}{\theta}^{a})\underset{\sim}{\theta}^{n} \qquad (23)$$

where,

$$\underset{\sim}{\theta}^{a} = (\underset{\sim}{\theta}^{n+1} + \underset{\sim}{\theta}^{n})/2 \quad .$$

The procedure in Equation (23), termed a modified Crank-Nicolson method, is identical to a standard Crank-Nicolson method ($\lambda = 1/2$ in Equation (22)) except that the solution is not extended to the end of the time interval. This lack of extrapolation accounts for the improved behavior of the solution with respect to temporal oscillations when compared to standard Crank-Nicolson solutions. For nonlinear problems in which $\underset{\approx}{M}$ and $\underset{\approx}{K}^{*}$ are functions of temperature, the algorithm in Equation (23) again produces a set of nonlinear algebraic equations. Iteration within a time step is avoided by quasi-linearization in which $\underset{\sim}{\theta}^{a}$ is set equal to $\underset{\sim}{\theta}^{n}$.

Steady State Analysis

For problems that are independent of time, the basic matrix equation reduces to,

$$\underset{\approx}{K}^{*}(\underset{\sim}{\theta})\cdot\underset{\sim}{\theta} = \underset{\sim}{F}^{*}(\underset{\sim}{\theta}) \quad . \qquad (24)$$

Considering first the case where $\underset{\approx}{K}^{*}$ and $\underset{\sim}{F}^{*}$ are not functions of $\underset{\sim}{\theta}$, then Equation (24) reduces to a linear matrix equation which may be solved directly.

When Equation (24) retains its nonlinear form, an iterative technique is required. The present code uses a simple Picard iteration (successive substitution) method which is expressed by,

$$\underset{\approx}{K}^{*}(\underset{\sim}{\theta}^{n})\cdot\underset{\sim}{\theta}^{n+1} = \underset{\sim}{F}^{*}(\underset{\sim}{\theta}^{n}) \quad , \qquad (25)$$

where superscript n indicates the iteration level. The algorithm in Equation (25) is equivalent to solving a time dependent problem using, for example, Equation (23) with $\Delta t \to \infty$.

PROGRAM DESCRIPTION

The numerical procedure described in the previous section has been implemented in a FORTRAN coded program called COYOTE. In the following sections, a general description of the code is given along with a discussion of some of the computational procedures.

Organization

The organization of COYOTE reflects the steps taken in setting up, solving, and evaluating a finite element analysis of a conduction problem. The code is self-contained with its own mesh generator and plotting packages. COYOTE is written in an OVERLAY form with each overlay handling a specific task in the analysis process. Transmission of data between overlays is through low speed disc files and extended core storage (ECS) as shown in Figure 3.

As indicated by Figure 3, the main overlay acts as a calling routine to the remaining primary overlays. The following list provides a brief description of the functions of each primary overlay.

(1) OVERLAY (1, 0) SETUP -- reads material property data, element and boundary condition data, creates mesh, organizes data for equation formulation.

(2) OVERLAY (2, 0), OVERLAY (10, 0) FORMKF -- reads material property, element and boundary condition data from SETUP, forms coefficient matrices for each element and applies boundary conditions, creates unique "nicknames" for each degree of freedom. Overlay (10, 0) is the axisymmetric version.

(3) OVERLAY (3, 0) ZIPP -- reads element coefficient matrices, assembles global coefficient matrices using "nicknames" to establish element connectivity, solves either transient or steady state matrix equation.

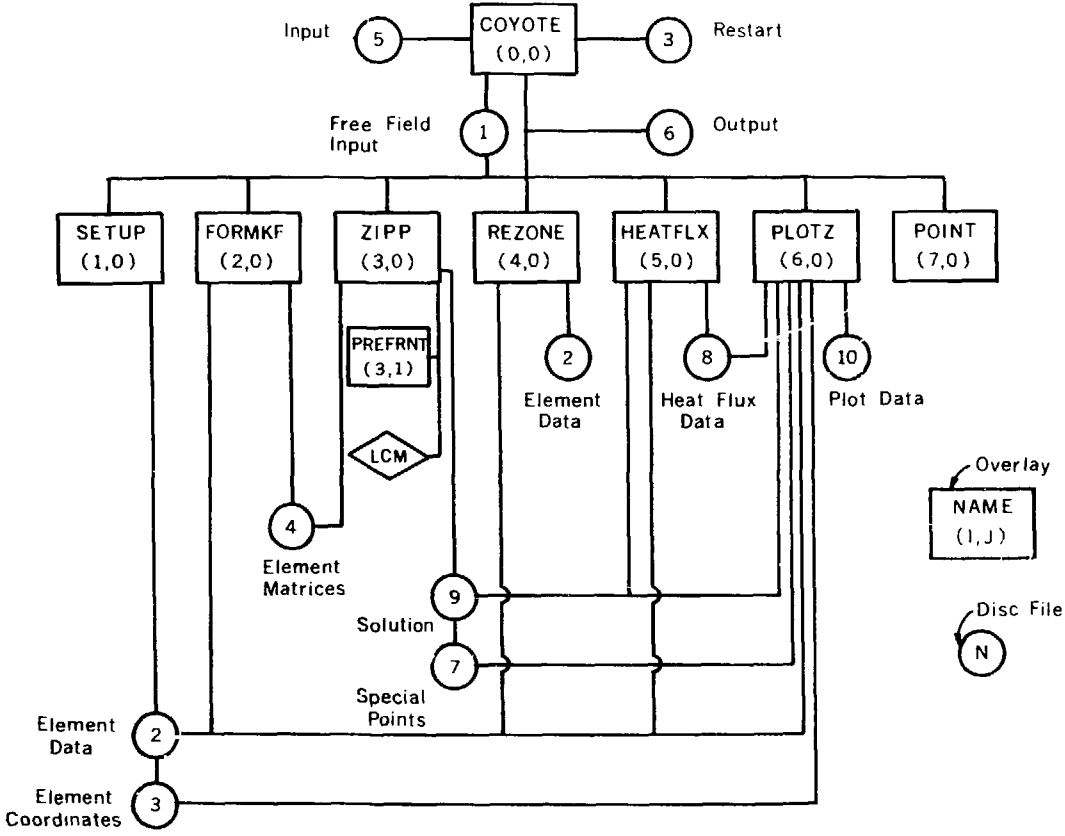


FIGURE 3. Schematic Diagram of COYOTE

(4) OVERLAY (4, 0) RZONE -- refines mesh within a specified region, organizes data for resolution of refined grid.

(5) OVERLAY (5, 0) HEATFLX -- reads temperature solution, computes heat flux quantities for selected elements.

(6) OVERLAY (6, 0) PLOTZ -- plots grid points, elements, isotherms, and temperature histories.

(7) OVERLAY (7, 0) POINT -- determines the location of special output points within the mesh.

COYOTE was originally designed for use on Control Data Corporation (CDC) series 6000 and 7000 computers. The code has also been made operational on the Cray Research Corporation S1 computer. In the Cray version of COYOTE, the OVERLAY organization is replaced by a subroutine structure and data previously stored in ECS is relocated to main memory. The CDC version of COYOTE is presently sized to accommodate grids with up to 500 elements; the Cray version allows up to 1000 elements. Increases in the element capacity of both versions may be made through changes in several dimension statements in the code.

Mesh Generation

The generation of grid points in COYOTE is achieved through an isoparametric mapping technique developed by Womack[14] and used in several previous codes.[3,4,5,15] The generation of nodal points for a particular problem is a distinct operation in COYOTE and independent of the specification of the element connectivity. This separation of operations allows the user to generate more nodes than may actually be used in the problem and to experiment with nodal point placement before element connectivity is established. These options are especially useful when gridding large and/or geometrically complicated problems.

For purposes of grid construction, the domain of interest is considered to be made up of parts or regions which are determined by the user. Within each part, an isoparametric mapping is used to approximate the region boundary. Specification of a number of x, y (or r, z) coordinates on the boundary of each region determines the limits of the region and the type of interpolation used

to define the boundary shape. Complex, curved boundary shapes can be easily and accurately modeled using this scheme which includes linear, quadratic, or cubic interpolation for the boundary shape.

The mesh points within a region are generated automatically once the number of nodes along each boundary is specified. Within each region, an I, J numbering system is used to identify individual grid points. Nodal point spacing within each part may be easily varied by use of a gradient specification. Options also exist for the generation of mesh points along a specified arc or at individual points. A user subroutine may be used to generate mesh points according to a formula supplied by the user.

Element Library

The formulation of the equations for an individual element as indicated by Equation (14) requires specification of shape function vectors for approximation of the temperature field and the element geometry. The form of the shape functions depend on the element being used; COYOTE employs four basic types of elements which are described below.

A. Quadrilateral Elements

The two basic quadrilateral elements used in COYOTE are the straight-sided, four node (QUAD4/4) element shown in Figure 4 and the curved-sided, eight node (QUAD8/4, QUAD8/8) element shown in Figure 5. The temperature field within the four node element is approximated using the bilinear functions given by

$$\underset{\sim}{N}^T = 1/4 \begin{Bmatrix} (1 - s)(1 - t) \\ (1 + s)(1 - t) \\ (1 + s)(1 + t) \\ (1 - s)(1 + t) \end{Bmatrix} \tag{26}$$

where the ordering of the functions corresponds to the ordering of the nodes in Figure 4.

The temperature in the eight node element is represented by the biquadratic functions which can be expressed as,

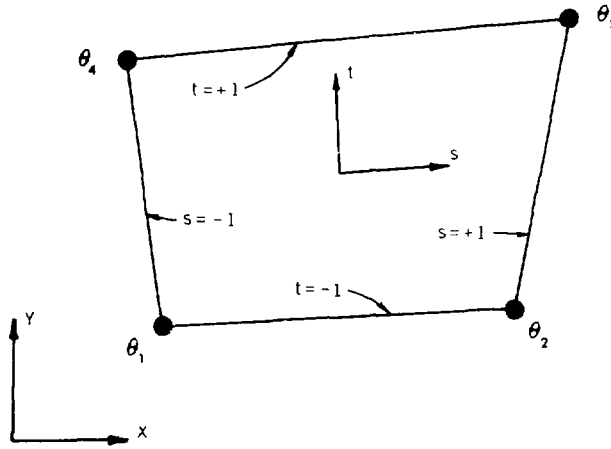


FIGURE 4. Quadrilateral Element

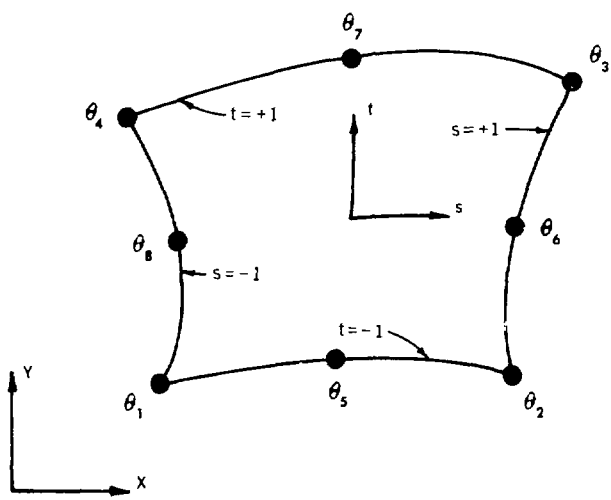


FIGURE 5. Isoparametric Quadrilateral Element

$$\underset{\sim}{M} = \frac{1}{2}\begin{Bmatrix} \frac{1}{2}(1-s)(1-t)(-s-t-1) \\ \frac{1}{2}(1+s)(1-t)(s-t-1) \\ \frac{1}{2}(1+s)(1+t)(s+t-1) \\ \frac{1}{2}(1-s)(1+t)(-s+t-1) \\ (1-s^2)(1-t) \\ (1+s)(1-t^2) \\ (1-s^2)(1+t) \\ (1-s)(1-t^2) \end{Bmatrix} \tag{27}$$

The shape functions in Equations (26) and (27) are expressed in terms of the normalized or natural coordinates for the element, s and t, which vary from -1 to +1 as shown in the figures. The relationship between the x, y (or r, z) coordinates and the natural coordinates is obtained through the isoparametric mapping concept discussed by Ergatoudis, et al.[16] That is, the coordinate transformation from x, y to s, t is represented by,

$$x = \underset{\sim}{N}^T\underset{\sim}{x} \quad ; \quad y = \underset{\sim}{N}^T\underset{\sim}{y} \tag{28}$$

for the case of a straight-sided element (i.e., QUAD4/4 or QUAD8/4) and by,

$$x = \underset{\sim}{M}^T\underset{\sim}{x} \quad ; \quad y = \underset{\sim}{M}^T\underset{\sim}{y} \tag{29}$$

for the case of a curved-sided element (i.e., QUAD8/8). In Equations (28) and (29), $\underset{\sim}{x}$ and $\underset{\sim}{y}$ are vectors of coordinates for points on the element boundary (generally nodal point coordinates).

Using the relations in Equations (26) through (29) and some algebraic manipulation allows the matrix coefficients defined in Equation (14) to be written as integrals of rational functions of the s, t coordinates. The evaluation of these integrals requires the use of numerical quadrature. The COYOTE program employs a 3x3 Gaussian integration for the evaluation of the matrix coefficients for a quadrilateral element.

B. Triangular Elements

The companion elements to the previously described quadrilaterals are the straight-sided, three node (TRI3/3) triangle shown in Figure 6 and the

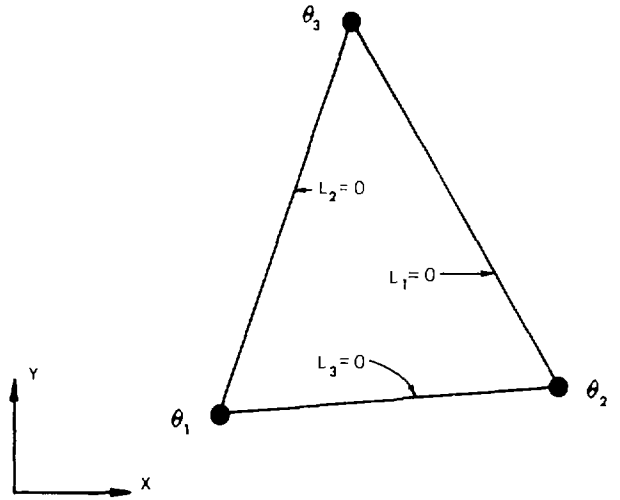


FIGURE 6. Triangular Element

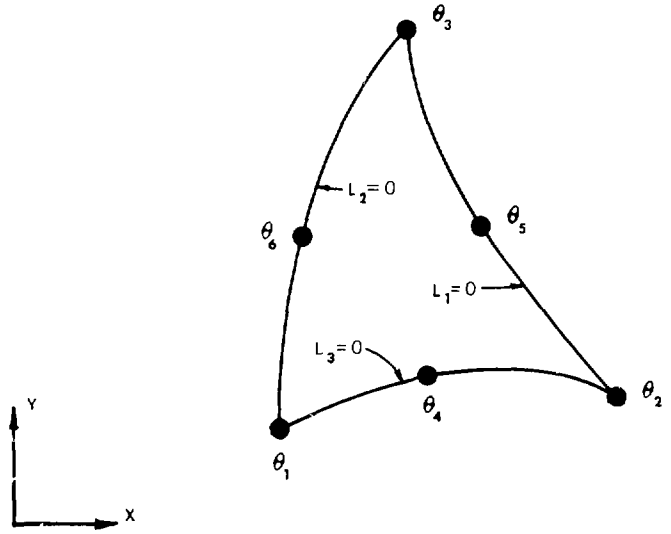


FIGURE 7. Isoparametric Triangular Element

curved-sided, six node (TRI6/3 and TRI6/6) triangle shown in Figure 7. The interpolation functions for these two elements are given by,

$$\underset{\sim}{N} = \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \end{Bmatrix} \tag{30}$$

for the three node element and

$$\underset{\sim}{M} = \begin{Bmatrix} L_1(2L_1 - 1) \\ L_2(2L_2 - 1) \\ L_3(2L_3 - 1) \\ 4L_1L_2 \\ 4L_2L_3 \\ 4L_1L_3 \end{Bmatrix} \tag{31}$$

for the six node element. The ordering of the functions in Equations (30) and (31) corresponds to the ordering of the nodes in Figures 6 and 7. The shape functions are expressed in terms of the natural coordinates for a triangle (see, e.g., Zienkiewicz[9]) where

$$L_1 + L_2 + L_3 = 1 \quad . \tag{32}$$

The coordinate transformations needed to implement the triangular elements directly parallel those for the quadrilateral elements and need not be repeated. Note that when performing the numerical evaluation of the integrals in Equation (14) for a triangular element, the COYOTE program uses a seven point quadrature formula developed by Hammer, et al.[17]

Boundary and Initial Conditions

Boundary conditions are accepted by COYOTE on an element basis. Specification of temperature is possible on both a nodal point and element side basis with the latter type being restricted to having a uniform value along the side. Specified boundary heat fluxes are applied by element side; the flux is assumed

uniform over each element. The adiabatic boundary condition is the "natural" boundary condition for the heat conduction problem and is obtained by default in COYOTE, i.e., no boundary condition is specified for an adiabatic surface. Convective and radiative boundary conditions are also specified by element side, the emissivity, ε, and the equilibrium temperatures T_c and T_r are assumed uniform over an element boundary. Specified temperature and heat flux (including convective fluxes) boundaries may vary arbitrarily in time through the use of user-supplied time functions. A volumetric heat source may be specified for any material as either a constant or as a user specified function of time, spatial location, and/or temperature.

Initial conditions for the solution of transient problems or nonlinear steady state problems may be input by two methods. Through the standard data input, the temperature field may be initialized at a different uniform temperature for each material. Arbitrary initial conditions may be input through a user supplied tape file written in a specified format.

Equation Solution

The solution algorithms used to solve the matrix equations obtained from the finite element discretization were presented in a previous chapter. The actual processing of elements during the solution phase is accomplished by use of the frontal solution technique developed by Irons.[18] The frontal method is a Gaussian elimination procedure that operates with a single element at a time. As each succeeding element is processed, the coefficient matrices for the element are added to the global coefficient matrix. Whenever a particular nodal point unknown is found to be complete (i.e., all the elements contributing to a node have been processed), that particular equation is condensed from the system. This process allows the minimum number of equations to be kept in core storage during the solution process. The condensed equations are stored in Extended Core Storage (ECS) until they are needed in the back-substitution phase. The logic for this elimination scheme is setup by an abstract elimination procedure called the pre-front which occurs just prior to the actual solution. The pre-front coding is located in a secondary overlay.

Rezoning

The rezone feature of COYOTE is a relatively new technique in heat conduction analysis though it has found some previous use in solid and fluid mechanics. For regions in the domain of interest that cannot be finely gridded for reasons of economy or computational limits, the rezone technique provides an efficient method of increasing the local detail of a solution field. Basically, the rezone procedure consists of refining a part of the original coarse grid, applying boundary conditions to this region based on the coarse grid solution and solving the resulting boundary value problem.

This process has been automated in the present code and allows local grid refinement to be made within regions containing quadrilateral elements. The rezone region is specified by establishing a quadrilateral shaped boundary around the region of interest with all of the quadrilateral elements lying entirely within that region considered as part of the rezone. After the region to be rezoned is specified by the user, appropriate temperature boundary conditions are automatically interpolated from the coarse grid solution and applied along interior boundaries of the newly refined grid. The rezone data is organized such that subsequent calls to the element formulation and solution overlays allow computation of the temperature field within the rezone region. The present version of COYOTE only allows rezoning to be carried out for steady state problems.

The decision whether to rezone a region or use an initially refined grid is very problem dependent and no clear guidelines for making this decision are currently available. Until a wider experience with the technique has been obtained, individual user experience and experiment remain the best guides.

Heat Flux Calculations

The calculation of heat flux values for a heat conduction problem is provided in COYOTE as a user option. The calculation procedure employed follows directly from the definition,

$$q_i = -k_{ij} \frac{\partial T}{\partial x_j} \quad , \tag{33}$$

and the finite element approximations.

$$T = \underset{\sim}{\Phi}^T \underset{\sim}{\theta}$$

$$x = \underset{\sim}{N}^T \underset{\sim}{x} \quad ; \quad y = \underset{\sim}{N}^T \underset{\sim}{y} \quad \text{or} \quad x = \underset{\sim}{M}^T \underset{\sim}{x} \quad ; \quad y = \underset{\sim}{M}^T \underset{\sim}{y} \tag{34}$$

where the shape function vectors in Equation (34) are those described in the element library section. Direct substitution of Equation (34) into Equation (33) shows that derivatives of the interpolation functions are required. Since the interpolation functions are in terms of the natural coordinates for an element, the following relations are needed (using the $\underset{\sim}{N}$ functions as an example),

$$\begin{Bmatrix} \frac{\partial \Phi}{\partial x} \\ \\ \frac{\partial \Phi}{\partial y} \end{Bmatrix} = \frac{1}{[J]} \begin{bmatrix} F_{11} & F_{12} \\ \\ F_{21} & F_{22} \end{bmatrix} \begin{Bmatrix} \frac{\partial \Phi}{\partial s} \\ \\ \frac{\partial \Phi}{\partial t} \end{Bmatrix} , \tag{35}$$

where,

$$F_{11} = \frac{\partial N^T}{\partial t} \underset{\sim}{y} \quad ; \quad F_{12} = - \frac{\partial N^T}{\partial s} \underset{\sim}{y}$$

$$F_{21} = - \frac{\partial N^T}{\partial t} \underset{\sim}{x} \quad ; \quad F_{22} = \frac{\partial N^T}{\partial s} \underset{\sim}{x}$$

$$[J] = F_{11} \cdot F_{22} - F_{12} \cdot F_{21} \quad .$$

Using Equation (35), the definition of the heat flux components becomes,

$$q_x = -k_{xx} \left(F_{11} \frac{\partial \Phi^T}{\partial s} \underset{\sim}{\theta} + F_{12} \frac{\partial \Phi^T}{\partial t} \underset{\sim}{\theta} \right) \frac{1}{[J]}$$

$$-k_{xy} \left(F_{21} \frac{\partial \Phi^T}{\partial s} \underset{\sim}{\theta} + F_{22} \frac{\partial \Phi^T}{\partial t} \underset{\sim}{\theta} \right) \frac{1}{[J]} \tag{36}$$

$$q_y = -k_{yx}\left(F_{11}\frac{\partial \underline{\phi}^T}{\partial s}\underline{\theta} + F_{12}\frac{\partial \underline{\phi}^T}{\partial t}\underline{\theta}\right)\frac{1}{|J|}$$

$$-k_{yy}\left(F_{21}\frac{\partial \underline{\phi}^T}{\partial s}\underline{\theta} + F_{22}\frac{\partial \underline{\phi}^T}{\partial t}\underline{\theta}\right)\frac{1}{|J|} \quad .$$

Note that for a triangular element, the s, t coordinates are replaced by the L_1, L_2 coordinates.

With the expressions in Equation (36), the heat flux components may be calculated at any point s_o, t_o within an element once the element geometry (x, y) and temperature field ($\underline{\theta}$) is known. Calculation of fluxes for axisymmetric problems follow a similar procedure. COYOTE provides evaluation of the heat flux at points on the element boundary.

In addition to the heat flux components, COYOTE also calculates the heat flux normal to the element boundary. Referring to Figure 8, the outward normal to an element boundary is,

$$\bar{n} = n_x\bar{e}_x + n_y\bar{e}_y \quad . \tag{37}$$

and the heat flux vector is,

$$\bar{q} = q_x\bar{e}_x + q_y\bar{e}_y \quad . \tag{38}$$

The heat flux normal to the boundary is then,

$$q_n = q_x n_x + q_y n_y \quad , \tag{39}$$

where the flux components are given by Equation (36). From geometry, the components of the normal are,

$$n_x = \frac{\partial y/\partial s}{\Delta} \quad ; \quad n_y = -\frac{\partial x/\partial s}{\Delta} \quad , \tag{40}$$

with,

$$\Delta = \left[\left(\frac{\partial x}{\partial s}\right)^2 + \left(\frac{\partial y}{\partial s}\right)^2\right]^{\frac{1}{2}} \quad .$$

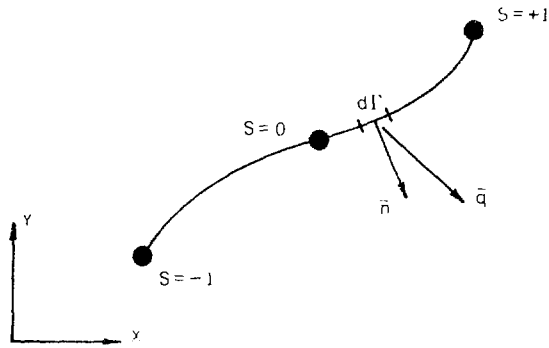


FIGURE 4. Notation for Heat Flux Computation

Use of the definitions of x and y in Equation (34) allows the components of the normal in Equation (40) to be evaluated; the normal heat flux follows directly from the definition in Equation (39). COYOTE also computes the total heat flow for each side of an element by taking the heat flux normal to the boundary times the appropriate surface area.

Plotting

The COYOTE program contains its own plotting package that allows the construction of a variety of plots that are useful in heat transfer analysis. Plots of nodal points, finite element grids and outline plots of the analysis region are generated at the option of the user. Also, isotherms, temperature histories, flux histories, and temperature profiles are available to the user.

INPUT GUIDE

The structure of an input data deck for the COYOTE computer code directly reflects the steps required to formulate and solve a finite element problem. Through the use of a series of command and data cards, the program is directed to such functions as grid generation, element construction, solution of the matrix equations and calculation of auxillary data. 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 command and data cards required by COYOTE are described in roughly the order in which they would normally appear in an input deck. Several example problems follow this section as an added aid to input format.

In the following section, the conventions listed below are used in the description of input cards:

(a) Upper case words imply an alphanumeric input value, e.g., FORMKF.

(b) Lower case words imply a user specified value for the indicated variable, e.g., xmax.

(c) All numerical values are input in a free field format with successive variables separated by commas. All input data is limited to ten characters under this format. WARNING: The CRAY version of COYOTE will read data with ten characters/word, however, only the first eight characters will be used.

(d) [] indicate optional parameters which may be omitted by using successive commas in the variable list. If the omitted parameter is not followed by any required parameters, no additional commas need be specified.

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

(f) The * character may be used to continue a variable list onto subsequent data cards. When the * character is used, it should replace the last comma in the data string.

(g) The $ character may be used to end a data card allowing the remaining space on the card to be used for comments.

(h) The contents of each input card are indicated by underlining.

(i) 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 obtainable from the association of the radial coordinate, r, with x and the axial coordinate, z, with y.

The descriptions of the command cards are presented in the following order:

(a) Header Card

(b) SETUP Command Card

(c) FORMKF Command Card

(d) OUTPUT Command Card

(e) ZIPP Command Card

(f) HEATFLUX Command Card

(g) REZONE Command Card

(h) PLOT Command Card

(i) RESTART Command Card

(j) Program Termination Card

Following the individual descriptions of the command cards are sections discussing input deck structure, user subroutines, initial conditions, error messages, and computer requirements for COYOTE.

Header Card

The header card must be the first card in a deck for any particular problem. If two or more problems are run in sequence, the header card for each new problem follows the END, PROBLEM card of the previous problem. A $ symbol must appear in Column 1; the remaining 79 columns are available for a problem title. The header card is of the following form:

```
$ PROBLEM TITLE
```

SETUP Command Card

The first task in formulating a finite element analysis of a problem involves the specification of the material properties and the definition of element mesh and boundary conditions for the problem geometry. These functions are accomplished through the SETUP command and its three sets of associated data cards.

The SETUP command card has the following form:

SETUP, [iprint], [maxi], [order], [grid plot]

where,

iprint <2>: determines the amount of printout produced during the setup operation. Output increases with the value of iprint and ranges from no printout for iprint = 1 to full printout for iprint = 4.

maxi <18>: is the maximum number of I rows to be generated in the grid. Maxi need only be specified if there are more than 18 I rows or more than 110 J rows. The limit on the maximum I's and J's is I*J $\leq$ 4000 for the CDC version, and I*J $\leq$ 5000 for the Cray version.

order< >: determines the numbering of the elements. For the default value (i.e., order left blank), the elements are numbered by increasing I, J values (e.g., (1,1), (2,1), (3,1) ... (1,2), (2,2) ...). For order = PRESCRIBED, the elements are numbered according to their order in the input list. The element ordering should be chosen such that the front width of the problem is minimized.

grid plot<>: determines if a grid point plot tape is written. If a plot of the grid points is to be made in a subsequent call to the plot routine, then grid plot = PLOT; if no plot of the grid points will be made, the grid plot parameter is omitted.

SETUP Data Cards

Following the SETUP command card, three sets of data cards are required for material property specification, grid point generation, and element and boundary condition specification. Each of the data sets is terminated by an END card. The last END card (i.e., the third), ends the setup portion of the program and readies the program for the next command card.

Material Data Cards:

The material data cards are of the following form:

[Material Name], number, ρ, C_p, k_{11}, k_{22}, β,

temperature dependence, Q, initial temperature

⋮

END

where,

Material Name: is an optional alphanumeric material name for user reference.

number: is the internal reference number for the material. COYOTE will accept up to ten materials, $1 \leq \text{number} \leq 10$.

ρ: is the material density.

C_p: is the material specific heat.

k_{11}, k_{22}<k_{11}>: are components of the material conductivity tensor (see below).

β<0^o>: is the angle in degrees between the principle material axis and the coordinate axis (see below).

temperature dependence <CONSTANT>: prescribes the dependence of the material properties on temperature, time, and/or spatial location. If all material properties (ρ, C_p, k_{ii}) are constant, this parameter is omitted or set to CONSTANT. If one or more of the properties are variable, this parameter is set equal to VARIABLE.

Q<0>: prescribes the volumetric heat source for the material. For no volumetric heating, this parameter is omitted: for constant volumetric heating, the parameter is set to the heating value. If the heat source varies with time, spatial location, or temperature, this parameter is set equal to VARIABLE.

initial temperature <0>: specifies the value of the initial temperature for the material.

The material models allowed in COYOTE include homogeneous materials with either an isotropic or orthotropic conductivity tensor. For isotropic materials ($k_{ij} = k$), the conductivity is specified by setting $k_{11} = k$; the parameters k_{22} and β are omitted. For orthotropic materials ($k_{ij} = \delta_{ij}k_{ij} = k_{ii}$), the conductivity tensor is determined by specifying components of the tensor with respect to the principle material axes. Referring to Figure 9, the 1 and 2 axes indicate the principle material axes while β specifies the orientation of the material with respect to the spatial reference frame. COYOTE requires k_{11} and k_{22} to be specified for an orthotropic material; β must be specified only if the material axes are not aligned with the coordinate axes. Note that β is measured from the positive x(r) axis and is positive in a clockwise direction.

The variation of material properties and volumetric heating with time, space and/or temperature is indicated by setting the parameters "temperature dependence" and "Q" equal to VARIABLE, respectively. The actual variation of these quantities is specified by several user supplied subroutines. If variable material properties have been indicated, COYOTE requires SUBROUTINE CAPACIT and SUBROUTINE CONDUCT to be supplied by the user; a variable heat source requires SUBROUTINE SOURCE. Further details on these subroutines are given in a later section of this chapter.

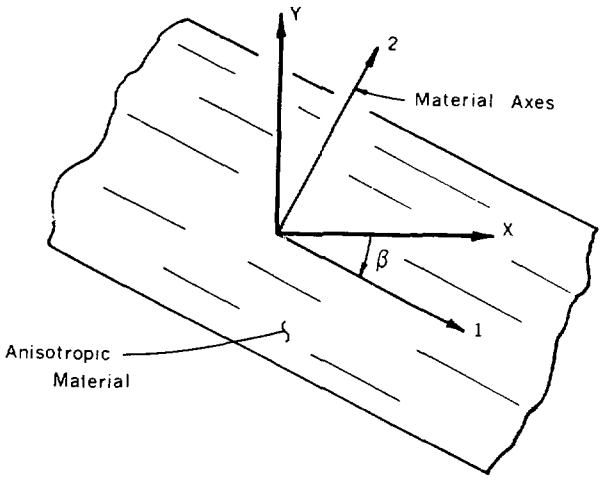


FIGURE 9. Notation for Orthotropic Conductivity

COYOTE does not contain 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 A.

Grid Data Cards:

Following the material specification, the grid points for the finite element mesh are generated. In contrast to many finite element codes, the COYOTE code 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 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 I, J axes must form a right handed coordinate 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. These ideas are more

clearly fixed through a description of the data cards required to generate points in a part.

For each part or region in the mesh, three data cards of the following form are required:

imin, jmin, imax, jmax, [g1], [g2], [g3], [g4], [POLAR], [xo], [yo]

x1, x2, x3, x4, [x5], [x6], ... [x12]

y1, y2, y3, y4, [y5], [y6], ... [y12]

⋮

END

where,

imin, jmin, imax, jmax: are the I, J limits for the region being generated as shown in Figure 10a. The difference between the maximum and minimum values determines the number of grid points generated in a particular direction.

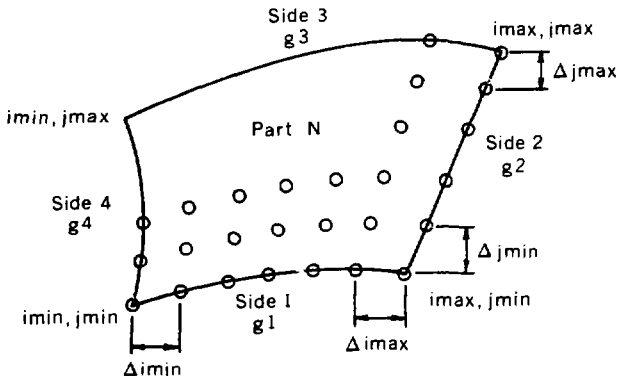
g1<1>, g2<1>, g3<1>, g4<1>: specify the gradients for the node spacing along the four sides of the region. The gradients are defined by,

$$g1 \text{ or } g3 = \frac{\text{node spacing at imin}}{\text{node spacing at imax}} = \frac{\Delta imin}{\Delta imax}$$

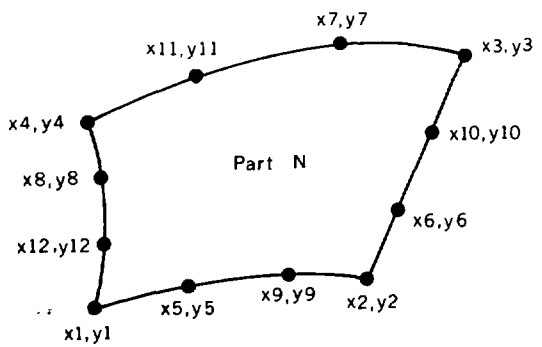
$$g2 \text{ or } g4 = \frac{\text{node spacing at jmin}}{\text{node spacing at jmax}} = \frac{\Delta jmin}{\Delta jmax}$$

and are illustrated in Figure 10a. The default values of unity give equal node spacing along a side. Gradients either larger or smaller than unity may be used to bias the spacing in either direction.

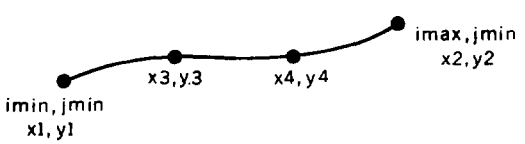
POLAR, xo<0>, yo<0>: specify the use of polar coordinates for describing the coordinates of the points defining the region. With POLAR specified, the x's and y's on the second and third data cards are interpreted as polar radii and angles referred to the local origin xo, yo. The polar angle is referenced to the positive x axis; positive angles are measured in a counterclockwise direction.



(a) I, J Nomenclature



(b) Coordinate Nomenclature



(c) ARC Nomenclature

FIGURE 10. Grid Point Generation Nomenclature

x1, x2, ... x12:
y1, y2, ... y12: define the coordinates of the four corner and optional side nodes 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 node, as shown in Figure 10b must be specified. If one side node is defined, a quadratic interpolation is used to define the boundary; specification of two side nodes allows a cubic interpolation. Side nodes should be located near the midside and third points for quadratic and cubic interpolation, respectively.

There are no limits on the number of parts which may be used to define a grid. The only restriction on the total number of grid points is that max (I*J) $\leq$ 4000 for the CDC version of COYOTE and (I*J) $\leq$ 5000 for the Cray version.

Node points may be generated for a triangular shaped region by allowing two adjacent corners of the quadrilateral part to coincide. However, when triangular meshes are created in this manner, the location of interior node points is generally unpredictable. The user is advised to verify the quality of such a mesh through use of a node point plot.

In generating mesh points for complex geometries, it is often convenient to be able to position an individual node point or a line of nodes. These situations are provided for in COYOTE by use of the following data cards:

POINT, i, j, x1, y1, [POLAR], [xo], [yo]

where,

i, j: is the I, J name for the point.

x1, y1: are the coordinates for the point.

POLAR, xc<0>, yc<0>: specify the use of polar coordinates as described previously.

ARC, imin, jmin, imax, jmax, [g1], [POLAR], [xo], [yo]

x1, x2, [x3], [x4]

y1, y2, [y3], [y4]

where,

imin, jmin, imax, jmax: are the I, J limits for the arc as shown in Figure 10c. Since a one-dimensional array of points is being generated either imin = imax or jmin = jmax. The difference between the maximum and minimum values determines the number of grid points generated along the arc.

g1<1.0>: specifies the gradient for the node spacing along the arc. The gradient is defined by,

$$g1 = \frac{\text{node spacing at ijmin}}{\text{node spacing at ijmax}} = \frac{\Delta ijmin}{\Delta ijmax}$$

and is illustrated in Figure 10c.

POLAR, xo<0>, yo<0>: specify the use of polar coordinates as described previously.

x1, x2, x3, x4:
y1, y2, y3, y4: define the coordinates for the ends of the arc and the optional intermediate points. If the arc is a straight line, only the first two coordinates need be specified. The generation of a curved arc requires the specification of one or two intermediate points as shown in Figure 10c.

There are no limits on the number of POINT and ARC data cards that may be used in generating a mesh. Both types of cards may appear at any point within the grid point data section of the SETUP command.

In certain circumstances, it is convenient to specify nodal point locations according to a given formula or a specific pattern that is not available in the basic mesh generator. These situations are provided for in COYOTE through use of a user supplied subroutine. This subroutine is accessed by the following data card:

EXTDEF

This data card causes the user supplied subroutine EXTDEF to be called by the mesh generator. Within this subroutine, the user may create an arbitrary array of I, J labeled nodal points. The EXTDEF data card may be used in conjunction with the standard nodal point generation schemes or may be used to create an entire mesh. The data card may appear at any appropriate point within the grid point data section of the SETUP command. Details on the form of SUBROUTINE EXTDEF are given in a later section.

Element and Boundary Condition Data Cards:

Following the generation of the grid points, the program is ready to accept element and boundary condition data. Since the mesh points for the problem geometry are generated independently of the elements, the selection of nodes from which to construct a given element is very flexible. The actual construction process for an element consists of identifying an appropriate group of previously generated mesh points that will serve as the corner and midside nodes of the element. This concept is apparent from the form of the element data card

element type, mat, i1, j1, [i2], [j2], ... [in], [jn]

where,

element type: is an alphanumeric name for the type of element. The element types used in COYOTE are described below.

mat: is the material number for the element. This number should be set to correspond to the material number used on the material property card.

i1, j1, [i2], [j2], ...: 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 11. 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 following values for the remaining nodes are assumed,

4 Node Element (QUAD4/4):

$$i2 = i3 = i1 + 1,\ i4 = i1$$
$$j2 = j1,\ j3 = j4 = j1 + 1$$

8 Node Element (QUAD8/4, QUAD8/8):

$$i4 = i8 = i1,\ i5 = i7 = i1 + 1,\ i2 = i3 = i6 = i1 + 2$$
$$j2 = j5 = j1,\ j6 = j8 = j1 + 1,\ j3 = j4 = j7 = j1 + 2 \quad .$$

When I, J values are specified for only the corner nodes of an eight node quadrilateral, the midside I, J values are computed as the average of the corner values. All of the appropriate I, J values for a triangular element must be specified explicitly.

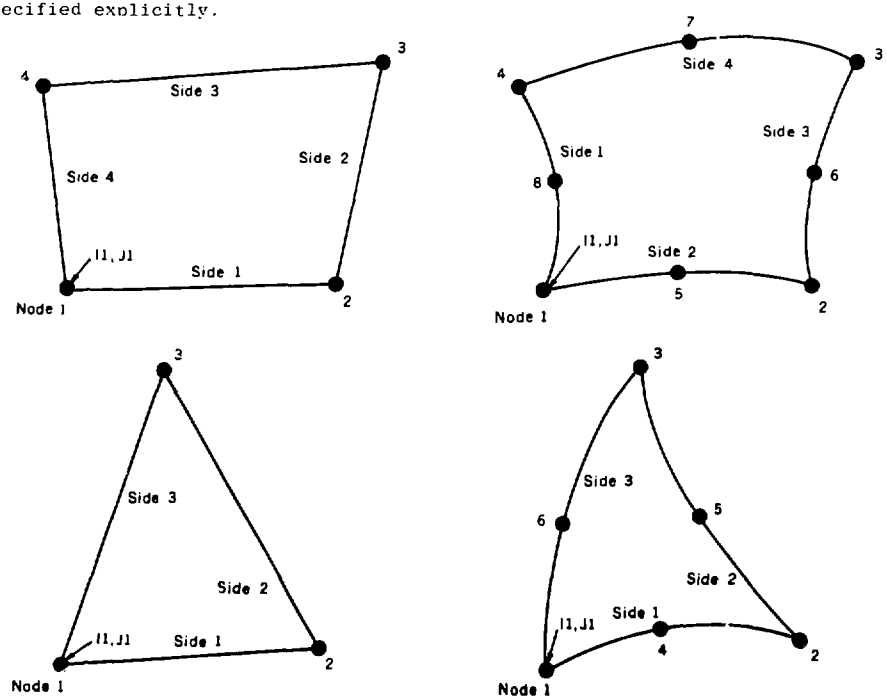


FIGURE 11. Element Node Numbering and Side Numbering

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:

(a) QUAD4/4: a four node (isoparametric) quadrilateral with arbitrarily oriented straight sides.

(b) QUAD8/4: An eight node (subparametric) quadrilateral with arbitrarily oriented straight sides.

(c) QUAD8/8: an eight node (isoparametric) quadrilateral with curved sides.

(d) TRI3/3: a three node (isoparametric) triangle with arbitrarily oriented straight sides.

(e) TRI6/3: a six node (subparametric) triangle with arbitrarily oriented straight sides.

(f) TRI6/6: a six node (isoparametric) triangle with curved sides.

Note that in the generation of the eight and six node subparametric elements, the physical coordinates (x, y) of the midside nodes need not lie precisely on the element side as these coordinates are not used in the element formulation.

During the construction of the element mesh, two points about the I, J identification scheme should be noted. Each element is identified internally by the I, J values for the first node named on the element data card, i.e., il, jl. Since any corner node may be named first, two or more elements may share the same internal 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. Secondly, during the generation of grid points, 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 non-continuous I, J labels is illustrated in the example section.

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, il, jl, side/node, value/set no./time curve no.

where,

b.c. type: is an alphanumeric name for the type of boundary condition. The types of boundary conditions used in COYOTE are described below.

il, jl: 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: 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 11.

value/set no./time curve no.: is the numerical value of the applied boundary condition, the number of the boundary condition SET in the case of convective or radiative boundary conditions or the number of the time history curve for time dependent boundary conditions. For specified temperature or heat flux boundary conditions, this parameter is set to the numerical value of the boundary condition (e.g., 100.0^{o}C or 50 watts/m^{2}). The specification of a set or time curve number is explained below.

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

(a) T: specifies the temperature at a node.

(b) TSIDE: specifies the temperature to have a constant value along an element side.

(c) TVARY: specifies the temperature along an element side to be a function of time.

(d) QSIDE: specifies a constant heat flux (energy/unit area/unit time) along an element side.

(e) QVARY: specifies a time dependent heat flux along an element side.

(f) QCONV: specifies a constant convective heat transfer process along an element side.

(g) QRAD: specifies a radiative/convective heat transfer process along an element side.

(h) Adiabatic Surface: no boundary condition need be applied.

All of the above boundary conditions can be employed with any of the previously described elements. Note that in the present version of COYOTE, only one QRAD. QCONV, QVARY, and TVARY boundary condition can be specified for any single element; other boundary condition types are not restricted.

The boundary condition options, TVARY and QVARY, permit temperatures or heat fluxes along an element boundary to vary with time. The time histories of the boundary condition are input through a set of user supplied subroutines (SUBROUTINE CURVEn). The association of a particular time history with the appropriate boundary condition is accomplished by use of the "time curve no." which appears on the boundary condition data card and in the user supplied subroutine name. COYOTE will allow a total of six different time histories to be input. The format for the time history subroutines is described in a subsequent section.

The use of convective and radiative boundary conditions requires the appropriate heat transfer coefficients (h_c, h_r) and reference temperatures (T_c, T_r) to be specified. These parameters are input through use of a SET data card which is of the form,

SET, b.c. type, set no., h, T

where,

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

set no.: is the number of the particular data set. The code allows up to ten data sets for each type of boundary condition. The set no. is used to identify the appropriate boundary condition set on the BC card.

h, T: are the parameters required for specifying a convective or radiative boundary condition. For convection, h = heat transfer coefficient = h_c and T = T_c. For fourth power radiation, h = emissivity·Stefan-Boltzmann constant = $\varepsilon \cdot \sigma$ and T = T_r. For generalized radiation/convection, h = VARIABLE and T = T_r.

The generalized radiation/convection condition mentioned above has been provided to allow an arbitrary variation of h with temperature or time in the flux expression,

$$q = h(T, t)(T - T_r) \quad .$$

To incorporate such a boundary condition, the parameter "b.c. type" must be set to QRAD, "h" must be set to VARIABLE, "T" is set to the appropriate reference temperature, and the user supplied subroutine, HTCOEF, which describes h(T, t) must be appended to the COYOTE program. This subroutine is described in detail in a later section of this chapter.

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 COYOTE. 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 form:

ILOOP, npass, inc	or	JLOOP, npass, inc
⋮		⋮
element or boundary condition data cards		element or boundary condition data cards
⋮		⋮
IEND		JEND

where,

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

inc: specifies the increment to be added to the I or J values found within the loop. The "inc" parameter may be negative.

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

FORMKF Command Card

With the completion of the SETUP command, the next task in the analysis sequence is the formulation of the matrix equations for the individual elements in the mesh. This function is carried out by the command card,

FORMKF, [geometry]

where,

geometry: is an alphanumeric name to indicate the type of coordinate system desired. If geometry is omitted, the two-dimensional planar formulation is used; if geometry = AXISYM, the axisymmetric formulation is used.

OUTPUT Command Card

Prior to solution of the heat conduction problem, it is sometimes convenient to limit the amount of printed output generated by COYOTE. It may also be desirable to specify spatial locations within the mesh, other than nodal points, at which temperature output is required. Both of these functions may be invoked through use of the OUTPUT command card. The selective printing of the computed temperature field is set up with the following command card:

OUTPUT, delimiter type, n1, n2, n3, ... n50

where,

delimiter type: is an alphanumeric name to indicate how the subsequent element numbers are to be interpreted in limiting the printed output. If delimiter type = SINGLE, individual element numbers are assumed to be listed in the following data. If delimiter type = STRING, the following data is interpreted as pairs of element numbers with each pair defining a sequence of elements.

n1, n2, ... n50: is a list of element numbers indicating which elements are to have temperature data printed during the solution process. A maximum of fifty individual element numbers or twenty-five element pairs may be specified on a single command card.

There is no restriction on the number of OUTPUT command cards of this form that can be used in a COYOTE input deck; OUTPUT cards with different delimiter types may be mixed in the same input deck. The OUTPUT command card(s) must preceed the ZIPP command card in the input deck. If no OUTPUT command is used in a COYOTE input deck, the entire temperature field is printed at each output interval.

The specification of special temperature output points is achieved with the following form of the command card:

OUTPUT, POINTS, x_1, y_1, x_2, y_2, ... x_{25}, y_{25}

where,

x_1, y_1, ... x_{25}, y_{25}: is a list of the x, y (or r, z) coordinate locations for the special points. A maximum of twenty-five points may be specified on a single command card.

COYOTE allows a maximum of fifty special points to be specified during any particular analysis. The OUTPUT command card in this form must proceed the ZIPP command card; OUTPUT command cards of both types may occur in the same input deck. The temperatures at the special output points are printed at each normal output interval. The temperatures at these points are also stored for possible later plotting as temperature histories.

ZIPP Command Card

The assembly and solution of the global system of matrix equations is carried out by the ZIPP command card. For steady state problems, this command is of the following form:

ZIPP, STEADY, [no. iter], [iprint], [tol], [initial conditions]

END

where,

no. iter <1 or 5>: specifies the number of iterations to be used in solving the steady state problem. The default values of 1 and 5 correspond to iteration limits for linear and nonlinear problems, respectively.

iprint <no. iter -1>: specifies how often the solution field is printed.

tol <0.0>: specifies a convergence tolerance that may be used to terminate the iteration. When the largest difference in temperature between successive iterations falls below the specified tolerance, the iteration process is terminated. All nodes in the mesh are checked for adherence to this criteria.

initial conditions < >: specifies the source of the initial estimate for the temperature field. If this parameter is omitted, the initial estimate for the temperature is set from data on the material property cards. If this parameter is set to TAPE, the initial temperature field may be input through a tape file as explained below.

The solution of steady state problems requires the specification of only one ZIPP command; for compatibility with the transient solution option this command is terminated by an END card.

The solution of transient problems is carried out by a command card of the following form:

ZIPP, integration type, [λ], [iprint], $t_{initial}$, [t_{final}],

Δt, no. steps , [initial conditions], [lumping]

⋮

END

where,

integration type: is an alphanumeric name to indicate what type of integration process is to be used. When integration type = TRANS1, the modified Crank-Nicolson procedure (Equation (23)) is selected. For integration type = TRANS2, the generalized Crank-Nicolson family of integration procedures (Equation (22)) are selected.

λ<.5>: specifies the weighting parameter for the generalized Crank-Nicolson methods (TRANS2). For λ = 1.0, a forward Euler method is produced, λ = .5 produces a Crank-Nicolson method, and λ = 0 produces a backward Euler method. The λ parameter may have any value in the range $0 \leq \lambda \leq 1.0$. This parameter is omitted for the TRANS1 integrator.

iprint <10>: specifies how often the solution field is printed.

$t_{initial}$, t_{final}, Δt: specify the time limits ($t_{initial}$, t_{final}) and the time step (Δt) for the time integration procedure.

no. steps: indicates the number of time steps to be taken in the transient analysis.

initial conditions: specifies the source of the initial conditions for the problem. If this parameter is omitted, the initial temperature fields are set from data on the material property card. If this parameter is set equal to TAPE, initial conditions may be input through a tape file as explained below.

lumping: specifies if the capacitance matrix is to be diagonalized (lumped). If this parameter is omitted, a consistent capacitance matrix is used; when lumping is set to LUMP, the capacitance matrix is diagonalized. This option is only valid when using the three node triangular element and/or the four node quadrilateral element.

In performing the analysis of transient problems, one or more ZIPP cards may be required depending on the need to change time step (Δt) or printout frequency (iprint) during the analysis. When using a sequence of ZIPP cards, all information pertinent to the continued analysis of the problem must be specified on cards following the first ZIPP card.

The control of the integration interval is accomplished by either of two methods--specification of a final time (t_{final}) or specification of the number of time steps to be taken (no. steps). When the running time reaches t_{final} or the specified number of time steps is equaled, the program checks for the presence of another ZIPP command and the definition of a new integration interval. This process continues until an END command is encountered, thus ending the integration process.

When the initial condition parameter is set to TAPE, COYOTE expects the initial temperature field to be supplied from a disc (or tape) file, called TAPE 19. Details on the format for this file are given in a later section of this chapter.

The choice of an appropriate time step for a transient problem is often a critical point in the analysis for reasons of computational economy and accuracy. A procedure for estimating a meaningful integration time step, based on problem boundary conditions and the element mesh, is outlined in Appendix B.

HEATFLUX Command Card

To aid in interpreting and using the computed temperature solution, COYOTE allows the computation of several heat flux quantities on an element basis. The computation of heat fluxes is initiated by the following command card,

HEATFLUX, time step no., location

where,

time step no.: is the number of the time step for which heat flux computations are desired. For steady state problems, this parameter is omitted. If time step no. = ALLTIMES, heat fluxes are calculated for every element at every time step.

location: specifies where the heat flux calculations are to be made. For location = FULL, heat flux calculations are made for every element in the mesh. A second option allows fluxes to be calculated in up to twenty elements specified by the user. This latter option is specified by listing the required element numbers after the "time step no." parameter. This parameter is omitted if time step no. = ALLTIMES.

Within a given element, heat flux values are calculated on the element boundary midway between nodes. Calculations are made for the x and y (or r and z) components of the flux vector, the heat flux normal to the element boundary, and the heat flux integrated over each side of the element (i.e., total energy transferred across the element boundary). When using the HEATFLUX command in conjunction with a transient analysis, note that the time steps are numbered continuously from 1 to n beginning with the initial conditions (i.e., solution at time = Δt is step number 2). Heat flux calculations can thus be made at any particular time step by appropriately setting the "time step no." parameter.

Note that the use of the "time step no. = ALLTIMES" option does not result in any heat flux data being printed by COYOTE. This option should be used only to create the data file for the subsequent plotting of flux histories (see HISTORY option under the PLOT command). There is no limit to the number of HEATFLUX command cards that can be used in a COYOTE input deck.

REZONE Command Card

The automatic coarse to fine mesh rezoning of a coarse grid solution is initiated through the command card,

REZONE, [xmin, ymin, xmax, ymax], ni, nj, [iprint], [imin, jmin, imax, jmax]

where,

xmin, ymin, xmax, ymax: specify the x, y coordinate limits on the region to be rezoned. Only elements entirely within this region are rezoned.

ni, nj: specify the number of fine grid elements to be placed in each coarse grid element in the I and J directions. ni and nj must be less than 10.

iprint <3>: specifies the amount of output produced by the rezone procedure. iprint may vary from 1 (limited output) to 4 (full output).

imin, jmin, imax, jmax: specify the limits of the region to be rezoned in terms of I, J coordinates. Only elements entirely within the region bounded by these limits are rezoned.

The region to be rezoned may be specified in terms of either the physical coordinates of the mesh (imax, jmax, etc.) or the I, J coordinates of the mesh (imin, jmin, etc.). Following a REZONE command, the program should be directed to the next suitable operation such as FORMKF or PLOT. The REZONE command only provides mesh refinement and interpolation of new boundary conditions and not automatic re-solution. The present version of COYOTE only allows time independent problems to be rezoned.

PLOT Command Card

The COYOTE program contains a plotting package to facilitate the interpretation of data obtained from a solution and to aid in setting up element meshes. There are six basic types of plots which are available in COYOTE and are obtained with the following command card,

PLOT, plot type, xmin, ymin, xmax, ymax,

[imin, jmin, imax, jmax], [xscale], [yscale], [number], [special pts]

where,

plot type: is an alphanumeric name which specifies the type of plot desired. The permissible parameter values are catalogued below.

xmin, ymin, xmax, ymax: specify the range of coordinates for the area to be plotted in a line drawing or define the range of the ordinate and abscissa for a graph. For line drawings (e.g., element plots, outline plots, contour plots) the xmin, ... ymax parameters define a rectangular window; 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 may be used to set the maximum and minimum values for the ordinate (ymin, ymax) and abscissa (xmin, xmax) of the plot. If these parameters are omitted, the axes for the graph are set by COYOTE.

imin, jmin, imax, jmax: is an optional specification of the limits of the region to be plotted. If the I, J limits of the region are specified, the xmin, ... ymax parameters are used to set the border for the plot. These parameters are omitted for graph plots.

xscale <1.0>, yscale <1.0>: specify magnification factors for the x and y coordinates of the plot. The default values produce a correctly proportioned plot of the largest possible size consistent with the plotting device. The use of a scale parameter produces a non-proportional plot. The use of these parameters is illustrated in the example problems. These parameters are omitted for graph plots.

number <>: specifies if the element numbers are to be displayed on the element plot. If number is omitted, element numbers are not plotted; if number = NUMBER, the element numbers are plotted at the element centroid.

special pts <>: specifies if the location of the special output points (see OUTPUT command) are to be displayed on the element plot. If special pts is omitted, special point locations are not plotted; if special pts = SPECIAL, the special point numbers are plotted at the appropriate location on the element plot.

The parameter on the PLOT command card, denoted "plot type" may be set to any of the following values as required.

(a) POINTS: generates a plot of the grid points generated by the SETUP command. The "grid plot" parameter on the SETUP command card must be set to PLOT to obtain this type of plot.

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

(c) CONTOUR: generates contour plots of the temperature.

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

(e) HISTORY: generates time history plots of the temperature or heat flux.

(f) PROFILE: generates plots of temperature versus position with time as a parameter.

Following the command cards for the CONTOUR, HISTORY, and PROFILE plot options, a series of data cards are required.

Contour Data Cards:

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

```
contour type, time step no., no. contours, [c1, c2, ... c20]
:
END
```

where,

contour type: is an alphanumeric name indicating the variable to be contoured. This parameter must have the value ISOTHERMS.

time step no.: is the number of the time step for which a plot is required. For steady state problems, this parameter may be omitted.

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

c1, c2, ... c20: are optional values that specify the value of the contour to be plotted. If these parameters are omitted, the plotted contours are evenly spaced over the interval between the maximum and minimum values of the variable.

Note that when the contour values are left unspecified, the maximum and minimum values used to compute the contour levels are those for the entire mesh. This procedure may produce an unsatisfactory (or blank) plot in the event only a small portion of the mesh is plotted and the number of contours specified is small. This situation may be avoided by specifying the contour values to be plotted. When using the contour option with a transient analysis, note that the time steps are numbered continuously from 1 to n beginning with the initial conditions (i.e., solution at time = Δt is step number 2). A contour plot can thus be made at any particular time step by appropriately setting the "time step no." parameter.

Any number and/or type of contour data cards may follow a CONTOUR command card; the sequence is terminated by an END card. To simplify the plotting of a

series of contour plots for a transient analysis, a looping feature is available. The looping command has the following form,

PLOTLOOP, no. plots, plot inc

contour type, time step no., no. contours, [c1, c2, ... c20]

PLOTEND

⋮

END

where,

no. plots: specifies the number of plots to be generated.

plot inc: indicates the frequency at which a plot is generated, i.e., every "plot inc" time steps a contour plot is produced beginning with the "time step no." indicated on the contour data card.

There is no limit to the number of PLOTLOOP data sets that may follow a CONTOUR command card; the sequence is terminated by an END card. Within any given PLOTLOOP, only a single contour data card may be defined. PLOTLOOP's and individual contour data card may be mixed under a single CONTOUR command.

History Data Cards:

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

location, no. points, time step 1, time step 2

elem no., node no., elem no., node no., ...

⋮

END

where,

location: is an alphanumeric name indicating the variable to be plotted as a function of time. For location = TLOCATION, nodal point

temperatures are plotted versus time; location = QLOCATION implies that heat fluxes (normal component) are to be plotted; location = TSPECIAL allows temperatures at the special output points (see OUTPUT command) to be plotted.

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

time step 1, time step 2: indicate the time step numbers at which the time histories are to begin and end, respectively. The maximum difference allowed between time step 2 and time step 1 is 400, i.e., only 400 time steps may be represented on a given plot.

elem no., node no.: 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 is allowed. When plotting histories for special output points, pairs of numbers are not required. In this latter case, the numbers of the special points to be plotted are listed in sequence. A maximum of ten such numbers per data card is allowed.

There is no restriction on the number or type of LOCATION data cards that may follow a HISTORY command card; the sequence is terminated by an END card. The numbering of the element nodal points is shown in Figure 11. Note that the numbering of fluxes in an element differs from the node numbering convention; fluxes are numbered sequentially (counterclockwise) around the element beginning with the point nearest the identifying node for the element. Also, the plotting of flux histories assumes that the HEATFLUX, ALLTIMES command was executed prior to the command for the history plot.

Profile Data Cards:

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

TIMEPLANE, no. planes, t1, t2, ... t10

location, delimiter type, no. pairs, elem no., side/node no.,
elem no., side/node no., ...

⋮

END

where,

no. planes: specifies the number of profiles (at different times) to be plotted. A maximum of ten profiles per plot is allowed.

t1, t2, ... t10: specify the time step numbers at which the profile is to be plotted. For steady state problems, this data is omitted.

location: is an alphanumeric name indicating the variable to be plotted as a function of position. Location should be set to TLOCATION.

delimiter type: is an alphanumeric name indicating how the profile geometry is to be specified. If delimiter type = SIDES, the profile geometry is described in terms of element sides. For delimiter type = NODES, the profile is given in terms of individual nodal points.

no. pairs: specifies the number of data pairs (elem no., side/node no.) needed to describe the profile geometry. A maximum of 20 such data pairs is permitted.

elem no., side/node no.: are pairs of numbers specifying the profile geometry in terms of an element number and a side or node number.

There is no restriction on the number of TIMEPLANE/LOCATION data cards that may follow a PROFILE plot command. However, TIMEPLANE and LOCATION data cards must occur in pairs with the TIMEPLANE card first. The data sequence for the PROFILE option is terminated by an END card.

With the delimiter type set to SIDES, the temperature profile is plotted along element sides with the spatial distance being measured along the element boundary. When delimiter type is set to NODES, the temperature profile is plotted by constructing the straight line path between successive nodes. In

both options, the input sequence of elements determines the profile path. There is no requirement that the profile have a continuous path, i.e., some elements along a path may be omitted. Multiple side/node specifications for an individual element are permitted. When the SIDE option is used, the profile path is directed along element sides with the direction of increasing path length determined by increasing corner node numbers. 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. If the path direction is inappropriate with this option, the NODES description may be used.

RESTART Command Card

The COYOTE program allows a computed solution and its associated problem data to be conveniently saved for further post-processing through the use of a restart command. In order to save a previously computed solution, the following command card may be used,

RESTART, SAVE

In order to restart a previously saved solution, the following command card is used,

RESTART, RESET, [nsteps]

where,

nsteps <0>: is a parameter to indicate at what time step number the solution file is to be positioned at during the restart process. If nsteps is omitted, the solution file is rewound. For nsteps > 0, the solution file is positioned after the nsteps time step. This allows restarting of the solution process with the nsteps + 1 time step. This parameter has no meaning for steady state problems.

The command to save a solution may occur at any point after the ZIPP command sequence, that is, after a solution has been obtained. In order to restart a solution, the RESET command should immediately follow the header card.

If the solution is to be continued in time following a restart, the FORMKF command must be executed prior to the ZIPP command as the restart procedure does not save the matrix equations for the problem. Also, the initial condition parameter on the ZIPP command card should be set equal to TAPE. The solution file should be re-positioned at the appropriate time step through use of the nsteps parameter.

The RESTART commands when executed from the data deck, direct the program to collect (or distribute) pertinent element and solution information onto (or from) two files, TAPE13 and TAPE19. To complete the restart process, these files must be saved (or attached) by the appropriate system control cards. Typically, these files would be saved on a magnetic tape or catalogued on a permanent file. The restart procedure is illustrated in the example problems section.

Program Termination Card

There are two modes of termination for a COYOTE analysis. If two or more problems are to be run in sequence, then the appropriate termination for any particular problem is,

END, [iprint]

However, if the program is to be terminated with no further computational operations, then the following command card is used,

STOP, [iprint]

where,

iprint: is an optional alphanumeric parameter that allows the printing of the last solution obtained to be suppressed. If iprint is omitted, the solution field is printed; if iprint = NOPRINT, then printing is suppressed.

Input Deck Structure

As noted previously, the order of the commands to COYOTE is 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 POINTS plot option must follow the SETUP command sequence as the file used to store the grid point coordinates is rewritten in later operations. Typically, a grid point plot is used in setting up a grid and is not generated during a complete solution sequence.

(b) The ELEMENTS and OUTLINE plot options can be located anywhere after the SETUP command sequence.

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

(d) The OUTPUT command must occur after the SETUP command and must precede the ZIPP command.

(e) The RESTART, SAVE command can only be executed after a solution has been obtained. No provisions are made in the restart process to retain the element coefficient matrices and thus the SAVE option has little meaning prior to obtaining a solution.

(f) If a solution is to be saved, the RESTART, SAVE command should generally be the last command (except for STOP) in a data deck. The restart process uses several tape files that are also employed by other program operations. The execution of commands following a RESTART, SAVE command could result in a conflict in file usage.

(g) If a transient solution has been saved, it may be continued in time by use of the RESTART, RESET command. The RESTART command should be followed by the FORMKF command and the ZIPP commands with the TAPE parameter indicated.

User Supplied Subroutines

There are a number of situations which require the user to supply FORTRAN coded subroutines to COYOTE; the definition of special nodal point locations, the use of variable material properties, the use of a variable volumetric heat source, the specification of a general radiation/convection boundary condition and the use of time dependent temperature or flux boundary conditions.

When the parameter EXTDEF is encountered during the generation of nodal points, COYOTE accesses SUBROUTINE EXTDEF which is supplied by the user. This subroutine allows nodal point locations to be defined in any appropriate manner by the user. This subroutine must have the following form:

```
SUBROUTINE EXTDEF (X, Y, MAXI)
DIMENSION X(MAXI, 1), Y(MAXI, 1)

  .
  .
  .

FORTRAN coding to generate an array of
   nodal point locations

  .
  .
  .

RETURN

END
```

where the variables in the subroutine parameter list are

X, Y: two-dimensional arrays containing the x, y (or r, z) coordinate locations of the nodal points. The indices for each two-dimensional array correspond to the I, J name for the nodal point (i.e., X(I, J), Y(I, J) for point I, J).

MAXI: an integer specifying the largest I value that can be defined in the mesh. This parameter corresponds to the maxi parameter specified on the SETUP command card.

When the "temperature dependence" parameter on the material data card is set equal to VARIABLE for any material in the problem, COYOTE expects SUBROUTINE CONDUCT and SUBROUTINE CAPACIT to be supplied by the user. For steady state problems, only SUBROUTINE CONDUCT need be supplied. These two subroutines allow the user to conveniently specify the conductivity and heat capacity (actually $\rho \cdot C_p$) as arbitrary functions of the temperature. In the event only one of the two indicated properties is to vary with temperature, the remaining property must still be set through use of the appropriate subroutine, i.e., COYOTE expects both subroutines to be present. Each subroutine is used to evaluate the appropriate material property for <u>all</u> materials labeled as temperature dependent on the material data cards.

The required subroutines must have the following forms:

```
SUBROUTINE CAPACIT (RHOCP, T, X, Y, NNODES, MAT, NELEM, TIME)
DIMENSION RHOCP (1), T(1), X(1), Y(1)

  .
  .
  .

FORTRAN coding to evaluate ρ·Cp for each material with
  temperature dependent capacitance

  .
  .
  .

RETURN

END
```

and,

```
SUBROUTINE CONDUCT (COND1, COND2, T, X, Y, NNODES, MAT, NELEM, TIME)
DIMENSION COND1(1), COND2(1), T(1), X(1), Y(1)

  .
  .
  .

FORTRAN coding to evaluate k11 and k22 for each material with
  temperature dependent conductivity

  .
  .
  .

RETURN

END
```

where the variables in the subroutine parameter lists are:

RHOCP (NNODES): an array containing the values of $\rho \cdot C_p$.

COND1 (NNODES), COND2 (NNODES): Arrays containing the values of the principle conductivities k_{11} and k_{22}. For isotropic materials, set COND2 (NNODES) = COND 1 (NNODES).

T (NNODES): an array containing the values of the temperatures at the nodes.

X (NNODES), Y (NNODES): arrays containing the values of the x, y (or r, z) coordinates for the nodes.

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

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

NELEM: an integer specifying the element number.

TIME: the current value of the time.

When the volumetric heating parameter, Q, on the material data card is set equal to VARIABLE, COYOTE expects SUBROUTINE SOURCE to be supplied by the user. This subroutine must have the following form:

```
SUBROUTINE SOURCE (QVALUE, T, X, Y, NNODES, MAT, NELEM, TIME)
DIMENSION QVALUE(1), T(1), X(1), Y(1)

  .
  .
  .

FORTRAN coding to evaluate the volumetric heat source for
   each material with a variable heat source

  .
  .
  .

RETURN

END
```

where the variables in the subroutine parameter list are:

QVALUE (NNODES): an array containing the values of the volumetric heating rate.

T (NNODES): an array containing the values of the temperatures at the nodes.

X (NNODES), Y (NNODES): arrays containing the values of the x, y (or r, z) coordinates for the nodes.

NNODES: an integer parameter specifying the number of nodes at which the volumetric heating is to be evaluated.

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

NELEM: an integer specifying the element number.

TIME: the current value of the time.

The use of a generalized radiation/convection boundary condition requires that the variation of the heat transfer coefficient with temperature and/or time be specified. When the "h" parameter on the SET data card is specified as VARIABLE, the COYOTE program expects SUBROUTINE HTCOEF to be supplied by the user. The required subroutine has the following form:

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

where the variables in the subroutine parameter list are:

HT: the heat transfer coefficient.

TSURF: the local surface temperature of the material.

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

XSURF, YSURF: the x, y (or r, z) coordinates on the surface.

TIME: the current value of the time.

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

NELEM: an integer specifying the element number.

Note that if more than one generalized radiation/convection boundary condition occurs in a problem, SUBROUTINE HTCOEF is used for the evaluation of all heat transfer coefficients that vary with time and/or temperature. The set no. (ISET) parameter is used to distinguish the different boundary conditions.

When the boundary condition options TVARY and/or QVARY are employed, COYOTE expects the appropriate time history subroutines to be supplied by the user. The correspondence between a particular boundary condition and its variation with time is established through the "time curve no." which appears on the BC data card and in the name of the subroutine providing the time history. The required subroutines have the following form:

```
SUBROUTINE CURVEn (NELEM, TSURF, TIME, BCVALUE)
:
FORTRAN coding to evaluate a boundary condition
   for a specified time history
:
RETURN
END
```

where the variables are:

n: an integer specifying the "time curve no." as set on the BC data card ($1 \leq n \leq 6$).

NELEM: an integer specifying the element number.

TSURF: the local surface temperature of the material.

TIME: the current value of the time.

BCVALUE: the value of the boundary temperature or heat flux (as required) for the current TIME.

The subroutines described above, when required by COYOTE, should follow the main overlay in the loading sequence. The control cards necessary to implement user supplied subroutines are described in a subsequent section.

Initial Conditions

The analysis of a transient conduction problem requires the specification of a set of initial conditions for the problem.* For cases where the initial temperature field may be assumed uniform (or at least constant over each material), the initial conditions may be set through a parameter on the material data card. For problems where the initial temperature field is more complex, COYOTE accepts the initial conditions from an external storage device (disc or magnetic tape file) denoted TAPE19.

In order to be compatible with COYOTE, the initial temperature field must be written to unit 19 with the following unformatted FORTRAN write statement,

```
WRITE(19) TIME, TMAX, TMIN, NUMEL,
((T(J,I), I = 1,8), J = 1, NUMEL)
```

where,

TIME: is the initial time.

TMAX, TMIN: are the maximum and minimum temperatures in the field.

NUMEL: is the number of elements in the mesh (the CDC version of COYOTE requires that NUMEL $\leq$ 500, the Cray version requires NUMEL $\leq$ 1000).

T(J,I): is the array containing the initial temperatures.

Note that the I index has an upper limit of 8 which corresponds to the number of nodes in a quadrilateral element. If triangular elements are present in the mesh, the I index must still run to 8; the last two entries in the T array are ignored by COYOTE when a triangular element is encountered. Ordering of the nodal point temperatures for each element must be as shown in Figure 11.

*Initial estimates for the temperature are also necessary for steady state problems when radiation boundary conditions are used.

When the file containing the initial conditions is attached to the job, the file name must be TAPE19. It should be noted that the output format for COYOTE is the same as the above; the output file for COYOTE is also TAPE19. Thus, solution fields from COYOTE may be used directly as initial conditions for subsequent problems. The use of these features are demonstrated in the next chapter.

Error Messages

COYOTE has been supplied with a number of error checks and tests for bad or inconsistent input data, overflow of storage, etc. When an error is encountered, a message is printed indicating in what subroutine the error occurred, the type of error, and any pertinent data associated with the error; the program is terminated with a STOP 1 if the error is fatal. The error messages imbedded in COYOTE are listed below according to overlay along with brief explanations of the error.

(0,0) OVERLAY

DRIVER-UNRECOGNIZED COMMAND: a command instruction was expected but was not found or was misspelled.

FFLD-END OF DATA: an end of file was encountered on the input file, check termination command.

FFLDSB-INPUT VARIABLE TOO LONG: an input variable with more than ten characters was encountered.

RESTART-UNRECOGNIZED COMMAND: a restart command was used with an incomplete or misspelled parameter list.

PRINTER-UNRECOGNIZED COMMAND: an output command was used with an incomplete or misspelled parameter list.

(1,0) OVERLAY

ELDATA-UNRECOGNIZED COMMAND: erroneous element, boundary condition, or looping specification, check spelling.

ELDATA-LOOP PREVIOUSLY DEFINED: error in looping specification, check for third loop within two existing loops.

ELDATA-MAXIMUM NUMBER OF ELEMENTS EXCEEDED: reduce number of elements used or re-dimension code.

ELDATA-UNRECOGNIZED DATA ON SET CARD: a set data card was used with an incomplete or misspelled parameter list.

ELDATA-BC APPLIED TO AN UNDEFINED ELEMENT: a boundary condition was applied to an element that has not yet been defined.

ELDATA-EXCESSIVE BOUNDARY CONDITIONS ON ELEMENT: too many boundary conditions (i.e., >6) have been applied to an element.

ELDATA-UNRECOGNIZED BOUNDARY CONDITION: boundary condition type is in error, check spelling.

ELDATA-BC ON IMPROPER SIDE OF ELEMENT: a boundary condition was specified for an improper side of an element ($1 \leq$ side ≤ 4).

NMESH-IJ MAX OR MIN EXCEEDS SPECIFIED VALUE: during generation of the grid points, a part was found with an imax, jmax, imin, or jmin that exceeded the specified maxi on the SETUP command.

MATREAD-TOO MANY MATERIALS SPECIFIED: more than ten materials were specified.

(2,0) and (10,0) OVERLAY

FORMKFP-BAD ELEMENT JACOBIAN: a negative element area was found, check element coordinates and connectivity.

TRI-ZERO JACOBIAN: a triangular element with a zero area was found, check element coordinates.

QUAD-ZERO JACOBIAN: a quadrilateral element with a zero area was found, check element coordinates.

(3,0) OVERLAY

ZIPP-UNRECOGNIZED COMMAND: a solution command was used with an incomplete or misspelled parameter list.

ZIPP-MAXIMUM STORAGE EXCEEDED: maximum active storage exceeded, reduce problem size or re-dimension code.

ZIPP-COMMAND CARD MISSING: another solution command was expected but not found, check for termination command.

ZIPP-ZERO PIVOT: zero on the diagonal of the equation being eliminated, equation system is singular or element connectivity is in error.

BCTIME-TIME CURVE NUMBER TOO LARGE: a time curve number greater than six was encountered, check data cards and time history subroutine names.

PREFRNT-INSUFFICIENT STORAGE: insufficient storage for element connectivity, reduce problem size or re-dimension code.

PREFRNT-INSUFFICIENT STORAGE FOR FRONT WIDTH: reduce problem size or re-dimension code.

(4,0) OVERLAY

REZONE-TOO MANY SUBDIVISIONS OF COARSE ELEMENT: more than ten fine grid elements per coarse grid element have been specified.

REZONE-TOO MANY COARSE GRID ELEMENTS IN REZONE: more than one hundred coarse grid elements have been found in the rezone region.

REZONE-EMPTY REZONE REGION: no coarse grid elements were found in the rezone region.

(6,0) OVERLAY

PLOTZ-UNRECOGNIZED COMMAND: a plot command was used with an incomplete or misspelled parameter list.

CONTOUR-UNRECOGNIZED COMMAND: error in contour specification, check spelling on contour type, looping command, and termination.

TIMEPLT-UNRECOGNIZED COMMAND: error in history plot specification, check spelling and termination.

SPLOT-UNRECOGNIZED COMMAND: error in profile plot specification, check spelling and termination.

SPLOT-TIMEPLANE DATA CARD NOT FOUND

Computer Requirements and Control Cards

COYOTE is a fairly large code (approximately 6000 source statements) that operates most effectively on a large computer system. The code was originally developed on a CDC 6600 computer with an extended core storage (ECS) capability. The present version of COYOTE is designed primarily for execution on the CDC CYBER 76 computer system and the Cray S1 computer. The following discussion of computer request parameters and control cards is specifically directed toward the use of COYOTE on the Sandia National Laboratories computer system.

The central processor time needed to run COYOTE is directly dependent on the number of elements in the mesh and to a lesser extent on the front width of the problem. To roughly estimate the CPU time required for a typical job, the following formula may be used for the CDC version of COYOTE:

$$\begin{aligned} \text{CPU} = (\text{Number elements}) &\times (0.02) + (\text{Number elements}) \\ &\times (\text{Number time steps or iterations}) \\ &\times (0.007) \quad . \end{aligned}$$

Note that the constants in the above formula are accurate for problems of moderate size but could increase by a factor of two for very large analyses. Also, the above estimate of CPU time does not include the time for post-processing the solution, i.e., computing heat fluxes, plotting, etc., nor does it include the time charged for unoverlapped I/O operations. In terms of CPU time, the Cray version of COYOTE executes approximately twice as fast as the CDC version.

The amount of Large Core Memory (LCM) needed for COYOTE (CDC) is also directly related to the number of elements in the mesh and the problem front width. Unfortunately, the LCM required cannot be estimated a priori. The experience of the user is, thus, the best guide for setting this parameter. After completion of a solution, COYOTE reports the total LCM required for the

job allowing the user to adjust the requested value for future runs. No LCM request is needed for COYOTE (Cray).

The COYOTE program is maintained in an UPDATE form in the Sandia National Laboratories permanent file system and may be accessed by executing the following control cards:

```
PFGET, FLNAME, COYOTE, AU = DKGARTL.      (CDC Version)
```

and,

```
FILE, FLNAME, RT = S.
PFGET, FLNAME, COYOTEC, AU = DKGARTL.     (Cray Version)
```

Before executing, the file attached with the above control cards must be run through the UPDATE processor and then compiled. In the following sections, control card decks are listed for using COYOTE in its standard modes of operation.

Run with Plotting:

The control card sequence shown in Figure 12a allows the user to run COYOTE (CDC) and plot results using the RSCORS software package. The graphical output is post-processed and returned through a Versatec plotter at the appropriate Remote Job Entry Terminal (RJET). Other options for processing and disposing of plotted output are given in References 19 and 20. To access the Cray version of COYOTE, the control stream in Figure 12b should be executed.

Run with Restart:

Figure 13a shows a control card stream to run COYOTE (CDC) and save the solution for later restarting. In this case, the solution and element data were saved on magnetic tapes; permanent file storage could also have been used. The listing in Figure 13a is also suitable for restarting a job if the STAGE commands are modified to cause files TAPE13 and TAPE19 to be PRE staged. The control cards needed to run COYOTE (Cray) with restarting are shown in Figure 13b.

Run with Initial Conditions:

The control stream shown in Figure 13a are also suitable for running a transient analysis with initial conditions provided from an external source. If the initial conditions are written on TAPE19 (in the format discussed previously), then the listing in Figure 13a is immediately applicable if the request for TAPE13 is deleted. The control stream for the COYOTE (Cray) version of this procedure is shown in Figure 13b.

Run with User Subroutines:

The control cards required to execute COYOTE (CDC) when user supplied subroutines are to be included are shown in Figure 14a; the equivalent COYOTE (Cray) control stream is shown in Figure 14b.

In Figures 12b, 13b, and 14b, the execution of COYOTE (Cray) is carried out through use of a set of procedure files (SLTLIB[21]). The use of these procedures simplifies the access to COYOTE on the Cray though in fact, the basic Cray Job Control Language can also be used to run the program.

```
COYOTE, TXXXX, ECYYY.                                   NAME      BOX NO.
USER .....
PFGET, FLNAME, COYOTE, AU=DKGARTL.
UPDATE, P=FLNAME, N, F.
FTN, I=COMPILE, L=0.
* PFGET, RSCOR76, AU=SLTHOMP.
* PFGET, RSCDI76, AU=SLTHOMP.
* LIBRARY, RSCOR76, RSCDI76.
RFL, L=YYY.
LGO, PL=777777B.
* PFGET, POP76, AU=SLTHOMP.
* POP76, TAPE10, POPOUT, HC1.
* COMQ, POPOUT, HC1.
7-8-9
7-8-9
COYOTE DATA
6-7-8-9
```

NOTES:

(1) For large jobs producing large quantities of output, consideration should be given to using a FICHE, OUTPUT command to divert the printed output to microfiche.

(2) Graphical output may also be generated via the SCORS software system by appropriate substitution for the lines marked by * (see Reference 21). The above control statements assume plots will be generated on a Versatec hardcopy plotter; other options are available (see Reference 20).

FIGURE 12a

```
COYOTE, TXXXX, STSCZ.                                    NAME      BOX NO.
USER .....
SLTLIB.
UID.      NAME      BOX NO.
CUPDATE, P=COYOTEC, UN=DKGARTL, F.
CFT, I=COMPILE, L=0.
* RSCORLB.
* ASSIGN, DN=POPIN, A=FT10.
* LDR, LIB=RSCORS.
* POP, POPIN, POPOUT, HC1.
* XCOMO. POPOUT. HC1. CS=R7.
7-8-9
7-8-9
COYOTE DATA
6-7-8-9
```

NOTES:

(1) For large jobs producing large quantities of output, consideration should be given to using a XFICHE, $OUT command to divert the printed output to microfiche.

FIGURE 12b

```
COYOTE, TXXXX, ECYYY.                                   NAME     BOX NO.
USER .....
PFGET, FLNAME, COYOTE, AU=DKGARTL.
UPDATE, P=FLNAME, N, F.
FTN, I=COMPILE, L=0.
LABEL, TAPE13, W, L=$$.
LABEL, TAPE19, W, L=$$.
STAGE, TAPE13, VSN=AAAAAA, HY, POST.
STAGE, TAPE19, VSN=BBBBBB, HY, POST.
RFL, L=YYY.
LGO, PL=777777B.
7-8-9
7-8-9
COYOTE DATA
6-7-8-9
```

NOTES:

(1) The above listing uses (skeleton labeled) magnetic tapes to store the element data and solution fields. The permanent file system could also be used to save the data by replacing the STAGE and LABEL commands with the appropriate PFSAVE commands. Note that PFSAVE commands should follow the LGO card.

(2) The above listing is suitable for restarting a COYOTE run from tape. When restarting, the POST parameter on both STAGE commands should be changed to PRE. When restarting from permanent file storage, the PFSAVE commands should be replaced by an appropriate set of PFGET commands.

(3) The inclusion of plotting in the above run is accomplished through the addition of the appropriate commands from the control stream shown in Figure 12a (i.e., those with an *).

FIGURE 13a

```
COYOTE, TXXXX, STSCZ.                                    NAME      BOX NO.
USER .....
SLTLIB.
UID.      NAME      BOX NO.
CUPDATE, P=COYOTEC, UN=DKGARTL, F.
CFT, I=COMPILE, L=0.
LDR.
XTAPEOT, FT13, L=($$), VSN=AAAAAA.
XTAPEOT, FT19, L=($$), VSN=BBBBBB.
7-8-9
7-8-9
COYOTE DATA
6-7-8-9
```

NOTES:

(1) The above listing uses (skeleton labeled) magnetic tapes to store the element data and solution fields. The permanent file system could also be used to save the data by replacing the XTAPEOT commands with the appropriate XPFSAVE commands.

(2) The above listing is suitable for restarting a COYOTE run from tape. When restarting, the XTAPEOT commands are replaced by XTAPEIN commands; the XTAPEIN commands precede the LDR command. When restarting from permanent file storage, the XPFSAVE commands are replaced by XPFGET commands.

(3) The inclusion of plotting in the above run is accomplished through the addition of the appropriate commands from the control stream shown in Figure 12b (i.e., those with an *).

FIGURE 13b

```
COYOTE, TXXXX, ECYYY.                                    NAME      BOX NO.
USER .....
PFGET, FLNAME, COYOTE, AU=DKGARTL.
UPDATE, P=FLNAME, N, F.
FTN, I=COMPILE, L=0.
RFL, L=YYY.
LGO, PL=777777B.
7-8-9
* IDENT, USERSUB
* DELETE, USER.2, USER.38
FORTRAN CODED USER SUBROUTINES
7-8-9
COYOTE DATA
6-7-8-9
```

NOTES:

(1) The inclusion of plotting in the above run is accomplished through the addition of the appropriate commands from the control stream shown in Figure 12a (i.e., those with an *).

FIGURE 14a

```
COYOTE, TXXXX, STSCZ.                                    NAME      BOX NO.
USER .....
SLTLIB.
UID.       NAME       BOX NO.
CUPDATE, P=COYOTEC, UN=DKGARTL, F.
CFT, I=COMPILE, L=0.
LDR.
7-8-9
* IDENT, USERSUB
* DELETE, USER.2, USER.38
      FORTRAN CODED USER SUBROUTINES
7-8-9
      COYOTE DATA
6-7-8-9
```

NOTES:

(1) The inclusion of plotting in the above run is accomplished through the addition of the appropriate commands from the control stream shown in Figure 12b (i.e., those with an *).

FIGURE 14b

EXAMPLE PROBLEMS

A series of six problems have been included in this section to demonstrate the use of the previously described command cards and some of the capabilities of the COYOTE code. Though the examples illustrate many of the salient features of the code, all possible options and subtleties of the program could not be covered. However, a careful study of the input listings and the associated figures should provide the user with sufficient background to successfully employ COYOTE on other heat conduction problems.

Grid Generation

The first example consists of four problems which demonstrate several points about the relationship between node and element generation. The I shaped region defined in Figure 15 is to be gridded using a number of quadrilateral elements. The input listing shown in Figure 16 indicates four methods (hereafter denoted A, B, C, and D, respectively) by which this could be accomplished.

In the four cases illustrated, 147 nodal points were generated in two parts; the physical location of the nodes remained constant for the three cases. A plot of the nodes, as generated by the PLOT, POINTS command, is shown in Figure 17.

The input deck for Case A resulted in the element mesh shown in Figure 18a. Note that in generating the eight-node elements, only the first I, J for the element was specified with the program assuming default values for the remaining element I, J quantities. The mesh generated by input deck B, produced the grid shown in Figure 18b. In this case, all the corner I, J values were specified for elements in the lefthand side of the region with only the mid-side I, J's taking on default values. The difference between grids A and B illustrate that the user is free to choose any meaningful group of nodal points produced by the grid data in the construction of an element.

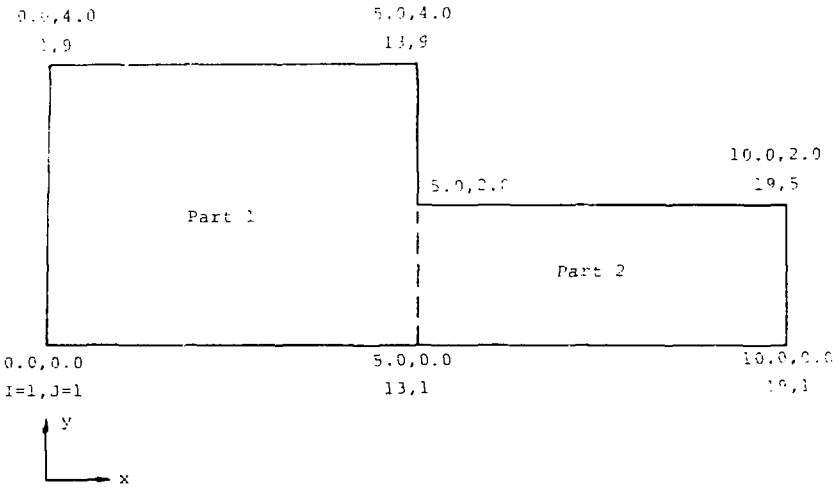


FIGURE 17. Example Problem 1 - L shaped region

The input deck for Case C illustrates the use of a discontinuous I, J numbering scheme between parts of the grid. Observe that in the second part of the grid generation (Figure 16c), the I numbering begins with 23 instead of 13 as in Cases A and B. The nodes that are common to the two parts (i.e., share the same physical location, denoted by X's in Figure 17) thus have two legitimate I, J names (e.g., (13,1) = (23,1)). In the construction of elements that use these doubly named nodes, care must be taken to consistently use only one of the names. This is shown in input deck C where the I, J names from grid part 1 have been used for the common nodes. The mesh generated by deck C is identical to that of deck A (Figure 18a).

The final case (D) demonstrates the appropriate formats for the element cards when using four-node elements. In the first part of the grid generation (Figure 16a), only the first I, J for the element was specified with the program assuming default values for the remaining corner I, J quantities; the I and J

```
LINE   DIRECT LIST OF INPUT DATA

   1 $EXAMPLE PROBLEM ONE--MESH GENERATION A
   2 SETUP,2,19,PRESCRIBED,PLOT                    $ SETUP COMMAND SEQUENCE
   3 MATERIAL,1,1.0,1.0,1.0                        $ MATERIAL DATA
   4 END
   5 1,1,13,9                                      $ GRID DATA-PART 1
   6 0.,5.,5.,0.
   7 0.,0.,4.,4.
   8 13,1,19,5                                     $ PART 2
   9 5.,10.,10.,5.
  10 0.,0.,2.,2.
  11 END
  12 ILOOP,6,2                                     $ ELEMENT DATA
  13 JLOOP,4,2
  14 QUAD8/4,1,1,1
  15 JEND
  16 IEND
  17 ILOOP,3,2
  18 JLOOP,2,2
  19 QUAD8/4,1,13,1
  20 JEND
  21 IEND
  22 END                                           $ END OF SETUP SEQUENCE
  23 PLOT,POINTS,0.,0.,10.,4.                      $ NODAL POINT PLOT
  24 PLOT,ELEMENTS,0.,0.,10.,4.                    $ ELEMENT PLOT
  25 END,NOPRINT
```

(a) Mesh A

```
  26 $EXAMPLE PROBLEM ONE--MESH GENERATION B
  27 SETUP,2,19,PRESCRIBED,PLOT                    $ SETUP COMMAND SEQUENCE
  28 MATERIAL,1,1.0,1.0,1.0                        $ MATERIAL DATA
  29 END
  30 1,1,13,9                                      $ GRID DATA-PART 1
  31 0.,5.,5.,0.
  32 0.,0.,4.,4.
  33 13,1,19,5                                     $ PART 2
  34 5.,10.,10.,5.
  35 0.,0.,2.,2.
  36 END
  37 ILOOP,3,4                                     $ ELEMENT DATA
  38 JLOOP,4,2
  39 QUAD8/4,1,1,1,5,1,5,3,1,3
  40 JEND
  41 IEND
  42 ILOOP,3,2
  43 JLOOP,2,2
  44 QUAD8/4,1,13,1
  45 JEND
  46 IEND
  47 END
  48 PLOT,POINTS,0.,0.,10.,4.                      $ NODAL POINT PLOT
  49 PLOT,ELEMENTS,0.,0.,10.,4.                    $ ELEMENT PLOT
  50 END,NOPRINT
```

(b) Mesh B

FIGURE 16. Input Listing--Example Problem 1

```
 51 $EXAMPLE PROBLEM ONE--MESH GENERATION C
 52 SETUP,2,29,PRESCRIBED,PLOT                    $ SETUP COMMAND SEQUENCE
 53 MATERIAL,1,1.0,1.0,1.0                        $ MATERIAL DATA
 54 END
 55 1,1,13,9                                      $ GRID DATA-PART 1
 56 0.,5.,5.,0.
 57 0.,0.,4.,4.
 58 23,1,29,5                                     $ PART 2
 59 5.,10.,10.,5.
 60 0.,0.,2.,2.
 61 END
 62 ILOOP,6,2                                     $ ELEMENT DATA
 63 JLOOP,4,2
 64 QUAD8/4,1,1,1
 65 JEND
 66 IEND
 67 QUAD8/4,1,13,1,25,1,25,3,13,3,24,1,25,2,24,3,13,2
 68 QUAD8/4,1,13,3,25,3,25,5,13,5,24,3,25,4,24,5,13,4
 69 ILOOP,2,2
 70 JLOOP,2,2
 71 QUAD8/4,1,25,1
 72 JEND
 73 IEND
 74 END                                           $ END OF SETUP SEQUENCE
 75 PLOT,POINTS,0.,0.,10.,4.                      $ NODAL POINT PLOT
 76 PLOT,ELEMENTS,0.,0.,10.,4.                    $ ELEMENT PLOT
 77 END,NOPRINT
```

(c) Mesh C

```
 78 $EXAMPLE PROBLEM ONE--MESH GENERATION D
 79 SETUP,2,29,PRESCRIBED,PLOT
 80 MATERIAL,1,1.0,1.0,1.0
 81 END
 82 1,1,13,9
 83 0.,5.,5.,0.
 84 0.,0.,4.,4.
 85 13,1,19,5
 86 5.,10.,10.,5.
 87 0.,0.,2.,2.
 88 END
 89 ILOOP,12,1
 90 JLOOP,3,1
 91 QUAD4/4,1,1,1
 92 JEND
 93 IEND
 94 ILOOP,3,2
 95 JLOOP,4,1
 96 QUAD4/4,1,13,1,15,1,15,2,13,2
 97 JEND
 98 IEND
 99 END
100 PLOT,POINTS,0.,0.,10.,4.                      $ NODAL POINT PLOT
101 PLOT,ELEMENTS,0.,0.,10.,4.                    $ ELEMENT PLOT
102 STOP,NOPRINT
```

(d) Mesh D

FIGURE 16. Input Listing--Example Problem 1

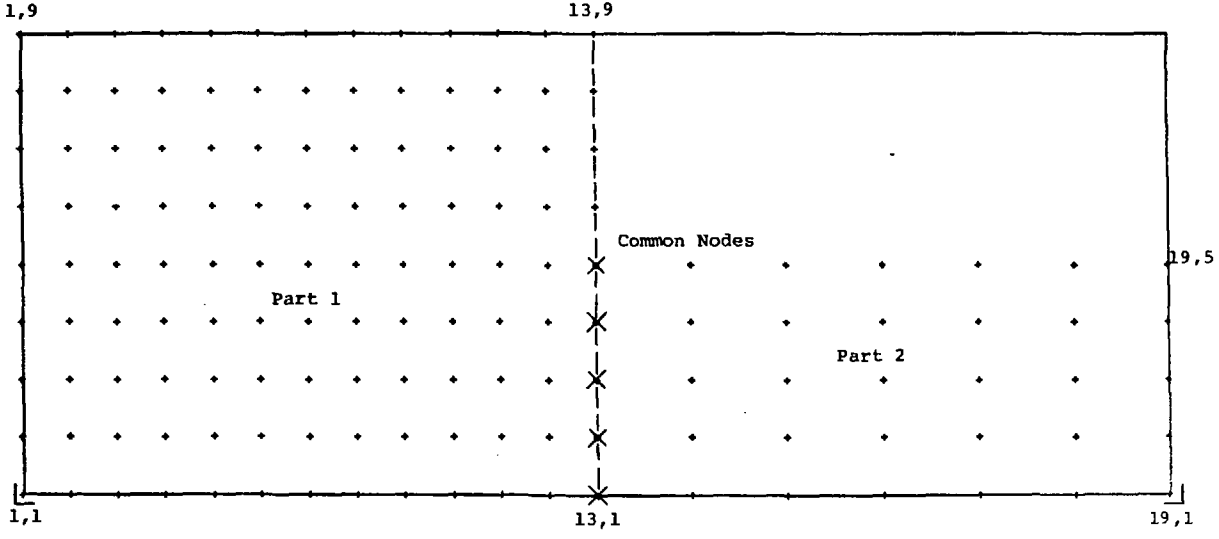


FIGURE 17. Points Plot--Example Problem 1

loop increments are set to unity. The second part of the mesh generation again shows how any suitable nodal point combination can be used to define an element. In this case, all four corner node I, J values are specified with the I and J loops being incremented by two and one, respectively. The mesh generated using deck D is shown in Figure 18c.

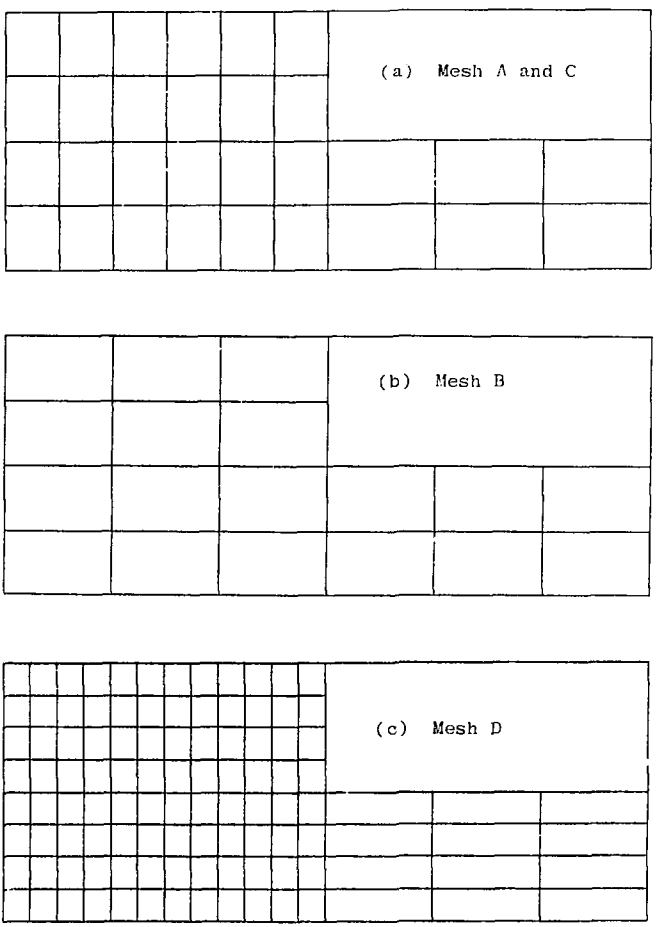


FIGURE 18. Element Mesh Plots--Example Problem 1

Heat Conduction in a Steel Bar

The second example consists of a steel bar (square cross section) with a circular hole that is subjected to prescribed temperature boundary conditions. The two-dimensional idealization of the problem is shown in Figure 19. Due to symmetry considerations, only one-eighth of the cross section needed to be considered in the finite element model. A COYOTE data deck used to solve the steady state version of this problem is shown in Figure 20a.

As indicated by the input listing, the grid points for this problem were generated in a single part; the curved part of the boundary was produced by use of a quadratic interpolation. The finite element mesh is shown in Figure 21. The input listing in Figure 20a also indicates that a steady state solution was saved by a RESTART command to allow additional processing at a later time. The control cards needed to complete the restart procedure are described in a previous section.

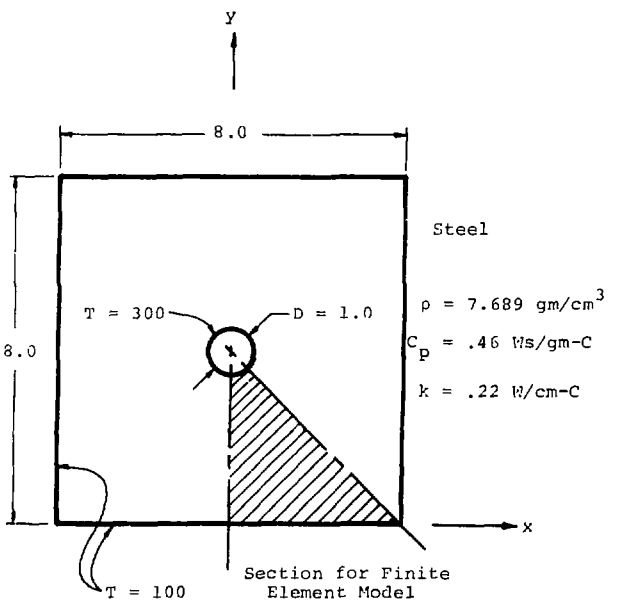


FIGURE 19. Example Problem 2--Heat Conduction in a Steel Bar

```
LINE    DIRECT LIST OF INPUT DATA

   1 $EXAMPLE PROBLEM TWO--HEAT CONDUCTION IN A STEEL BAR
   2 SETUP,2,25,PRESCRIBED                    $ SETUP COMMAND SEQUENCE
   3 STEEL,1,7.689,.46,.22                    $ MATERIAL DATA
   4 END
   5 1,1,25,21                                $ GRID DATA
   6 0.,.4,1,.353,0.,,,,.191
   7 0.,0.,3.645,3.5,,,3.538
   8 END
   9 ILOOP,12,2                               $ ELEMENT DATA
  10 JLOOP,10,2
  11 QUAD8/9,1,1,1
  12 JEND
  13 IEND
  14 ILOOP,12,2                               $BOUNDARY CONDITION DATA
  15 BC,TSIDE,1,1,1,100.
  16 BC,TSIDE,1,19,3,300.
  17 IEND
  18 JLOOP,10,2
  19 BC,QSIDE,1,1,4,0
  20 JEND
  21 END                                      $ END OF SETUP SEQUENCE
  22 FORMKF                                   $ FORM MATRIX EQUATIONS
  23 ZIPP,STEADY                              $ STEADY STATE SOLUTION
  24 END
  25 PLOT,ELEMENTS,0.,,0.,,4.,4.              $ ELEMENT PLOTS
  26 PLOT,ELEMENTS,0.,,0.,,4.,,,,,,,,,NUMBER
  27 PLOT,CONTOUR,0.,,0.,,4.,,4.              $ PLOT RESULTS
  28 ISOTHERMS,1,10
  29 END
  30 RESTART,SAVE                             $ SAVE SOLUTION
  31 STOP
```

(a) Steady State Solution

```
LINE    DIRECT LIST OF INPUT DATA

   1 $EXAMPLE PROBLEM TWO--HEAT CONDUCTION IN A STEEL BAR
   2 RESTART,RESET                            $ RECALL SOLUTION
   3 HEATFLUX,1,11,21,31,41,51                $ COMPUTE HEAT FLUXES
   4 HEATFLUX,1,61,71,81,91,101,111
   5 HEATFLUX,1,10,20,30,40,50,60,70,80,90,100,110,120
   6 STOP,NOPRINT
```

(b) Restart

FIGURE 20. Input Listing--Example Problem 2

```
LINE   DIRECT LIST OF INPUT DATA

   1 $EXAMPLE PROBLEM TWO--HEAT CONDUCTION IN A STEEL BAR
   2 SETUP,2,25,PRESCRIBED                    $ SETUP COMMAND SEQUENCE
   3 STEEL,1,7.689,.46,.22                    $ MATERIAL DATA
   4 END
   5 1,1,25,21                                $ GRID DATA
   6 0.,4.0,.353,0.,,,.191
   7 0.,0.,3.646,3.5,,,3.538
   8 END
   9 ILOOP,12,2                               $ ELEMENT DATA
  10 JLOOP,10,2
  11 QUAD8/8,1,1,1
  12 JEND
  13 IEND
  14 ILOOP,12,2                               $BOUNDARY CONDITION DATA
  15 BC,TSIDE,1,1,1,300.
  16 BC,TSIDE,1,19,3,300.
  17 IEND
  18 JLOOP,10,2
  19 BC,QSIDE,1,1,4,0.
  20 JEND
  21 END                                      $ END OF SETUP SEQUENCE
  22 FORMKF                                   $ FORM MATRIX EQUATIONS
  23 ZIPP,TRANS1,,2,0.,4.,,4,10,TAPE          $ TRANSIENT SOLUTION
  24 ZIPP,TRANS1,,2,4.0,10.,.5,12
  25 ZIPP,TRANS1,,2,10.,15.,1.0,5
  26 END
  27 PLOT,CONTOUR,0.,3.,4.,4.                 $ PLOT RESULTS
  28 PLOTLOOP,23,2
  29 ISOTHERMS,1,9,120.,140.,160.,180.,200.,220.,240.,260.,280.
  30 PLOTEND
  31 END
  32 PLOT,HISTORY                             $ HISTORY PLOTS
  33 TLOCATION,4,1,30,2,4,4,4,6,4,8,4
  34 TLOCATION,4,1,30,112,3,114,3,116,3,118,3
  35 END
  36 STOP,NOPRINT
```

(c) Transient Solution

FIGURE 20. Input Listing--Example Problem 2

The input listing in Figure 20b indicates the use of the restart capability. In this case, heat flux calculations were performed for a number of elements in the previously computed solution.

A second problem was analyzed to demonstrate the setting of initial conditions from an external tape file and a transient solution procedure. In Figure 20c, an input listing is provided for a transient analysis of the steel bar geometry. For this case, the outside temperature of the bar was raised from $100^{\circ}C$ to $300^{\circ}C$ to allow the bar to reach a uniform temperature state. The ZIPP command card indicates that the initial temperature field was to be read from a file called TAPE19. Through the job control cards, the steady state solution obtained from the listing in Figure 20a was identified as TAPE19. The steady state solution thus became the initial condition for a related transient analysis. Figure 22 shows several of the contour plots of the isotherms obtained during this analysis.

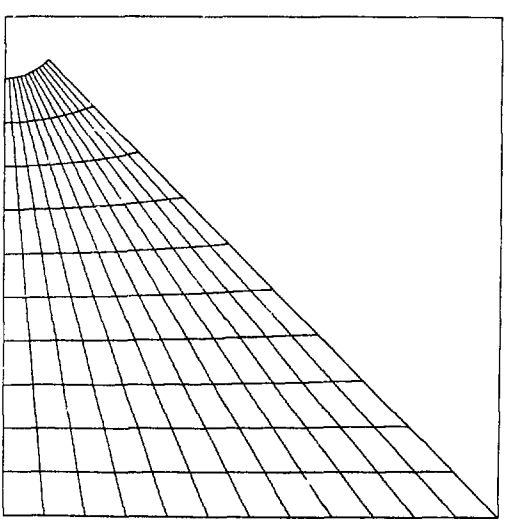


FIGURE 21. Element Mesh Plot--Example Problem 2

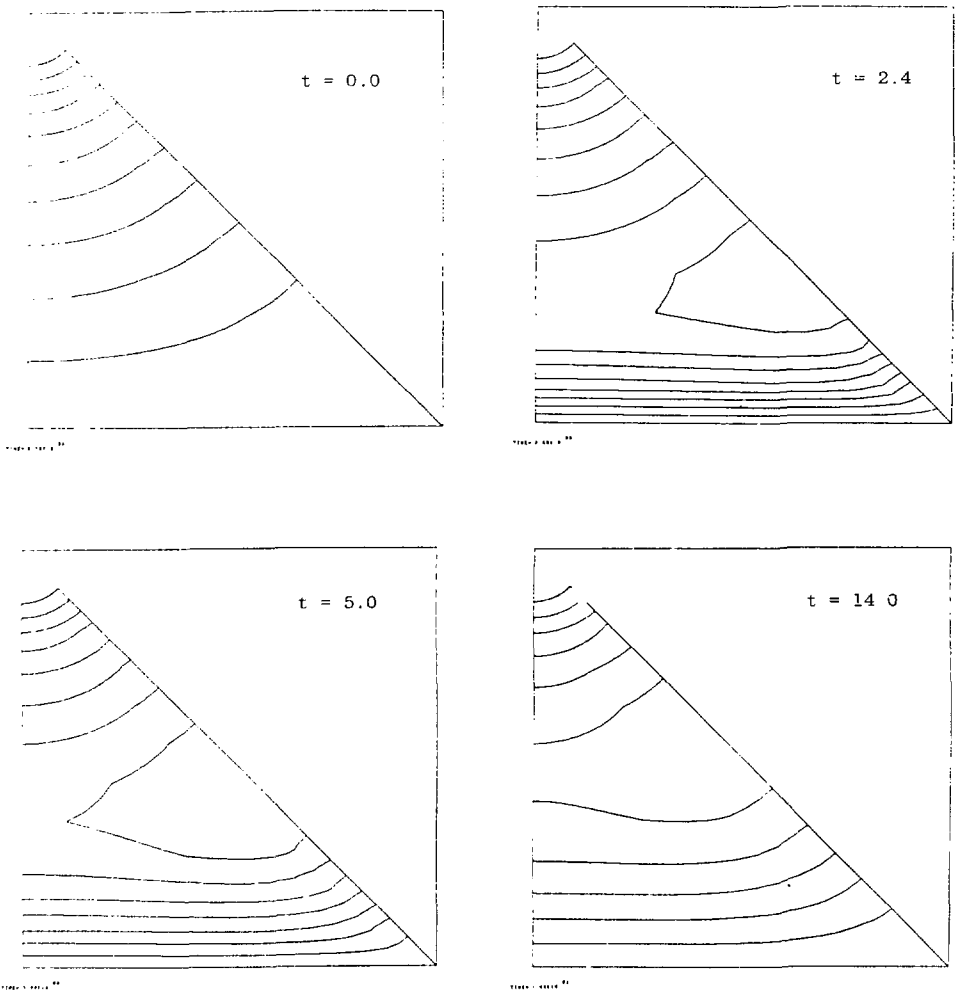


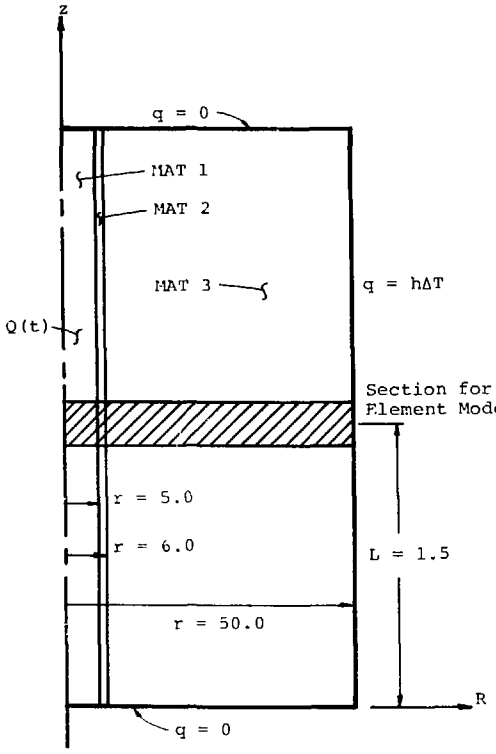
FIGURE 22. Contour Plots--Example Problem 2

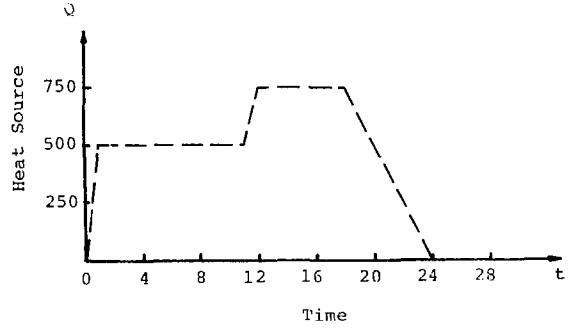
One-Dimensional Heat Conduction in a Cylinder

To demonstrate the use of user subroutines for volumetric heating and generalized convection/radiation boundary conditions, a one-dimensional problem was considered. A cylindrical region of heat generating material is encased by a thin layer of low conductivity material and a thicker layer of material having a relatively high thermal conductivity. The outer surface of the cylinder loses heat to the surrounding environment by natural convection. The geometry, boundary conditions, and material properties for this problem are shown in Figure 23.

The volumetric heating history and the heat transfer correlations for laminar and turbulent natural convection shown in Figure 23 were provided to the COYOTL program through subroutines SOURCE and HTCOEF. A listing of these subroutines and the input data deck for this problem are given in Figure 24. Note that the material data card for MAT 1 indicates a variable heat source; the boundary condition SET card indicates a variable heat transfer coefficient.

A transient analysis of this problem was carried out using four integration intervals as shown by the TIDP command cards. The temperature histories at several radial positions were generated using the HISTORY plot command. This plot is shown in Figure 25 where the element mesh is also shown for reference.





	ρ	C_p	k
MAT 1	2.274×10^{-3}	.244	.01
MAT 2	1.081×10^{-3}	.122	.001
MAT 3	2.162×10^{-3}	.244	.04

$h = .29(\frac{\Delta T}{L})^{.25}$ for $10^4 \leq Gr \leq 10^9$

$h = .19(\Delta T)^{.33}$ for $10^9 > Gr$

$Gr = \rho^2 g \beta L^3 \Delta T / \mu^2$

FIGURE 23. Example Problem 3--One-Dimensional Heat Conduction in a Cylinder

```
LINE   DIRECT LIST OF INPUT DATA

   1 $EXAMPLE PROBLEM THREE--ONE-DIMENSIONAL CONDUCTION IN A CYLINDER
   2 SETUP,2,61                                  $ SETUP COMMAND SEQUENCE
   3 MAT1,1,2.274E-3,.244,.01,,,,VARIABLE,20.    $MATERIAL DATA
   4 MAT2,2,1.031E-3,.122,.004,,,,,20.
   5 MAT3,3,2.162E-3,.244,.04,,,,,20.
   6 END
   7 1,1,11,3                                    $ GRID DATA-PART 1
   8 0.,5.,5.,9.
   9 0.,0.,1.,1.
  10 11,1,13,3                                   $ PART 2
  11 5.,6.,6.,5.
  12 0.,0.,1.,1.
  13 13,1,61,3                                   $ PART 3
  14 6.,50.,50.,6.
  15 0.,0.,1.,1.
  16 END
  17 ILOOP,5,2                                   $ ELEMENT DATA
  18 QUAD8/4,1,1
  19 IEND
  20 QUAD8/4,2,11,1
  21 ILOOP,24,2
  22 QUAD8/4,3,13,1
  23 IEND
  24 SET,QRAD,1,VARIABLE,20.                     $ BOUNDARY CONDITION DATA
  25 BC,QRAD,59,1,2,1
  26 END                                         $ END SETUP SEQUENCE
  27 FORMKE,AXISYM                               $ FORM MATRIX EQUATIONS
  28 ZIPP,TRANS1,,2,0  ,2.0,,10,20               $ TRANSIENT SOLUTION
  29 ZIPP,TRANS1,,5,2.0,11.0,,25,36
  30 ZIPP,TRANS,,5,11.0,13.0,,1,20
  31 ZIPP,TRANS1,,5,13.,30.,,25,68
  32 END
  33 PLOT,ELEMENTS,0.,,0.,,50.,,1.,,,,,,1  0,4.0 $ ELEMENT PLOT
  34 HEATFLUX,ALLTIMES                           $ COMPUTE HEAT FLUXES
  35 PLOT,HISTORY                                $ HISTORY PLOTS
  36 TLOCATION,4,1,150,1,8,5,6,6,6,30,6
  37 QLOCATION,2,1,150,5,3,6,3
  38 END
  39 STOP
```

(a) Input Listing

FIGURE 24. Input Listing--Example Problem 3

```
      SUBROUTINE SOURCE (QVALUE,T,X,Y,NNODES,MAT,NELEM,TIME)
      DIMENSION QVALUE(1),T(1),X(1),Y(1)
C
C     SUBROUTINE TO EVALUATE THE VOLUME HEATING
C
      IF(TIME.GT.24.) GO TO 50
      IF(TIME.GT.18.) GO TO 40
      IF(TIME.GT.12.) GO TO 30
      IF(TIME.GT.11.) GO TO 20
      IF(TIME.GT.1.) GO TO 10
C
      DO 5 I=1,NNODES
      QVALUE(I)=.120*TIME
    5 CONTINUE
      RETURN
   10 CONTINUE
      DO 15 I=1,NNODES
      QVALUE(I)=.120
   15 CONTINUE
      RETURN
   20 CONTINUE
      DO 25 I=1,NNODES
      QVALUE(I)=.120+.060*(TIME-11.)
   25 CONTINUE
      RETURN
   30 CONTINUE
      DO 35 I=1,NNODES
      QVALUE(I)=.180
   35 CONTINUE
      RETURN
   40 CONTINUE
      DO 45 I=1,NNODES
      QVALUE(I)=.180-.130*(TIME-18.)/6.
   45 CONTINUE
      RETURN
   50 CONTINUE
      DO 55 I=1,NNODES
      QVALUE(I)=0.
   55 CONTINUE
      RETURN
      END
```

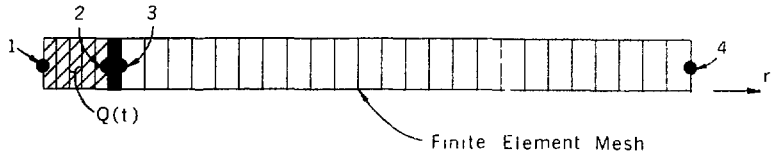
(b) Source Subroutine

FIGURE 24. Input Listing--Example Problem 3

```
      SUBROUTINE HTCOEF (HT,TSURF,TREF,XSURF,YSURF,TIME,ISET,NELEM)
C
C     SUBROUTINE TO EVALUATE THE HEAT TRANSFER COEFFICIENT FOR
C     FREE CONVECTION IN AIR
C
      AL=1.5
      G=32.2
      CONVERT=5.678E-4
C
C     CONVERT TEMPERATURE TO RANKINE
C
      DELT=(TSURF-TREF)*9./5.
      TFILM=((TSURF+TREF)*.5+273.)*9./5.
C
C     EVALUATE AIR PROPERTIES
C
      BETA=1./TFILM
      AMU=(7.3094E-7)*((TFILM**1.5)/(TFILM+198.))
      RHO=39.68/TFILM
C
C     EVALUATE GRASHOF NUMBER
C
      GR=(RHO*RHO*G*BETA*AL*AL*AL*DELT)/(AMU*AMU)
C
C     EVALUATE H
C
      IF(GR.LT.1.0E9) 10,20
   10 CONTINUE
C     LAMINAR FLOW
      HT=.29*(DELT/AL)**.25
      IF(GR.LT.1.0E4) HT=0.
      HT=HT*CONVERT
      RETURN
   20 CONTINUE
C     TURBULENT FLOW
      HT=.19*(DELT**.333333)
      HT=HT*CONVERT
      RETURN
      END
```

(c) Heat Transfer Coefficient Subroutine

FIGURE 24. Input Listing--Example Problem 3



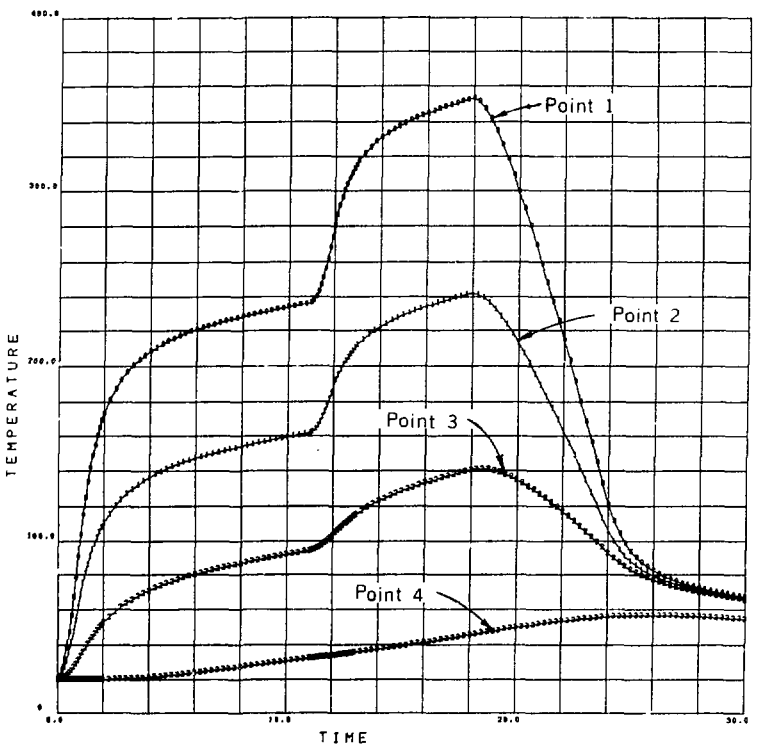


FIGURE 25. Time History Plot--Example Problem 3

Heat Conduction in a Finned Radiator

The finned tube radiator section shown in Figure 26a was chosen to illustrate the use of time dependent boundary conditions. Symmetry considerations allowed the finite element model in Figure 26b to be constructed for this problem. The inside temperature of the aluminum tube was maintained at $100^{\circ}F$; the outside surfaces of the radiator were subjected to the heat flux history shown in Figure 26b.

The input data deck for the analysis of this problem is shown in Figure 27a. The use of the QVARY boundary condition requires that a time history subroutine (corresponding to the time history number) be appended to the COYOTE program. For the present case, the flux history shown in Figure 26b was supplied through SUBROUTINE CURVE1, which is listed in Figure 27b. As a result of the transient analysis of this problem, the contour plots in Figure 28 were produced, indicating the thermal response of the radiator.

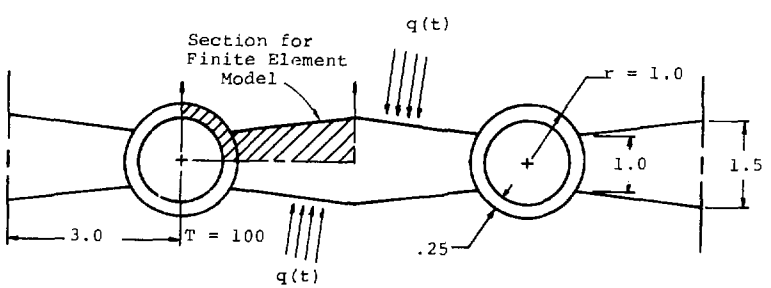
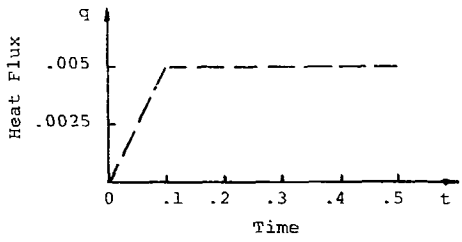


FIGURE 26a. Example Problem 4--Heat Conduction in a Finned Radiator



Aluminum

ρ = .0978 lbm/in^3

C_p = .208 Btu/lbm-F

k = 2.778 x 10^{-3} Btu/s-in-F

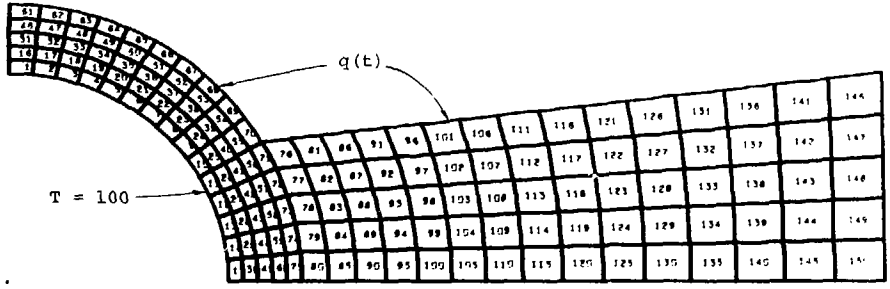


FIGURE 26b. Example Problem 4--Heat Conduction in a Finned Radiator

```
LINE   DIRECT LIST OF INPUT DATA

   1 $EXAMPLE PROBLEM FOUR--FINNED RADIATOR
   2 SETUP,2,31,PRESCRIBED                    $ SETUP COMMAND SEQUENCE
   3 ALUMINUM,1.,.0978,.208,2.778E-3,,,,,100.     $ MATERIAL DATA
   4 END
   5 1,1,11,11,,,,,POLAR                      $ GRID DATA-PART 1
   6 .75,.75,1.0,1.0,.75,,1.0
   7 90.,60.,60.,90.,75.,,75.
   8 11,1,21,11,,,,,POLAR                     $ PART 2
   9 .75,.75,1.0,1.0,.75,,1.0
  10 60.,30.,30.,60.,45.,,45.
  11 21,1,31,11,,,,,POLAR                     $ PART 3
  12 .75,.75,1.0,1.0,.75,,1.0
  13 30.,0.,0.,30.,15.,,15.
  14 21,11,31,41,,,5,,,5                      $ PART 4
  15 .966,1.0,3.0,3.0,.9659
  16 .5,0.,0.,.75,.2588
  17 END
  18 JLOOP,5,2                                $ ELEMENT DATA
  19 ILOOP,15,2
  20 QUAD8/9,1,1,1
  21 IEND
  22 JEND
  23 JLOOP,15,2
  24 ILOOP,5,2
  25 QUAD8/8,1,21,11
  26 IEND
  27 JEND
  28 ILOOP,15,2                               $ BOUNDARY CONDITION DATA
  29 BC,TSIDE,1,1,1,100.
  30 IEND
  31 ILOOP,10,2
  32 BC,QVARY,1,9,3,1
  33 IEND
  34 JLOOP,15,2
  35 BC,QVARY,21,11,4,1
  36 JEND
  37 END                                      $ END OF SETUP SEQUENCE
  38 PLOT,ELEMENTS,-.01,0.,3.3,1.01           $ PLOT ELEMENTS
  39 PLOT,ELEMENTS,-.01,0.,3.3,1.01,,,,,,,NUMBER
  40 PLOT,ELEMENTS,-.01,0.,1.0,1.01
  41 FORMKF                                   $ FORM MATRIX EQUATIONS
  42 ZIPP,TRANS1,,5,0.,,.5,,.01,50            $ TRANSIENT SOLUTION
  43 END
  44 PLOT,CONTOUR,-.01,0.,3.3,1.01            $ PLOT RESULTS
  45 PLOTLOOP,10,5
  46 ISOTHERMS,5,12
  47 PLOTEND
  48 END
  49 PLOT,HISTORY
  50 TLOCATION,4,1,51,61,4,66,4,71,4,73,7
  51 TLOCATION,4,1,51,96,4,98,7,146,4,148,7
  52 END
  53 STOP
```

(a) Input Listing

FIGURE 27. Input Listing--Example Problem 4

```
      SUBROUTINE CURVE1 (NELEM,TSURF,TIME,QVALUE)
C
C     SUBROUTINE TO EVALUATE A TIME DEPENDENT BOUNDARY FLUX
C
      QVALUE=5.0E-3
      IF(TIME.LE..1) QVALUE=QVALUE*TIME/.1
      RETURN
      END
```

(b) Heat Flux Time History Subroutine

FIGURE 27. Input Listing--Example Problem 4

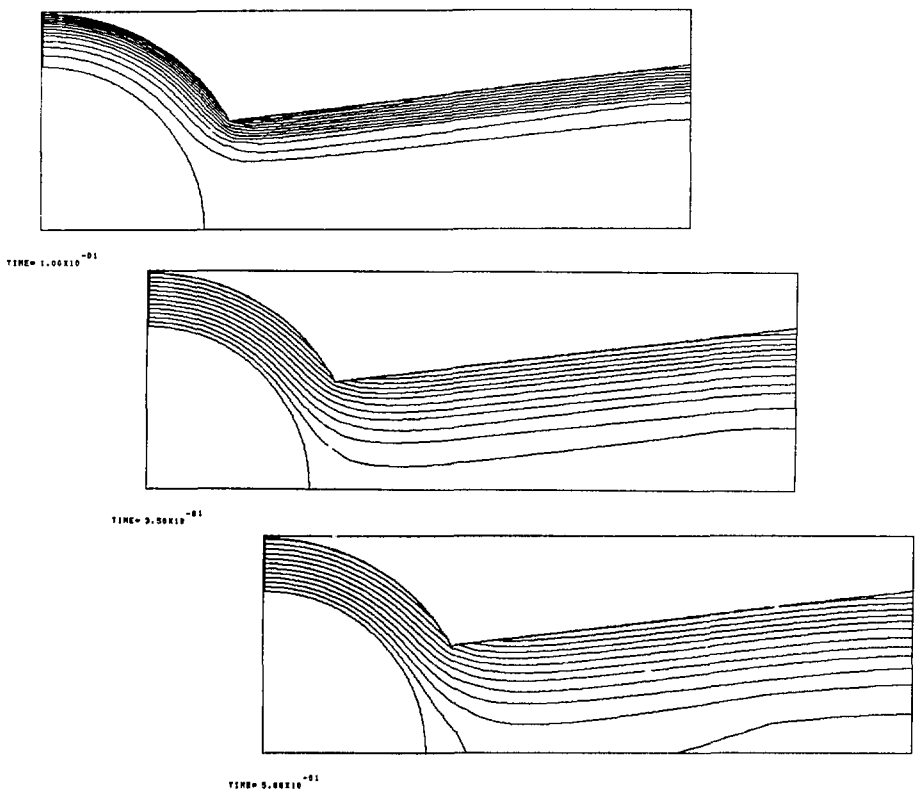
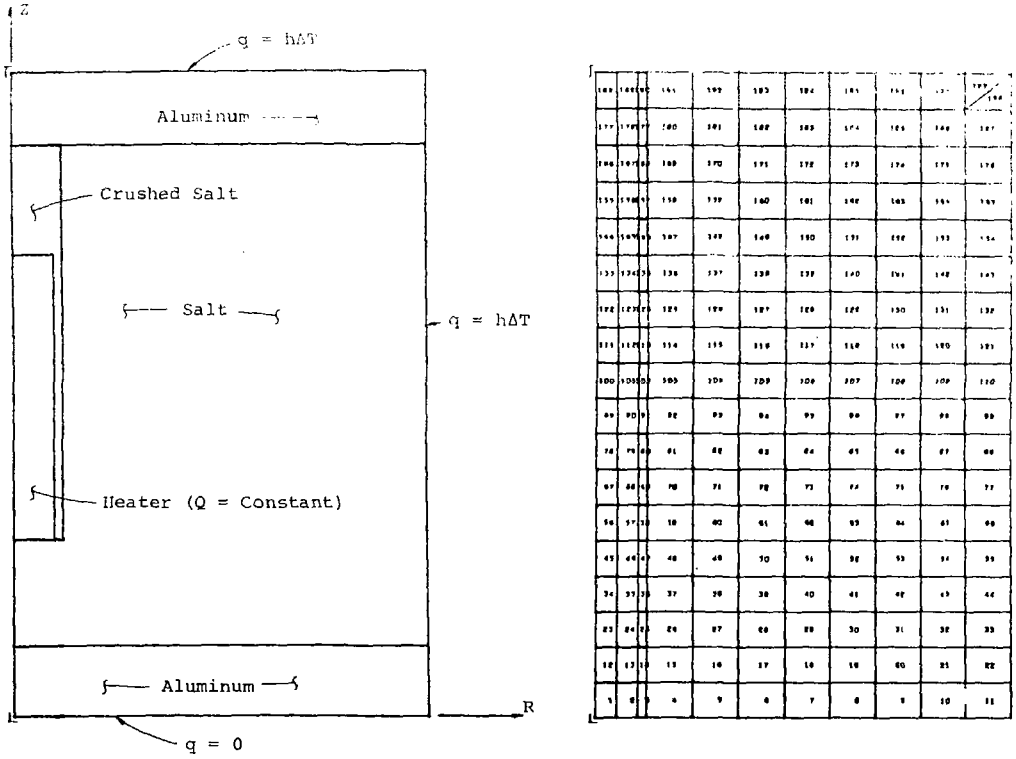


FIGURE 28. Contour Plots--Example Problem 4

Heated Salt Block

The penultimate problem has been selected to demonstrate the use of subroutines for temperature dependent properties and temperature dependent heat transfer coefficients. The physical problem consists of a right circular cylinder of bedded salt that is heated internally by a cylindrical heater. The annular space between the heater and salt block and the cylindrical space above the heater is filled with crushed salt. The salt block is confined on the upper and lower surfaces by circular aluminum plates. The problem geometry and finite element mesh are shown in Figures 29a and 29b. The heater is assumed to be producing heat at a constant volumetric rate; the cylinder is assumed to lose energy to the environment by natural convection through the top and vertical surfaces. The bottom surface is assumed insulated.

In the analysis of this problem, it was assumed that the thermal conductivity of the bedded salt and crushed salt varied with temperature. The aluminum and heater properties were assumed constant. Correlations for the heat transfer coefficient as a function of temperature were also to be included in the analysis. Figure 30a shows the input listing for the steady state solution to this nonlinear problem. The material property subroutine and heat transfer coefficient subroutine that were appended to the COYOTE program for this problem are listed in Figure 30b. Note that SUBROUTINE CONDUCT is used to evaluate the thermal conductivity for both bedded salt and crushed salt; SUBROUTINE HTCOEF is used to evaluate the film coefficient for both top and vertical boundary conditions (based on ISET parameter). The results of this analysis are shown in Figure 31 in the form of an isotherm plot.



a) Schematic

b) Finite Element Model

FIGURE 29. Example Problem 5--Heated Salt Block

```
 1 $EXAMPLE PROBLEM FIVE--HEATED SALT BLOCK
 2 SETUP,2,23,PRESCRIBED                          $ SETUP COMMAND SEQUENCE
 3 ALUMINUM,1,2.71,2.35E-4,2.042,,,CONSTANT,,491.    $ MATERIAL DATA
 4 SALT,2,2.15,2.56E-4,,.0351,,,VARIABLE,,540.
 5 HEATER,3,3.19,1.28E-4,,.450,,,CONSTANT,,1193,540.
 6 CRUSH-SALT,4,1.57,2.56E-4,,.0035,,,VARIABLE,,540.
 7 END
 8 1,1,5,37                                       $ GRID DATA-PART 1
 9 0.,5.,5.,0.
10 0.,0.,30.,30.
11 5,1,7,37                                       $ PART 2
12 5.,6.,6.,5.
13 0.,0.,30.,30.
14 7,1,23,37                                      $ PART 3
15 6.,50.,50.,6.
16 0.,0.,90.,90.
17 END
18 JLOOP,2,2                                      $ ELEMENT DATA
19 ILOOP,11,2
20 QUAD8/4,1,1,1
21 IEND
22 JEND
23 JLOOP,3,2
24 ILOOP,11,2
25 QUAD8/4,2,1,5
26 IEND
27 JEND
28 JLOOP,3,2
29 ILOOP,2,2
30 QUAD8/4,3,1,11
31 IEND
32 QUAD9/4,4,5,11
33 ILOOP,3,2
34 QUAD8/4,2,7,11
35 IEND
36 JEND
37 JLOOP,3,2
38 ILOOP,3,2
39 QUAD8/4,4,1,27
40 IEND
41 ILOOP,3,2
42 QUAD8/4,2,7,27
43 IEND
44 JEND
45 ILOOP,11,2
46 QUAD8/4,1,1,33
47 IEND
48 ILOOP,10,2
49 QUAD8/4,1,1,35
50 IEND
51 TRI6/3,1,21,35,23,35,23,37
52 TRI6/3,1,23,37,21,37,21,35
53 SET,QUAD,1,VARIABLE,20.                        $ BOUNDARY CONDITION DATA
54 SET,QUAD,2,VARIABLE,20.
```

(a) Input Listing

FIGURE 30. Input Listing--Example Problem 5

```
55 JLOOP,13,2
56 BC,QRAD,21,1,2,1
57 JEND
58 ILOOP,1),2
59 BC,QRAD,1,35,3,2
60 IEND
61 BC,QRAD,23,37,1,2
62 END                                          $ END OF SETUP SEQUENCE
63 PLOT,OUTLINE,0.,0.,50.,80.                  $ PLOT MODEL
64 PLOT,ELEMENTS,0.,0.,50.,80.,,,,,,,,NUMBER
65 FORMKE,AXISYM                               $ FORM MATRIX EQUATIONS
66 ZIPP,STEADY,3,4                             $ STEADY STATE SOLUTION
67 END
68 PLOT,CONTOUR,0.,0.,50.,80.                  $ PLOT RESULTS
69 ISOTHERMS,,15
70 END
71 STOP
```

(a) Input Listing (cont)

```
      SUBROUTINE CONDUCT (COND1,COND2,T,X,Y,NNODES,MAT,NELEM,TIME)
      DIMENSION COND1(1),COND2(1),T(1),X(1),Y(1)
C
C     SUBROUTINE TO EVALUATE THERMAL CONDUCTIVITY
C
      IF(MAT.EQ.2) GO TO 10
      IF(MAT.EQ.4) GO TO 50
   10 CONTINUE
C
C     SALT CONDUCTIVITY
C
      DO 20 I=1,NNODES
      TT=T(I)
      AK=3.702+(-.00387)*TT+(.252E-4)*TT*TT+(-.40E-7)*TT*TT*TT
      COND1(I)=AK/100.
      COND2(I)=COND1(I)
   20 CONTINUE
      RETURN
   50 CONTINUE
C
C     CRUSHED SALT CONDUCTIVITY
C
      DO 60 I=1,NNODES
      TT=T(I)
      AK=3.702+(-.00387)*TT+(.252E-4)*TT*TT+(-.40E-7)*TT*TT*TT
      AK=AK/.10
      COND1(I)=AK/100.
      COND2(I)=COND1(I)
   60 CONTINUE
      RETURN
      END
```

(b) Material Property Subroutine

FIGURE 30. Input Listing--Example Problem 5

```
      SUBROUTINE HICOEF (HT,TSURF,TREF,XSURF,YSURF,TIME,ISET,NELEM)
C
C     SUBROUTINE TO EVALUATE THE HEAT TRANSFER COEFFICIENT FOR
C     FREE CONVECTION IN AIR
C
      AL=1.5
      G=32.2
      CONVERT=5.678E-4
C
C     CONVERT TEMPERATURE TO RANKINE
C
      DELT=(TSURF-TREF)*9./5.
      TFILM=((TSURF+TREF)*.5+273.)*9./5.
C
C     EVALUATE AIR PROPERTIES
C
      BETA=1./TFILM
      AMU=(7.3094E-7)*((TFILM**1.5)/(TFILM+198.))
      RHO=39.68/TFILM
C
C     EVALUATE GRASHOF NUMBER
C
      GR=(RHO*RHO*G*BETA*AL*AL*AL*DELT)/(AMU*AMU)
C
C     CHECK BOUNDARY CONDITION SET NUMBER
C
      IF(ISET.EQ.2) GO TO 30
C
C     EVALUATE H--VERTICAL CYLINDER
C
      IF(GR.LT.1.0E9) 10,20
   10 CONTINUE
C     LAMINAR FLOW
      HT=.29*(DELT/AL)**.25
      IF(GR.LT.1.0E4) HT=0.
      HT=HT*CONVERT
      RETURN
   20 CONTINUE
C     TURBULENT FLOW
      HT=.19*(DELT**.333333)
      HT=HT*CONVERT
      RETURN
   30 CONTINUE
C
C     EVALUATE H--HORIZONTAL PLATE
C
      IF(GR.LT.1.0E9) 40,50
   40 CONTINUE
C     LAMINAR FLOW
      HT=.27*(DELT/AL)**.25
      IF(GR.LT.1.0E4) HT=0.
      HT=HT*CONVERT
      RETURN
   50 CONTINUE
C     TURBULENT FLOW
      HT=.22*(DELT**.333333)
      HT=HT*CONVERT
      RETURN
      END
```

(b) Heat Transfer Coefficient Subroutine

FIGURE 30. Input Listing--Example Problem 5

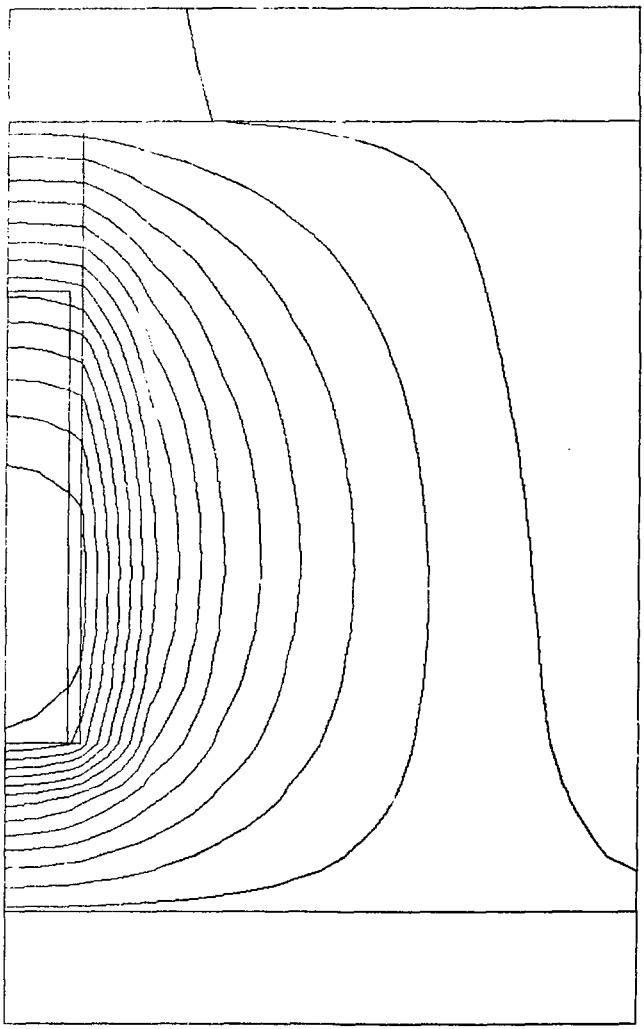


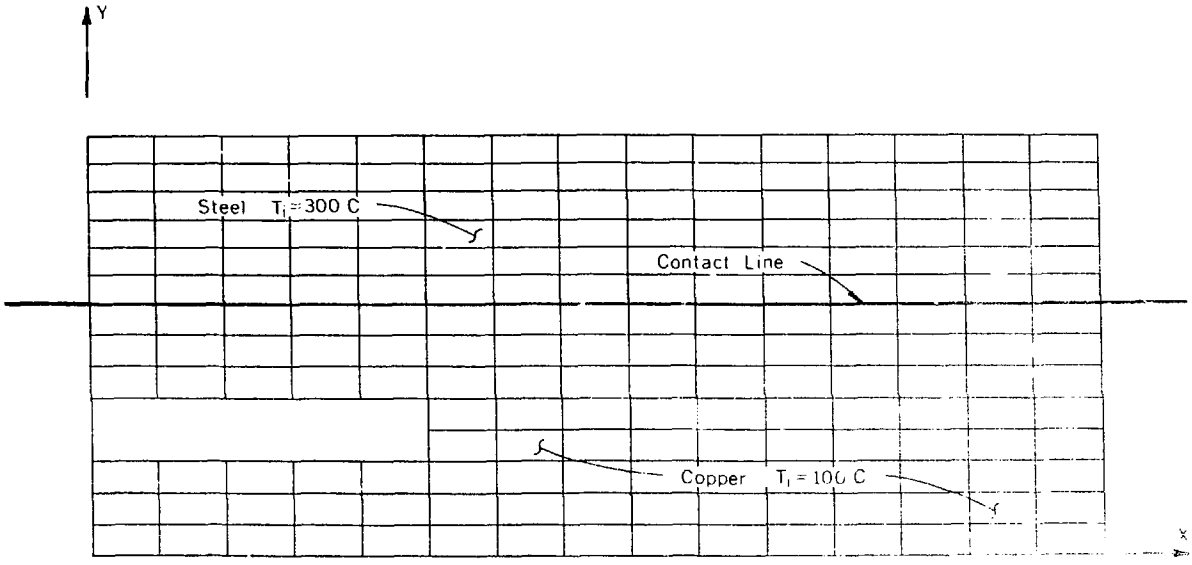
FIGURE 31. Contour Plots--Example Problem 5

Heat Conduction in a Slotted Bar

The final example problem was chosen to illustrate the use of four node elements and the selective output options in COYOTE. The problem of interest consists of a copper bar of rectangular cross-section that contains an internal rectangular slot. The bar is assumed insulated on three sides; energy is transmitted to the bar through the fourth side by placing it in contact with a steel bar of higher temperature. Both bars are initially at uniform temperatures, the steel is at 300°C and the copper at 100°C.

The finite element model used to solve this problem is shown in Figure 32 along with the material properties and boundary conditions. The input listing for the problem is given in Figure 33. Note the use of QUAD4/4 elements. Also, several points have been identified within the mesh as special output points. A transient solution to the problem is indicated with the generalized Crank-Nicolson time integration scheme ($\alpha = 0.667$).

In Figures 34-35 are shown some of the graphical results generated by this analysis. In particular, a series of temperature profiles taken just below the contact line is shown in Figure 34; a typical contour is shown in Figure 35. Finally, Figure 36 shows time histories of some of the special output points noted in Figure 32.



Steel Properties

ρ = 7.8 g/cm^3

C = 0.45 Ws/gK

k = 0.54 W/cmK

Copper Properties

ρ = 8.9 g/cm^3

C = 0.383 Ws/gK

k = 3.86 W/cmK

FIGURE 32. Example Problem 6--Heat Conduction in a Notched Bar

```
  1 $EXAMPLE PROBLEM SIX--HEAT CONDUCTION IN A SLOTTED BAR
  2 SETUP,2,16                                  $ SETUP COMMAND SEQUENCE
  3 STEEL,1,7.8,.465,.59,,,,,300.               $ MATERIAL DATA
  4 COPPER,2,8.9,.383,3.86,,,,,100.
  5 END
  6 1,1,16,9                                    $ GRID DATA-PART1
  7 0.,3.,3.,0.
  8 0.,0.,.75,.75
  9 1,9,16,15                                   $ PART 2
 10 0.,3.,3.,0.
 11 .75,.75,1.25,1.25
 12 END
 13 JLOOP,3,1                                   $ ELEMENT DATA
 14 ILOOP,15,1
 15 QUAD4/4,2,1,1
 16 IEND
 17 JEND
 18 JLOOP,2,1
 19 ILOOP,10,1
 20 QUAD4/4,2,5,4
 21 IEND
 22 JEND
 23 JLOOP,3,1
 24 ILOOP,15,1
 25 QUAD4/4,2,1,6
 26 IEND
 27 JEND
 28 JLOOP,6,1
 29 ILOOP,15,1
 30 QUAD4/4,1,1,9
 31 IEND
 32 JEND
 33 END                                         $ END SETUP SEQUENCE
 34 FORMKE                                      $ FORM MATRIX EQUATIONS
 35 OUTPUT,POINTS,1.5,.375,2.5,.375,.5,.8,1.5,.9,2.5,.8
 36 OUTPUT,POINTS,.5,.2,1.5,.2,2.5,.2
 37 PLOT,OUTLINE,0.,0.,3.,1.25                  $ PLOT MODEL
 38 PLOT,ELEMENTS,0.,0.,3.,1.25,,,,,,,NUMBER
 39 ZIPP,TRANS2,.6667,5,0.,15.,.5,30            $ TRANSIENT SOLUTION
 40 END
 41 PLOT,PROFILE                                $ PLOT RESULTS
 42 TIMEPLANE,4,6,11,16,21
 43 TLOCATION,SIDES,15,96,1,97,1,98,1,99,1,100,1,101,1,102,1,103,1,104,1*
 44  105,1,106,1,107,1,108,1,109,1,110,1
 45 TIMEPLANE,4,6,11,16,21
 46 TLOCATION,SIDES,10,56,1,57,1,58,1,59,1,60,1,61,1,62,1,63,1,64,1,65,1
 47 TIMEPLANE,4,6,11,16,21
 48 TLOCATION,NODES,13,31,4,32,4,33,4,34,4,35,4,36,4,46,4,56,4,70,1,69,1*
 49  68,1,67,1,66,1
 50 END
 51 PLOT,HISTORY
 52 TLOCATION,3,1,31,6,1,56,1,101,4
 53 END
 54 PLOT,HISTORY
 55 TSPECIAL,2,1,31,3,6
 56 END
 57 PLOT,CONTOUR,0.,0.,3.,1.25
 58 PLOTLOOP,5,1
 59 ISOTHERMS,2,9,120.,140.,160.,180.,200.,220.,240.,260.,280.
 60 PLOTEND
 61 END
 62 STOP,NOPRINT
```

FIGURE 33. Input Listing--Example Problem 6

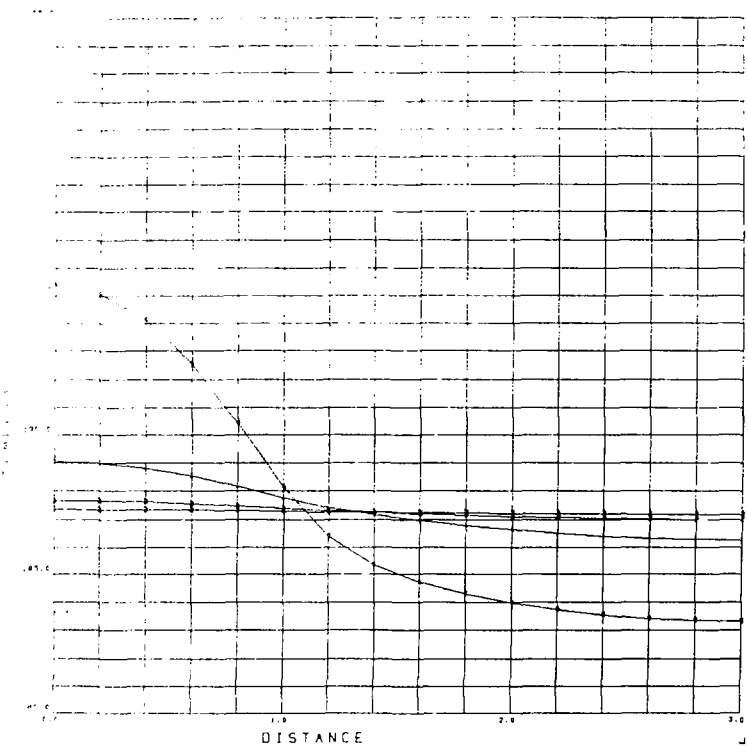


FIGURE 34. Temperature Profiles--Example Problem 6

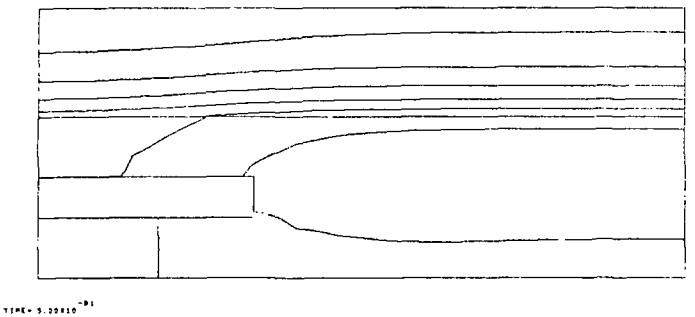


FIGURE 35. Contour Plot--Example Problem 6

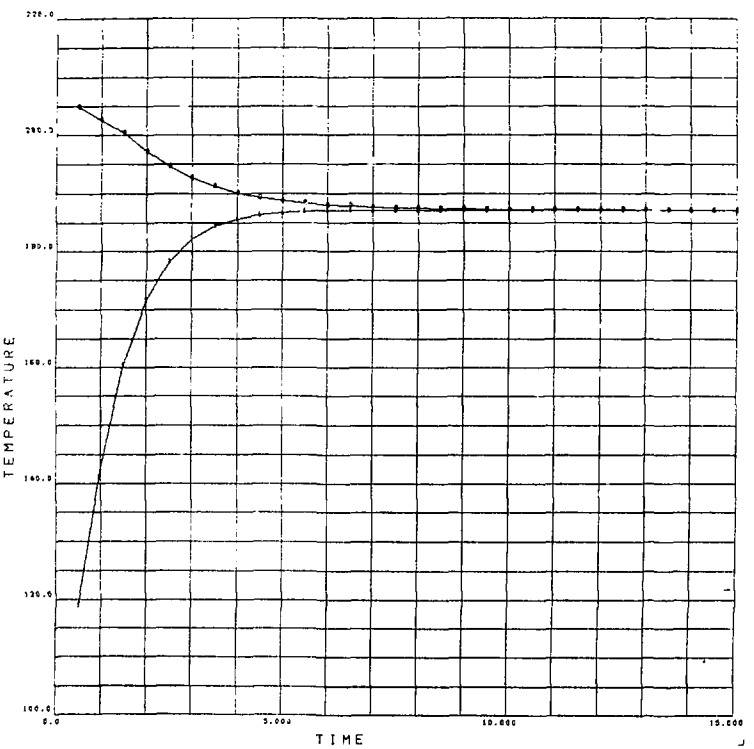


FIGURE 36. Time Histories, Special Points--Example Problem 6

APPENDIX A

CONSISTENT UNITS

The following list provides examples of consistent units for quantities encountered in the use of the COYOTE program:

Quantity	English	Metric	SI
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	calorie (cal)	Joules (J)
Temperature	Fahrenheit (F) or Rankine (R)	Centrigrade (C) or Kelvin (K)	Kelvin (K)
Density	lb_m/ft^3	gm/cm^3	kg/m^3
Specific Heat	Btu/lb_m-F	cal/gm-C	J/kg-K
Power	Btu/s	cal/s	J/s (Watt)
Heat Flux	Btu/ft^2-s	cal/cm^2-s	J/m^2-s
Heat Transfer Coefficient	Btu/ft^2-s-F	cal/cm^2-s-C	J/m^2-s-K
Thermal Conductivity	Btu/ft-s-F	cal/cm-s-C	J/m-s-K
Volume Heat Source	Btu/ft^3-s	cal/cm^3-s	J/m^3-s
σ	4.755×10^{-13} $\frac{Btu}{s\text{-}ft^2\text{-}R^4}$	1.355×10^{-12} $\frac{cal}{s\text{-}cm^2\text{-}K^4}$	5.6697×10^{-8} $\frac{J}{s\text{-}m^2\text{-}K^4}$

APPENDIX B

TIME STEP ESTIMATION

The analysis of a transient conduction problem requires the selection of a suitable time step for the integration procedure. The selection of too large a time step can result in a loss of the transient temperature response and the generation of only a steady state solution. An inappropriately small time step may produce nonphysical spatial oscillations in the early time temperature field due to the limited resolution ability (for temperature gradients) of the finite element mesh. Obviously, the judicious choice of a time step is important for both economy and accuracy of the analysis. The method for estimating a time step as outlined below is due to Nickell[22] and Levi[23] and is based on an analytic solution to the one-dimensional heat conduction equation.

In the following discussion, it is assumed that a finite element mesh has been constructed that will adequately model the thermal phenomena of interest (e.g., thermal shock problems will require a fine mesh near the boundary). For a given spatial model, a local characteristic length, Δx, is chosen based on element size. Typically, this characteristic length is measured normal to a boundary on which a temperature or heat flux (source) disturbance occurs. Based on the characteristic length, the local heat transfer coefficient, h, and the local thermal conductivity, a local Biot number ($Bi = h\Delta x/k$) can be computed. For prescribed temperature, heat flux, or heat source boundary conditions, the Biot number is assumed large.

In order to bound the thermal gradient that will occur during the first time step, the ratio of the temperature at a distance Δx (characteristic length) from the boundary to the boundary temperature is selected. Let this ratio be defined by $\theta = T(\Delta x)/T_{surface}$. Typical values for θ will range from 10 to 25%.

With the estimated values for local Biot number and temperature ratio, a local Fourier number ($Fo = \alpha\Delta t/\Delta x^2$) may be found from the charts in Figures 37 through 39. These charts are based on an analytic solution for one-dimensional conduction with a convective boundary condition.[24] A value for the local Fourier number and values for the characteristic length and thermal diffusivity, then allow a time step, Δt to be computed.

As an example of the procedure outlined above, consider the transient problem described in Example 2 of a previous section. The pertinent material properties (steel) were given as,

$$\rho = 7.689 \text{ gm/cm}^3 \quad ,$$

$$C_p = 0.46 \text{ W-s/gm-C} \quad ,$$

$$k = 0.22 \text{ W/cm-C} \quad ,$$

$$\alpha = 0.0622 \text{ cm}^2/\text{s} \quad .$$

The characteristic length Δx is based on the size of element number 1 (or any element along the outside edge of the bar, Figure 21) and is equal to $\Delta x = 0.35$ cm. Since the boundary temperature is specified, the Biot number is taken as infinite. Assuming $\theta = 10\%$, then Figure 39 yields a Fourier number of $Fo = 0.2$. Thus,

$$\Delta t = \frac{(0.2)(0.35)^2}{(0.0622)} = 0.393 \text{ s} \quad .$$

In the analysis of Example 2, a time step of $\Delta t = 0.4$ s was used in the first time integration interval.

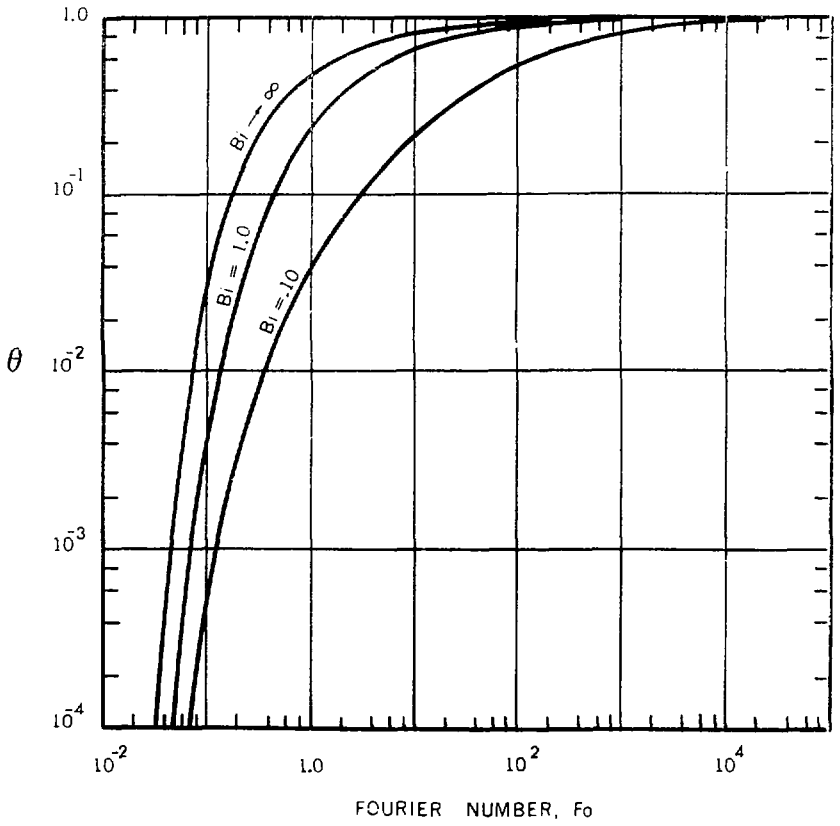


FIGURE 39. θ Versus Fourier Number--$10^{-1} \leq$ Bi $\leq \infty$

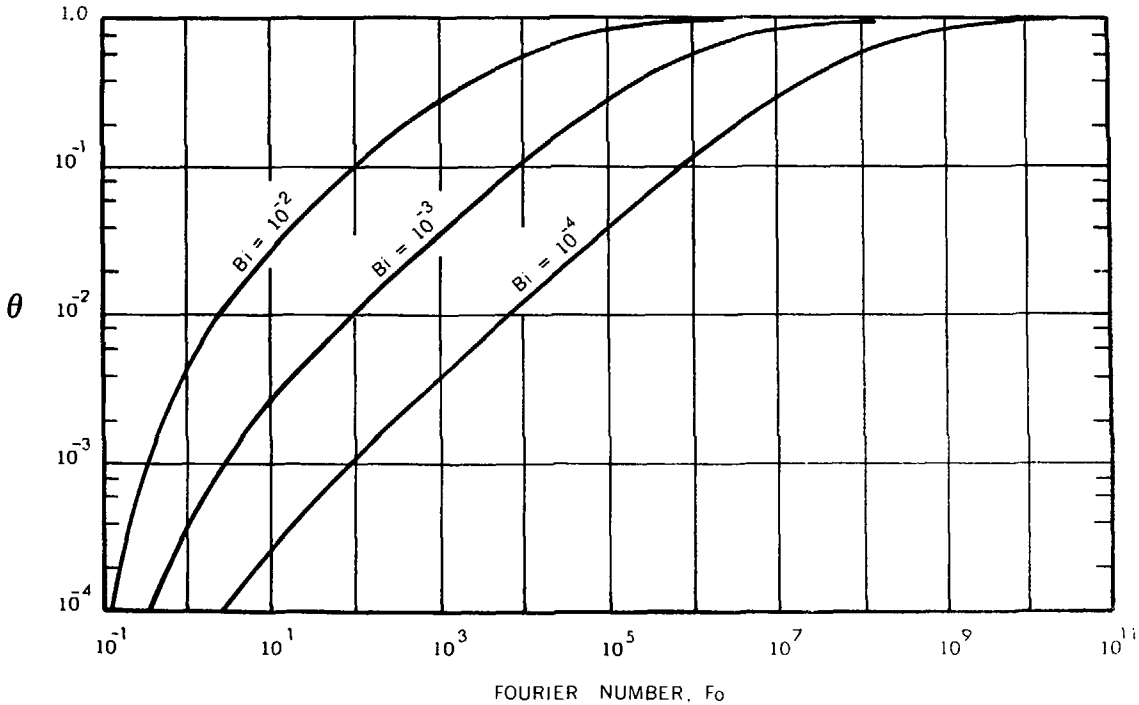


FIGURE 38. θ Versus Fourier Number--$10^{-1} \leq Bi \leq 10^{-2}$

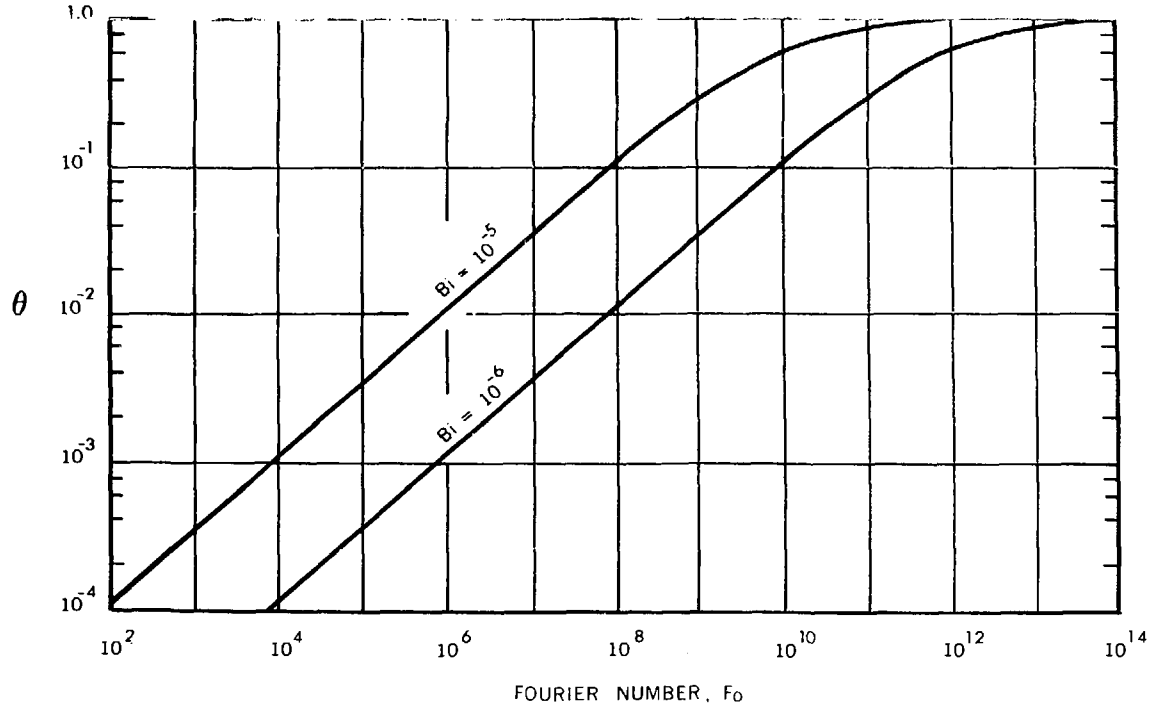


FIGURE 37. θ Versus Fourier Number--$10^{-6} \leq Bi \leq 10^{-5}$

REFERENCES

1. J. D. Gaski, D. R. Lewis, and L. R. Thompson, "Chrysler Improved Numerical Differencing Analyzer for 3rd Generation Computers (CINDA-3G)," TN-AP-67-287, Chrysler Corporation, Space Division, New Orleans, LA, October 1967.

2. D. K. Gartling, "MERLIN--A Computer Program to Transfer Data Between Finite Element Meshes," SAND81-0463, Sandia National Laboratories, Albuquerque, NM, September 1981.

3. D. K. Gartling, "NACHOS--A Finite Element Computer Program for Incompressible Flow Problems," SAND77-1333 and SAND77-1334, Sandia National Laboratories, Albuquerque, NM, April 1978.

4. D. K. Gartling and C. E. Hickox, "MARIAH--A Finite Element Computer Program for Incompressible Porous Flow Problems," SAND79-1622 and SAND79-1623, Sandia National Laboratories, Albuquerque, NM, August 1980.

5. R. S. Dunham and E. B. Becker, "TEXGAP--The Texas Grain Analysis Program," TICOM Report 73-1, Texas Institute for Computational Mechanics, Austin, TX, August 1973.

6. E. L. Wilson and R. E. Nickell, "Application of the Finite Element Method to Heat Conduction Analysis," Nuclear Engineering and Design, Vol. 4, 1966, pp. 276-286.

7. G. Comini, S. del Guidice, R. W. Lewis, and O. C. Zienkiewicz, "Finite Element Solution of Non-Linear Heat Conduction Problems with Special Reference to Phase Change," Int. J. Num. Meth. Engng., Vol. 8, 1974, pp. 613-624.

8. B. A. Finlayson, The Method of Weighted Residuals and Variational Principles, Academic Press, New York, NY, 1972.

9. O. C. Zienkiewicz, The Finite Element Method, McGraw-Hill, London, 1977.

10. W.-L. Wood and R. W. Lewis, "A Comparison of Time Marching Schemes for the Transient Heat Conduction Equation," Int. J. Num. Meth. Engng., Vol. 9, 1975, pp. 679-689.

11. J. H. Argyris, L. E. Vaz, and K. J. William, "Higher Order Methods for Transient Diffusion Analysis," Comp. Meth. Appl. Mech. Engr., Vol. 12, 1977, pp. 243-278.

12. R. E. Nickell and H. D. Hibbitt, "Thermal and Mechanical Analysis of Welded Structures," Nucl. Eng. Des., 32, 1975, pp. 110-120.

13. R. E. Nickell, "Applications of the Finite Element Method in Solid Mechanics, Fluid Mechanics, and Heat Transfer," Developments in Mechanics, Vol. 8, Proc. 14th Midwestern Mech. Conf., Norman, OK, 1975, pp. 599-626.

14. S. J. Womack, "Shape Function Techniques for Generating Finite Element Grids," TICOM Report 73-3, Texas Institute for Computational Mechanics, Austin, TX, 1973.

15. D. K. Gartling, "Texas Fluid Analysis Program--User's Manual," TICOM Report 75-2, Texas Institute for Computational Mechanics, Austin, TX, 1975.

16. I. Ergatoudis, B. M. Irons, and O. C. Zienkiewicz, "Curved Isoparametric, 'Quadrilateral', Elements for Finite Element Analysis," Int. J. Num. Meth. Engng., Vol. 4, 1968, pp. 31-42.

17. P. C. Hammer, O. P. Marlowe, and A. H. Stroud, "Numerical Integration over Simplexes and Cones," Math. Tables Aids Comp., Vol. 10, 1956, pp. 130-137.

18. B. M. Irons, "A Frontal Solution Program for Finite Element Analysis," Int. J. Num. Meth. Engng., Vol. 2, 1970, pp. 5-32.

19. S. L. Thompson, "RSCORS--A Revised SCORS Plot System," SAND77-1957, Sandia National Laboratories, Albuquerque, NM, May 1978.

20. S. L. Thompson, G. C. Padilla, and W. A. Sexson, "RSCORS Plotting at SNLA, Version 1.08," Internal Computer Document (unpublished), Sandia National Laboratories, Albuquerque, NM, January 1982.

21. R. W. Simons, "Remote Hard Copy Programming Reference Manual," *Internal Computer Document (unpublished), Sandia National Laboratories, Albuquerque, NM, September 1979.*

22. R. E. Nickell, private communication, 1977.

23. I. M. Levi, "User's Guide to the Transient Heat Conduction Finite Element Code HTCON," MM70-5424-17, Bell Telephone Laboratories Memorandum, Whippany, NJ, 1970.

24. H. S. Carslaw and J. C. Jaeger, Conduction of Heat in Solids, Oxford, NY. 1947.