SANDIA REPORT
SAND86—0594 • UC—32
Unlimited Release
Printed March 1987

PRONTO 2D
A Two-Dimensional Transient Solid Dynamics Program

L. M. Taylor, D. P. Flanagan

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-76DP00789

DISCLAIMER

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation

Printed in the United States of America
Available from
National Technical Information Service
U S Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes
Printed copy A11
Microfiche copy A01

SAND--86-0594

DE87 007823

SAND86-0594
Unlimited Release
Printed March 1987

Distribution
UC-32

PRONTO 2D

A TWO-DIMENSIONAL TRANSIENT SOLID DYNAMICS PROGRAM

L. M. Taylor and D. P. Flanagan
Applied Mechanics Division III
Sandia National Laboratories, Albuquerque, NM 87185

ABSTRACT

PRONTO 2D is a two-dimensional transient solid dynamics code for analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates. This Lagrangian finite element program uses an explicit time integration operator to integrate the equations of motion. Four node uniform strain quadrilateral elements are used in the finite element formulation. A number of new numerical algorithms which have been developed for the code are described in this report. An adaptive time step control algorithm is described which greatly improves stability as well as performance in plasticity problems. A robust hourglass control scheme which eliminates hourglass distortions without disturbing the finite element solution is included. All constitutive models in PRONTO are cast in an unrotated configuration defined using the rotation determined from the polar decomposition of the deformation gradient. An accurate incremental algorithm was developed to determine this rotation and is described in detail. A robust contact algorithm was developed which allows for the impact and interaction of deforming contact surfaces of quite general geometry. A number of numerical examples are presented to demonstrate the utility of these algorithms.

ACKNOWLEDGMENTS

From the beginning of this project we were given encouragement and support from a number of people. Perhaps the most significant factor in the development of PRONTO was the rich research environment found at Sandia. We would particularly like to express our gratitude to our supervisor, Johnny Biffle, for supporting us in this endeavor.

A number of people used PRONTO in its early stages and pointed out numerous bugs and deficiencies. Joel Miller, Greg Sjaardema, Steve Burchett, and Jerry Wellman were particularly helpful and patient in debugging the early versions of the code. To them and all others who believed us when we said we just fixed the proverbial "last bug" in the slide line logic, we extend a heartfelt thanks for their patience and sense of humor.

We owe a special word of thanks to Marlin Kipp for taking the time to discuss equations of state with us and help us to decipher the baffling number of secrets required to successfully implement them in a production finite element code. We have borrowed deeply and with great humility from the descriptions of equations of state found in the WONDY and TOODY finite difference codes and owe the authors of those codes a heartfelt thanks.

Marlin Kipp and Greg Sjaardema reviewed this document and suggested numerous helpful and creative changes to make it more readable. We appreciate the effort it took to read such a long manuscript in such detail.

Last, and certainly not least, we want to thank Diane Kolb for her typing of this manuscript. Her cheerful and patient attention to the manuscript under its continual revision due to our oversights and mistakes will always be remembered.

CONTENTS

FIGURES

TABLES

1.0 INTRODUCTION

PRONTO 2D is a finite element FORTRAN program for the analysis of the two-dimensional response of solid bodies to transient dynamic loading. The program includes nonlinear constitutive models, and accurately analyzes large deformations which may lead to geometric nonlinearities. PRONTO is a powerful tool for analyzing a wide variety of problems, including classes of problems in impact dynamics, rock blasting, and accident analyses.

PRONTO is a new generation code following in a long line of finite difference and finite element programs which were developed in the last thirty years to analyze transient solid dynamics problems. These original codes had their origins in the Manhattan Project and the work of Von Neumann [1]. They are often referred to as "wave codes" and were developed at the national laboratories for numerous weapons projects. All of these codes use an explicit time integration operator to advance the equations of motion from the initial state.

The first general finite difference FORTRAN codes were the WONDY [2,3] and TOODY [4] codes developed at Sandia National Laboratories and the HEMP [5] code developed at Lawrence Livermore Laboratories. The HONDO [6] code developed at Sandia National Laboratories was the first wave code to use the finite element method. HONDO drew heavily upon the experience of the finite difference code developers; many of the algorithms in HONDO came directly from the finite difference literature. The more robust numerical simulation capability of the finite element method allowed the introduction of many new innovations in HONDO. These include multiple material libraries and general surface contact algorithms. The DYNA [8] family of codes descended directly from HONDO. DYNA made a significant step forward by structuring the code to take advantage of the vector processors available on new generation computers. As a result, DYNA achieved a four-fold increase in speed over the HONDO code. DYNA also significantly expanded the material library and was the first finite element wave code to implement hydrodynamic equations of state. Another general purpose finite element program which has seen widespread use is the EPIC [9] series of codes. The EPIC codes contain an innovative algorithm for the impact and erosion of contact surfaces for

penetration problems. We were able to develop PRONTO into a production tool in a very short time as a direct result of the rich algorithmic environment which we inherited from the developers who came before us.

We developed a flexible problem-oriented language for the input to PRONTO which allows the user to define a complex mechanics problem with a few concise commands. Experience with the code has shown that after a user has gained some experience with the code, reference to the user's instructions (Appendix A) is seldom needed. There is no reference to node or element numbers in the problem definition. All boundary conditions are defined through the concept of node and element side sets which are defined using the GENESIS [10] mesh definition data base.

PRONTO contains no mesh generation or post-processing capabilities; it relies on external mesh generators and external post-processors. The program writes the SEACO [11] plotting data base for graphical display of the results. The form of the GENESIS and SEACO data bases are given in Appendices D and E, respectively.

The development of PRONTO was motivated by the need for a code which could serve as a testbed for research into numerical algorithms and new constitutive models for nonlinear materials. Towards this goal, the code contains a well documented and easy to use interface for implementing new constitutive models (Appendix C). Complete documentation of the code architecture and computer storage requirements is provided in Appendix B.

PRONTO is written in completely standard FORTRAN [7]. Any system dependent coding such as the determination of the date or the memory management is part of the SUPES [12] package. Figure 1.1 shows the architectural layout of PRONTO. Some minor utility subroutines have been omitted from the figure. These are routines such as GATHER, which are called from numerous locations. The only input/output units which PRONTO uses are 5, 6, 9, 11, 30 and 32. Their use is described in Table 1.1.

TABLE 1.1
INPUT/OUTPUT UNITS

Unit	Use
5	Formatted input instructions for PRONTO 2D
6	Formatted output from PRONTO 2D
9	Unformatted GENESIS mesh file
11	Unformatted SEACO post-processing file
30	Unformatted restart output file
32	Unformatted restart input file

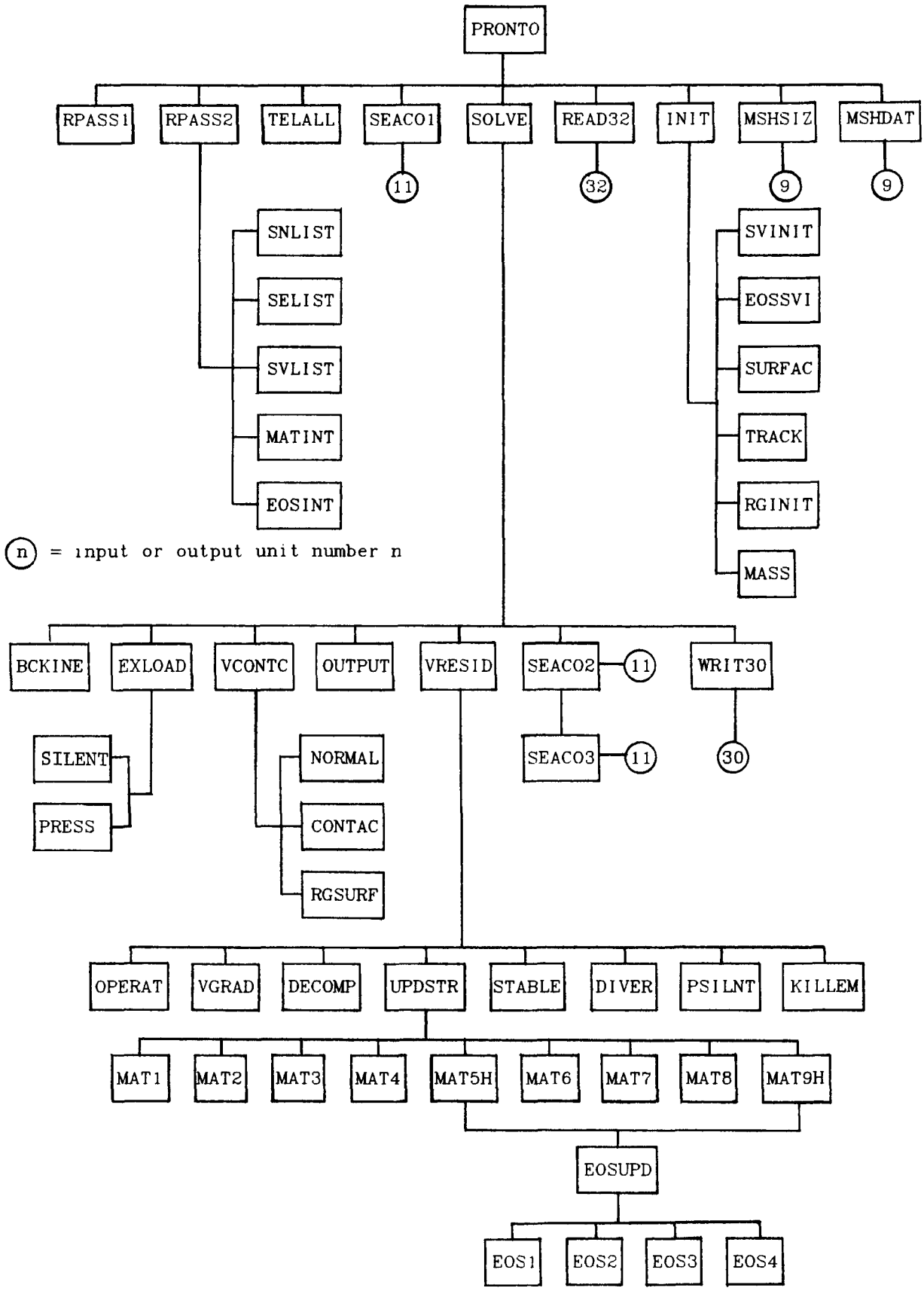


Figure 1.1. Architectural Layout of the PRONTO 2D Code

2.0 GOVERNING EQUATIONS

In this chapter, we present the underlying continuum mechanics concepts which are necessary to follow the development of the numerical algorithms in the following chapters. Bold face characters denote tensors. The order of the tensor is implied by the context of the equation.

2.1 Kinematics

A material point in the reference configuration B_0 with position vector $\mathbf{X}$ occupies position $\mathbf{x}$ at time t in the deformed configuration B. Hence we write $\mathbf{x} = \chi(\mathbf{X},t)$. The motion from the original configuration to the deformed configuration shown in Figure 2.1.1 has a deformation gradient F given by

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} , \quad \det(\mathbf{F}) > 0 \tag{2.1.1}$$

Applying the polar decomposition theorem to F :

$$\mathbf{F} = \mathbf{V}\,\mathbf{R} = \mathbf{R}\,\mathbf{U} \tag{2.1.2}$$

where $\mathbf{V}$ and $\mathbf{U}$ are the symmetric, positive definite left and right stretch tensors, respectively, and $\mathbf{R}$ is a proper orthogonal rotation tensor. Figure 2.1.1 illustrates the intermediate orientations defined by the two alternate decompositions of F defined by Equation (2.1.2). The determination of $\mathbf{R}$ as defined by Equation (2.1.2) presents a significant numerical challenge. In Section 3.3, we describe the incremental algebraic algorithm that we use to determine $\mathbf{R}$.

The velocity of the material point $\mathbf{X}$ is written as $\mathbf{v} = \dot{\mathbf{x}}$ where the superposed dot indicates time differentiation holding the material point fixed. The velocity gradient is denoted by L and may be expressed as

$$\mathbf{L} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \frac{\partial \mathbf{v}}{\partial \mathbf{X}} \frac{\partial \mathbf{X}}{\partial \mathbf{x}} = \dot{\mathbf{F}}\,\mathbf{F}^{-1} . \tag{2.1.3}$$

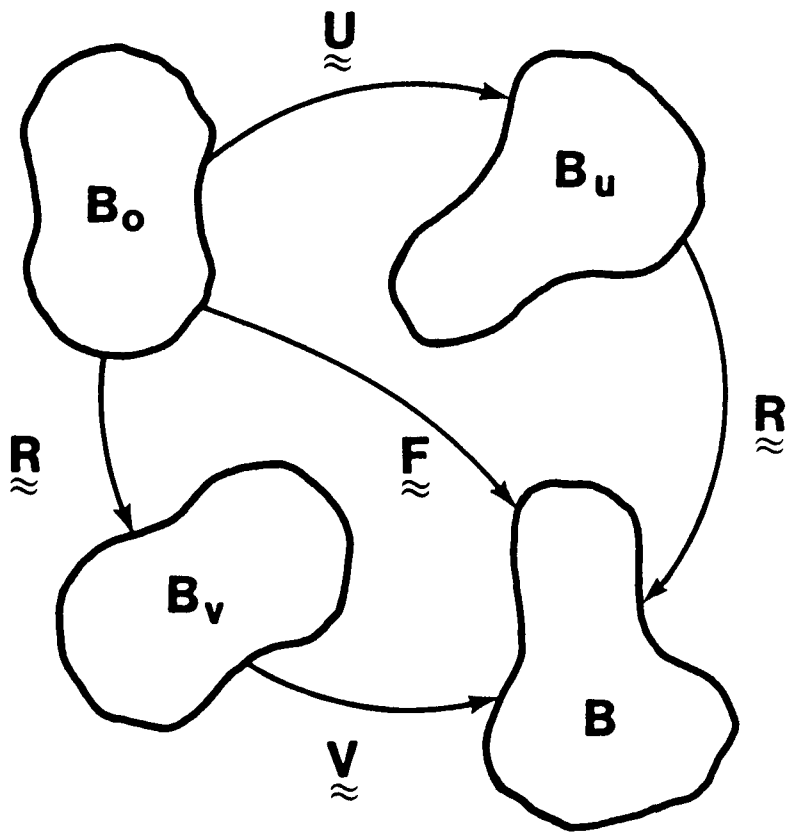


Figure 2.1.1. Original, Deformed and Intermediate Configurations of a Body

The velocity gradient can be written in terms of the symmetric (**D**) and antisymmetric (**W**) parts, respectively,

$$\mathbf{L} = \mathbf{D} + \mathbf{W} \ . \tag{2.1.4}$$

Using the right decomposition from Equation (2.1.2) in Equation (2.1.3) gives

$$\mathbf{L} = \dot{\mathbf{R}}\,\mathbf{R}^{T} + \mathbf{R}\,\dot{\mathbf{U}}\,\mathbf{U}^{-1}\,\mathbf{R}^{T} \ . \tag{2.1.5}$$

Dienes [13] denoted the first term on the right-hand side of Equation (2.1.5) by $\boldsymbol{\Omega}$:

$$\boldsymbol{\Omega} = \dot{\mathbf{R}}\,\mathbf{R}^{T} \ . \tag{2.1.6}$$

Both **W** and $\boldsymbol{\Omega}$ are antisymmetric and represent a rate of rotation (or angular velocity) about some axes. In general, $\boldsymbol{\Omega} \neq \mathbf{W}$. The difference arises when the last term of Equation (2.1.5) is not symmetric. The symmetric part of $\dot{\mathbf{U}}\,\mathbf{U}^{-1}$ is the unrotated deformation rate tensor **d** as defined below (note that both $\dot{\mathbf{U}}$ and $\mathbf{U}^{-1}$ are symmetric).

$$\mathbf{d} = \frac{1}{2}\,(\dot{\mathbf{U}}\,\mathbf{U}^{-1} + \mathbf{U}^{-1}\,\dot{\mathbf{U}}) = \mathbf{R}^{T}\,\mathbf{D}\,\mathbf{R} \ . \tag{2.1.7}$$

There are two possible cases which can cause rotation of a material line element: rigid body rotation and shear. Since total shear vanishes along the axes of principal stretch, the rotation of these axes defines the total rigid body rotation of a material point.

It is a simple exercise in vector analysis to show that Equation (2.1.6) represents the rate of rigid body rotation at a material point (as shown by Dienes [13]). It is equally simple to show that **W** represents the rate of rotation of the principal axes of the rate of deformation **D**. Since **D** and **W** have no sense of the history of deformation, they are not sufficient to define the rate of rotation in a finite deformation context.

Line elements where the rate of shear vanishes rotate solely due to rigid body rotations. These line elements are along the principal axes of $\dot{\mathbf{U}}$. We will apply a similar observation below as we derive Dienes' [13] expression for calculating $\boldsymbol{\Omega}$.

Using the left decomposition of Equation (2.1.2) in Equation (2.1.3) gives

$$\mathbf{L} = \dot{\mathbf{V}}\,\mathbf{V}^{-1} + \mathbf{V}\,\boldsymbol{\Omega}\,\mathbf{V}^{-1} \ . \tag{2.1.8}$$

Postmultiplying by $\mathbf{V}$ yields an expression which defines the decomposition of $\mathbf{L}$ into $\mathbf{V}$ and $\boldsymbol{\Omega}$:

$$\mathbf{L}\,\mathbf{V} = \dot{\mathbf{V}} + \mathbf{V}\,\boldsymbol{\Omega} \ . \tag{2.1.9}$$

When the dual vector of the above expression is taken, the symmetric $\dot{\mathbf{V}}$ vanishes to yield a set of three linear equations for the three independent components of $\boldsymbol{\Omega}$.

The antisymmetric part of a tensor may be expressed in terms of its dual vector and the permutation tensor e_{ijk}. Define the following dual vectors:

$$\omega_i = e_{ijk}\,\Omega_{jk} \tag{2.1.10}$$

$$w_i = e_{ijk}\,W_{jk} \ . \tag{2.1.11}$$

Using Equations (2.1.4), (2.1.10), and (2.1.11) in Equation (2.1.9) results in the expression that Dienes [13] gave for determining $\boldsymbol{\Omega}$ from $\mathbf{W}$ and $\mathbf{V}$;

$$\boldsymbol{\omega} = \mathbf{w} - 2[\mathbf{V} - \mathbf{I}\ \mathrm{tr}(\mathbf{V})]^{-1}\,\mathbf{z} \tag{2.1.12}$$

where

$$z_i = e_{ijk} V_{jm} D_{mk} . \tag{2.1.13}$$

We observe from the above expressions that $\boldsymbol{\Omega} = \mathbf{W}$ if and only if the product $\mathbf{V}\ \mathbf{D}$ is symmetric. This condition requires that the principal axes of the deformation rate $\mathbf{D}$ coincide with the principal axes of the current stretch $\mathbf{V}$. Clearly, a pure rotation is a special case of this condition since $\mathbf{D}$, and consequently (2.1.13), vanish.

2.2 Stress and Strain Rates

Our constitutive model architecture is posed in terms of the conventional Cauchy stress, but we adopt the approach of Johnson and Bammann [14] and define a Cauchy stress in the unrotated configuration. The reader seeking more detail than is presented here should see Flanagan and Taylor [15]. The "true" stress in the deformed configuration is denoted by $\mathbf{T}$. The Cauchy stress in the unrotated configuration is denoted by $\boldsymbol{\sigma}$. These two stress measures are related by

$$\boldsymbol{\sigma} = \mathbf{R}^T\ \mathbf{T}\ \mathbf{R} . \tag{2.2.1}$$

Each material point in the unrotated configuration has its own reference frame which rotates such that the deformation in this frame is a pure stretch. Then $\mathbf{T}$ is simply the tensor $\boldsymbol{\sigma}$ in the fixed global reference frame. The conjugate strain rate measures to $\mathbf{T}$ and $\boldsymbol{\sigma}$ are $\mathbf{D}$ and $\mathbf{d}$, respectively. These strain rates were defined by Equations (2.1.4) and (2.1.7), respectively.

The Principal of Material Frame Indifference (or objectivity) stipulates that a constitutive law must be insensitive to a change of reference frame [16]. This requires that only objective quantities may be used in a constitutive law. An objective quantity is one which transforms in the same manner as the energy conjugate stress and strain rate pair under a superposed rigid body motion. The fundamental advantage of the unrotated stress

over the true stress is that the material derivative of $\boldsymbol{\sigma}$ is objective, whereas the material derivative of $\mathbf{T}$ is not.

The Jaumann rate defined below is frequently used in constitutive relationships to resolve the need for an objective rate of Cauchy stress.

$$\hat{\mathbf{T}} = \dot{\mathbf{T}} - \mathbf{W}\,\mathbf{T} + \mathbf{T}\,\mathbf{W} \,. \tag{2.2.2}$$

It is an easy task to show that the Jaumann rate is objective.

A similar stress rate, called the Green–Naghdi rate by Johnson and Bammann [14], can be derived by transforming the rate of the unrotated Cauchy stress to the fixed global frame as follows:

$$\hat{\boldsymbol{\sigma}} = \mathbf{R}\,\dot{\boldsymbol{\sigma}}\,\mathbf{R}^{\mathsf{T}} = \dot{\mathbf{T}} - \boldsymbol{\Omega}\,\mathbf{T} + \mathbf{T}\,\boldsymbol{\Omega} \,. \tag{2.2.3}$$

The Jaumann rate and the Green–Naghdi rate are very similar in form. The important difference between the two is that the Green–Naghdi rate is kinematically consistent with the rate of Cauchy stress, while the Jaumann rate is not. By this statement we mean that $\hat{\boldsymbol{\sigma}}$ is identical to $\dot{\mathbf{T}}$ in the absence of rigid body rotations. It is clear that $\hat{\mathbf{T}}$ need not equal $\dot{\mathbf{T}}$ under the same conditions since $\mathbf{W}$ need not vanish with rigid body rotations.

The simple shear problem presented by Dienes [13] serves as an excellent demonstration of the symptoms which can occur due to the deficiency of the Jaumann rate. Figure 2.2.1 shows a body which undergoes the following motion:

$$x(t) = X + k\,t\,Y \,, \qquad y(t) = Y \,, \qquad z(t) = Z \,. \tag{2.2.4}$$

Dienes applied a simple linear isotropic hypoelastic material law to both the Jaumann rate (2.2.2) and the Green–Naghdi rate (2.2.3). The analytic solution for the true stresses as a function of time using the Jaumann rate is shown in Figure 2.2.1. The Green–Naghdi rate solution is shown in Figure 2.2.2 and demonstrates a monotonic increase in stress with increasing shear strain, while the Jaumann rate results in a harmonic oscillation of the

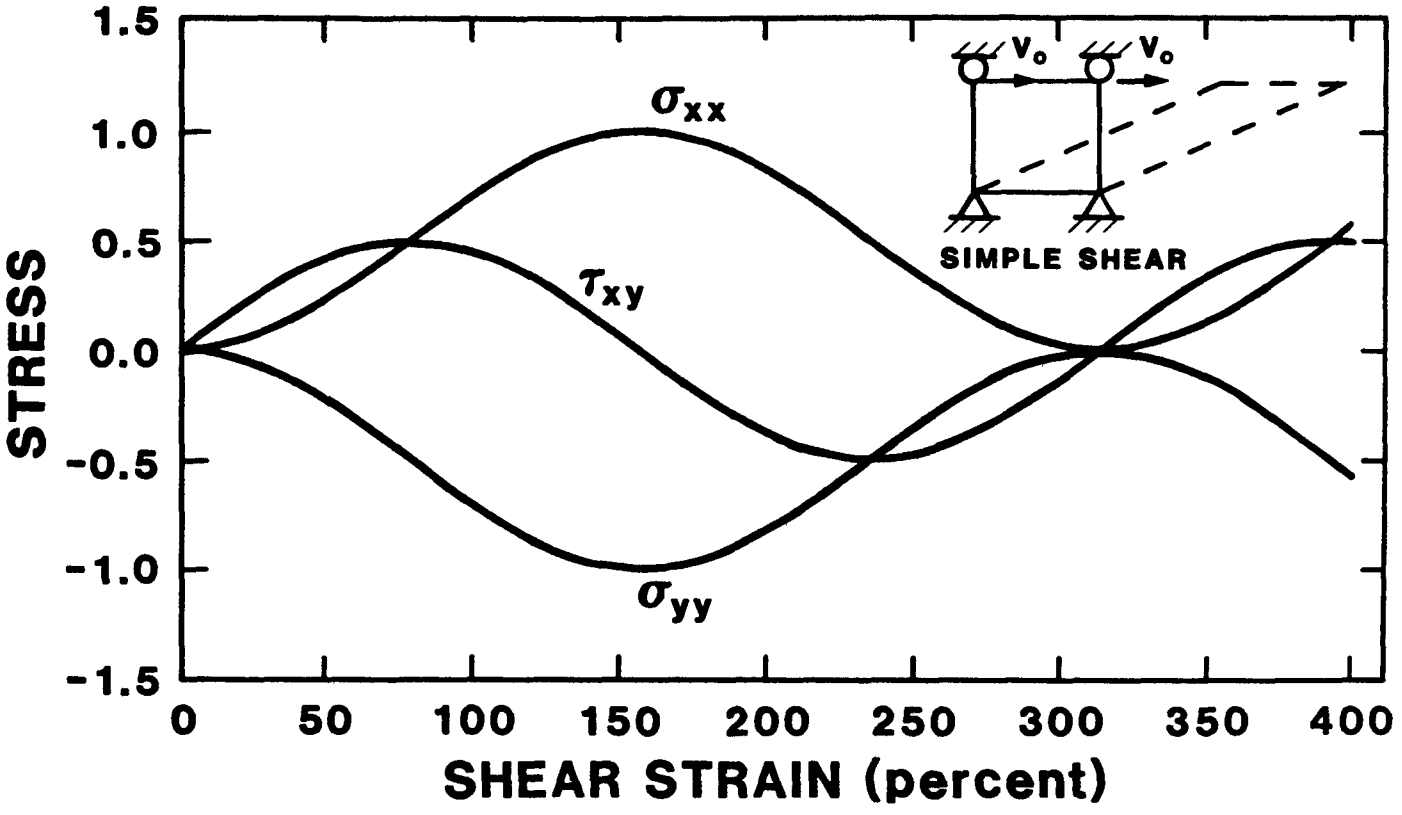


Figure 2.2.1. Computed Stress-Strain Curves for a Body Undergoing Simple Shear Using the Jaumann Rate

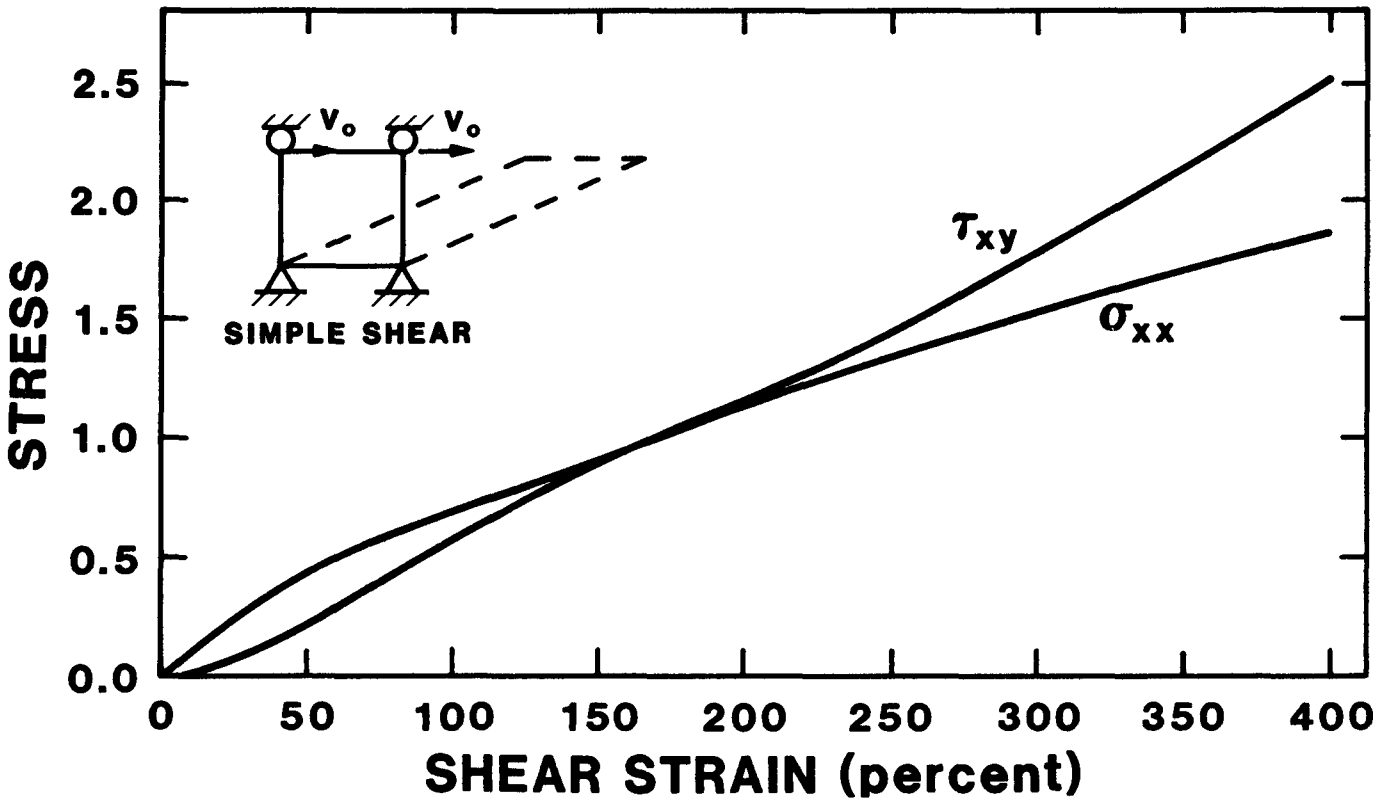


Figure 2.2.2. Computed Stress-Strain Curves for a Body Undergoing Simple Shear Using the Green-Naghdi Rate

stress. The reason that the Jaumann rate produces this oscillation in stress is that $\mathbf{W}$ gives a constant rate of rotation for the motion defined by Equation (2.2.4), while $\boldsymbol{\Omega}$ vanishes with time. Clearly the body experiences rotations which diminish over time, but the Jaumann rate continues to drive the stress convection terms at a constant rate. This leads to the oscillatory behavior of the stresses shown in Figure 2.2.1.

A distinct advantage of the unrotated reference frame is that all constitutive models are cast without regard to finite rotations. This greatly simplifies the numerical implementation of new constitutive models. The rotations of global state variables (e.g., stress and strain) are dealt with on a global level which insures that all constitutive models are consistent. Internal state variables (e.g., backstress) see no rotations whatsoever.

The drawback to working in the unrotated reference frame is that we must accurately determine the rotation tensor, $\mathbf{R}$, which is not a straightforward numerical calculation. We present an incremental, algebraic algorithm to accomplish this task in Section 3.4.

2.3 Fundamental Equations

The equations of motion for the body are the momentum equations

$$\operatorname{div} \mathbf{T} - \rho \ddot{\mathbf{u}} + \rho \mathbf{f}_B = 0 \ . \tag{2.3.1}$$

where ρ is the mass density per unit volume, $\boldsymbol{\mu}$ is the acceleration of the material point, and $\mathbf{f}_B$ is a specific body force vector.

We seek the solution to Equation (2.3.1) subject to the boundary conditions

$$\mathbf{u} = \mathbf{f}(t) \text{ on } S_u \tag{2.3.2}$$

where S_u represents the portion of the boundary on which kinematic quantities are specified (displacement, velocity, and acceleration). In addition to satisfying the kinematic boundary conditions given by (2.3.2), we must satisfy the traction boundary conditions

$$\mathbf{T}\cdot\mathbf{n} = \mathbf{s}(t) \text{ on } S_T \tag{2.3.3}$$

where S_T represents the portion of the boundary on which tractions are specified. The boundary of the body is given by the union of S_u and S_T, and we note that for a valid mechanics problem S_u and S_T have a null intersection.

The jump conditions at all contact discontinuities must satisfy the relation

$$(\mathbf{T}^+ - \mathbf{T}^-)\cdot\mathbf{n} = 0 \text{ on } S_c \tag{2.3.4}$$

where S_c represents the contact surface intersection and the subscripts "+" and "−" denote different sides of the contact surface.

The Lagrangian form of the continuity equation is written as

$$\dot{\rho} - \rho \text{ tr } \mathbf{D} = 0 \ . \tag{2.3.5}$$

This is satisfied trivially in our formulation since we do not allow mass transport. Equation (2.3.5) degenerates to

$$\rho V = \rho_o V_o \tag{2.3.6}$$

where V is the volume and the subscript "o" denotes a reference configuration.

The conservation of energy principle equates the increase in internal energy per unit volume to the rate at which work is being done by the stresses plus the rate at which heat is being added. In the absence of heat conduction

$$\dot{E}_v = \rho \frac{\partial E_m}{\partial t} = \boldsymbol{\sigma}:\mathbf{d} + \rho \dot{Q} \qquad (2.3.7)$$

where E_v is the energy per unit volume, E_m is the energy per unit mass, and $\dot{Q}$ is the heat rate per unit mass. The stress $\boldsymbol{\sigma}$ and the strain rate $\mathbf{d}$ were discussed in the Section 2.2.

3.0 NUMERICAL FORMULATION

In this chapter, we describe the finite element formulation of the problem and the numerical algorithms required to perform the spatial and temporal integration of the equations of motion.

3.1 Four Node Uniform Strain Element

The 4-node two-dimensional isoparametric element is widely used in computational mechanics. Optimal integration schemes for these elements, however, present a difficult dilemma. A one point integration of the element under-integrates the element resulting in a rank deficiency for the element which manifests itself in spurious zero energy modes, commonly referred to as hourglass modes. A two-by-two integration of the element over-integrates the element and can lead to serious problems of element locking in fully plastic and incompressible problems. The four point integration also carries a tremendous computational penalty compared to the one point rule. We use the one point integration of the element and implement an hourglass control scheme to eliminate the spurious modes. The development presented below follows directly from Flanagan and Belytschko [17]. We assume that the reader is somewhat familiar with the finite element method and will not go into a complete description of the method. The reader can consult numerous texts on the method [41].

The quadrilateral element relates the spatial coordinates x_i to the nodal coordinates x_{iI} through the isoparametric shape functions ϕ_I as follows:

$$x_i = x_{iI}\, \phi_I(\xi,\eta) \tag{3.1.1}$$

In accordance with indicial notation convention, repeated subscripts imply summation over the range of that subscript. The lowercase subscripts have a range of two corresponding to the spatial coordinate directions. Uppercase subscripts have a range of four, corresponding to the element nodes.

The same shape functions are used to define the element displacement field in terms of the nodal displacements u_{iI}:

$$u_i = u_{iI}\,\phi_I \tag{3.1.2}$$

Since the same shape functions apply to both spatial coordinates and displacements, their material derivative (represented by a superposed dot) must vanish. Hence, the velocity field may be given by

$$\dot{u}_i = \dot{u}_{iI}\,\phi_I \tag{3.1.3}$$

and likewise for the acceleration field

$$\ddot{u}_i = \ddot{u}_{iI}\,\phi_I \tag{3.1.4}$$

The velocity gradient tensor, $\mathbf{L}$, is defined in terms of nodal velocities as

$$L_{ij} = \dot{u}_{i,j} = \dot{u}_{iI}\,\phi_{I,j} \tag{3.1.5}$$

By convention, a comma preceding a lowercase subscript denotes differentiation with respect to the spatial coordinates (e.g., $\dot{u}_{i,j}$ denotes $\partial\dot{u}_i/\partial x_j$).

The 2-D isoparametric shape functions map the unit square in ξ-η to an arbitrary quadrilateral in x-y, as shown in Figure 3.1.1. We choose to center the unit square at the origin in ξ-η space so that the shape functions may be conveniently expanded in terms of an orthogonal set of base vectors, given in Table 3.1, as follows:

$$\phi_I = \frac{1}{4}\,\Sigma_I + \frac{1}{2}\,\xi\Lambda_{1I} + \frac{1}{2}\,\eta\Lambda_{2I} + \xi\eta\Gamma_I \tag{3.1.6}$$

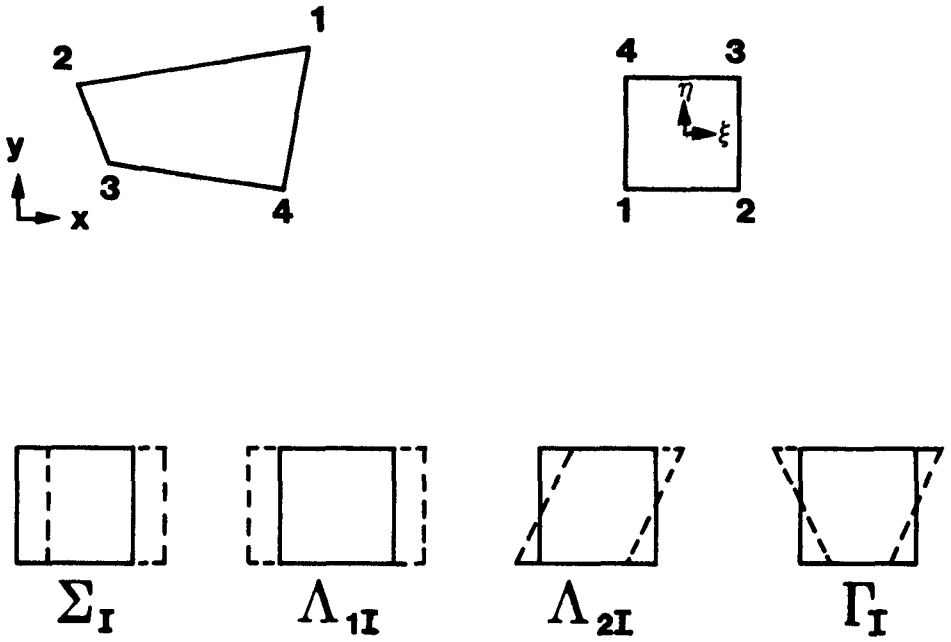


Figure 3.1.1. Mode Shapes for the Four-Node Constant Strain Quadrilateral Element

TABLE 3.1

node	ξ	η	Σ_I	Λ_{1I}	Λ_{2I}	Γ_I
1	-.5	-.5	1	-1	-1	1
2	.5	-.5	1	1	-1	-1
3	.5	.5	1	1	1	1
4	-.5	.5	1	-1	1	-1

The above vectors represent the displacement modes of a unit square. The first vector, Σ_I, accounts for rigid body translation. We call Σ the summation vector since it may be employed in indicial notation to represent the algebraic sum of a vector.

The linear base vectors Λ_{iI} may be readily combined to define the uniform normal strains and shear strain in the element. We refer to Λ_{iI} as the volumetric base vectors since, as we will illustrate below, they are the only base vectors which appear in the element area expression.

The last vector, Γ_I, gives rise to linear strain modes which are neglected in the uniform strain integration. This vector defines the hourglass patterns for a unit cube. The displacement modes represented by the vectors in Table 3.1 are also shown in Figure 3.1.1.

3.1.1 Plane Strain Case

In the finite element method, we replace the momentum Equation (2.3.1) with a weak form of the equation. Using the principle of virtual work, we write the weak form of the equation as

$$\sum_e \int_{V_e} (T_{ij,j} + \rho b_i - \rho \ddot{u}_i)\, \delta u_i \, dV = 0 \qquad (3.1.7)$$

where δu_i represents an arbitrary virtual displacement field, with the same interpolation as Equation (3.1.2), which satisfies the kinematic constraints. In plane strain, the thickness of the body is considered uniform and arbitrary, and therefore can be eliminated from the preceding expression. Integrating by parts and applying Gauss' divergence theorem to Equation (3.1.7) then gives

$$\sum_e \left[\int_{S_e} T_{iJ} n_J \delta u_i \, d\ell - \int_{A_e} T_{iJ} \delta u_{i,J} \, dA + \int_{A_e} \rho b_i \delta u_i \, dA - \int_{A_e} \rho \ddot{u}_i \delta u_i \, dA \right] = 0 \quad (3.1.8)$$

The summation symbol represents the assembly of element force vectors into a global nodal force array. We assume that the reader understands the details of this assembly; we will not discuss it further in this document.

The second integral in the preceding equation is used to define the element internal force vector f_{iI} as

$$\delta u_{iI} \, f_{iI} = \int_{A_e} T_{iJ} \, \delta u_{i,J} \, dA \quad (3.1.9)$$

The first and third integrals define the external force vector, and the fourth integral defines the inertial response.

We perform one point integration by neglecting the nonlinear portion of the element displacement field, thereby considering a state of uniform strain and stress. The preceding expression is approximated by

$$f_{iI} = \bar{T}_{iJ} \int_{A_e} \phi_{I,J} \, dA \quad (3.1.10)$$

where we have eliminated the arbitrary virtual displacements, and $\bar{T}_{iJ}$ represents the assumed uniform stress field which will be referred to as the mean stress tensor. By neglecting the nonlinear displacements, we have assumed that the mean stresses depend only on the mean strains. Mean kinematic quantities are defined by integrating over the element as follows:

$$\dot{\bar{u}}_{i,j} = \frac{1}{A}\int_V \dot{u}_{i,j}\, dA \ . \tag{3.1.11}$$

We now define the discrete gradient operator as

$$B_{iI} = \int_A \phi_{I,i}\, dA \ . \tag{3.1.12}$$

The mean velocity gradient, applying Equation (3.1.5), is given by

$$\dot{\bar{u}}_{i,j} = \frac{1}{A}\, \dot{u}_{iI}\, B_{jI} \ . \tag{3.1.13}$$

Combining Equations (3.1.10) and (3.1.12), we may express the nodal forces by

$$f_{iI} = \bar{T}_{ij}\, B_{jI} \ . \tag{3.1.14}$$

Computing nodal forces with this integration scheme requires evaluation of the gradient operator and the element area. These two tasks are linked since

$$x_{i,j} = \delta_{ij} \tag{3.1.15}$$

where δ_{ij} is the Kroneker delta. Equations (3.1.1), (3.1.12), and (3.1.15) yield

$$x_{iI}\, B_{jI} = \int_V (x_{iI}\, \phi_I)_{,j}\, dA = A\delta_{ij} \tag{3.1.16}$$

Consequently, the gradient operator may be expressed by

$$B_{iI} = \frac{\partial A}{\partial x_{iI}} \tag{3.1.17}$$

To integrate the element area in closed form, we use the Jacobian of the isoparametric transformation to transform the integral in x–y space to an integral over the unit square:

$$A = \int_{-1/2}^{+1/2} \int_{-1/2}^{+1/2} J \, d\eta \, d\xi \tag{3.1.18}$$

where

$$J = \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \xi} \tag{3.1.19}$$

Therefore, Equation (3.1.18) can be written as

$$A = x_I \, y_J \, C_{IJ} \tag{3.1.20}$$

where

$$C_{IJ} = \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} \left(\frac{\partial \phi_I}{\partial \xi} \frac{\partial \phi_J}{\partial \eta} - \frac{\partial \phi_I}{\partial \eta} \frac{\partial \phi_J}{\partial \xi} \right) d\eta \, d\xi \tag{3.1.21}$$

In light of Equation (3.1.6), the above integration involves at most bilinear functions. Therefore, only the constant term does not vanish and the integration yields

$$C_{IJ} = \frac{1}{4} \left(\Lambda_{1I} \, \Lambda_{2J} - \Lambda_{2I} \, \Lambda_{1J} \right) \tag{3.1.22}$$

Note that C_{IJ} is antisymmetric:

$$C_{IJ} = -C_{IJ} \tag{3.1.23}$$

Evaluating equation (3.1.22), we obtain the following explicit representation for C_{IJ}:

$$C_{IJ} = \frac{1}{2} \begin{bmatrix} 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \end{bmatrix} \tag{3.1.24}$$

Substituting the above expression into Equation (3.1.20), we obtain the familiar expression for the area of a quadrilateral:

$$A = \frac{1}{2}\left[(x_3 - x_1)(y_4 - y_2) + (x_2 - x_4)(y_3 - y_1)\right] \tag{3.1.25}$$

Using this result in Equation (3.1.17), the B matrix may be expressed as

$$B_{1J} = C_{IJ} \begin{Bmatrix} y_J \\ -x_J \end{Bmatrix} = \frac{1}{2} \begin{bmatrix} (y_2 - y_4)(y_3 - y_1)(y_4 - y_2)(y_1 - y_3) \\ (x_4 - x_2)(x_1 - x_3)(x_2 - x_4)(x_3 - x_1) \end{bmatrix} \tag{3.1.26}$$

The mean stress approach used here gives the same result in two dimensions as the one-point quadrature rule for the quadrilateral since the Jacobian is at most bilinear.

3.1.2 Axisymmetric Case

The axisymmetric quadrilateral poses a special problem for the finite element method in that we must reduce a three-dimensional variational Equation (3.1.7) to a two-dimensional element domain. The formulation is complicated by the fact that the variational principal is cast in cylindrical, rather than Cartesian coordinates.

We will start by defining the cylindrical coordinate system as follows:

$$r^{\alpha} = (r, z, \theta) \tag{3.1.27}$$

While the above ordering of the coordinates is unconventional (and not right-handed), it degrades cleanly to the axisymmetric case. Note that Greek indices have a range of three and that superscripts and subscripts indicate contravariant and covariant tensor components, respectively.

The shape functions of the axisymmetric uniform strain quadrilateral are the same as those for the plane strain case (Table 3.1) and are defined implicitly in terms of the nodal coordinates

$$r_1 = r_{1I}\, \phi_I \tag{3.1.28}$$

Note that lower case English indices have a range of two and that, since the two-dimensional coordinate system is Cartesian, there is no distinction between covariant and contravariant tensor components.

In our Lagrangian formulation the same shape functions are applied to the displacement fields. This implies that the material derivatives of the shape functions vanish. As a result, these shape functions also apply to the velocity field, just as in the plane strain case:

$$\dot{r}_i = \dot{r}_{iI}\,\phi_I \tag{3.1.29}$$

The weak form given by Equation (3.1.7) is expressed in cylindrical coordinates as

$$\sum_e \int_{V_e} \left(T^{\alpha\beta}\Big|_{,\beta} + \rho b^\alpha - \rho u^\alpha \right) \delta u_\alpha dV = 0 \tag{3.1.30}$$

We are now faced with a three-dimensional variational principle, but only a two-dimensional element. Since the differential of volume imposes a factor of r on the differential of area ($dV = 2\pi r dA$), there is an implicit r weighting on the integrand of the weak form in Equation (3.1.30). This means that the integrand vanishes near the axis of symmetry (r = 0) regardless of the variations! This also means that the discretized equations generated by the finite element method become ill-conditioned near the axis.

This difficulty is resolved by dividing the integrand of Equation (3.1.30) by r to reduce the integration to the element domain. However, we must carry this weighting factor in order to apply Gauss' theorem in three dimensions. This technique was referred to as a Petrov-Galerkin, or area-weighted finite element, formulation by Goudreau and Hallquist [18].

$$\sum_e \left[\frac{1}{2\pi} \int_{V_e} T^{\alpha\beta}\Big|_\beta \left(\frac{1}{r}\,\delta u_\alpha \right) dV + \int_{A_e} \rho b^\alpha\, \delta u_\alpha dA - \int_{A_e} \rho u^\alpha \delta u_\alpha dA \right] = 0 \tag{3.1.31}$$

Integrating by parts and applying Gauss' theorem yields the following:

$$\sum_e \left[\int_{S_e} T^{a\beta} \delta u_a n_\beta ds - \int_{A_e} T^{a\beta} \left(\frac{1}{r} \delta u_a \right) \Big|_\beta r dA + \int_{A_e} \rho b^a \delta u_a dA - \int_{A_e} \rho \ddot{u}^a \delta u_a dA \right] = 0 \quad (3.1.32)$$

Evaluating the covariant derivative (see Fung [40]) in the preceding equation yields

$$\begin{aligned} \left(\frac{1}{r} \delta u_a \right) \Big|_\beta &= \left(\frac{1}{r} \delta u_a \right)_{,\beta} - \Gamma^\gamma_{a\beta} \left(\frac{1}{r} \delta u_\gamma \right) \\ &= \frac{1}{r} \delta u_{a,\beta} - \frac{1}{r} \Gamma^\gamma_{a\beta} \delta u_\gamma - \frac{1}{r^2} \delta_{1\beta} \delta u_a \end{aligned} \quad (3.1.33)$$

where $\Gamma^\gamma_{a\beta}$ are the Euclidian Christoffel symbols associated with the cylindrical coordinate system. The only nonzero components are

$$\begin{aligned} \Gamma^1_{33} &= -r \\ \Gamma^3_{13} &= \Gamma^3_{31} = \frac{1}{r} \end{aligned} \quad (3.1.34)$$

We are now in a position to degenerate the variational equations to the axisymmetric case. The axisymmetry conditions require that variations and derivatives in θ vanish. Combining Equations (3.1.32) to (3.1.34) and enforcing axisymmetry gives

$$\begin{aligned} \sum_e \Bigg[\int_{S_e} T_{1J} n_J \delta u_1 dS - \int_{A_e} \left(T_{1J} \delta u_{1,J} + r T^{33} \delta u_1 - \frac{1}{r} T_{11} \delta u_1 \right) dA \\ + \int_{A_e} \rho b_1 \delta u_1 dA - \int_{A_e} \rho u_1 \delta u_1 dA \Bigg] = 0 \end{aligned} \quad (3.1.35)$$

Note that we have dropped the contravariant superscript notation for English indices in going from Equations (3.1.32) to (3.1.35) because, as we stated previously, there is no distinction between contravariant and covariant components in our two-dimensional coordinate system.

A by-product of the Petrov-Galerkin formulation is that the resulting weak form for the axisymmetric case, Equation (3.1.35), is nearly identical to that of the plane strain case, Equation (3.1.8). The only difference is

the addition of the last two terms to the internal force expression, which is the second integral above. This is clearly a major architectural advantage to PRONTO.

Note that the last term of the axisymmetric internal force expression is not associated with strain. These forces are analogous to the covected force terms which appears in the stress divergence as shown below (see Fung [40]).

$$T^{\alpha\beta}\Big|_{\beta} = T^{\alpha\beta}{}_{,\beta} + \Gamma^{\alpha}_{\gamma\beta}\, T^{\gamma\beta} + \Gamma^{\beta}_{\gamma\beta}\, T^{\alpha\gamma}$$

$$= T^{\alpha\beta}{}_{,\beta} + \Gamma^{\alpha}_{\gamma\beta}\, T^{\gamma\beta} + \frac{1}{r}\, T^{\alpha 1} \qquad (3.1.36)$$

If the 1/r correction is omitted in Equation (3.1.31), the final term in the axisymmetric internal force disappears.

It is convenient for a finite element program to work with physical, rather that tensoral, stress components. In our formulation, the hoop stress is the only component which requires such a distinction. The physical hoop stress T_{33} is given by

$$T_{33} = r^2\, T^{33} \qquad (3.1.37)$$

The internal forces are then given by

$$f_{iI} = \int_A T_{ij}\phi_{I,j}\, dA + \int_A (T_{33}\delta_{i1} - T_{i1})\, \frac{1}{r}\, \phi_I\, dA \qquad (3.1.38)$$

Evaluating all these integrals with single point integration yields

$$f_{iI} = \bar{T}_{ij} B_{jI} + (T_{33}\delta_{i1} - T_{i1})\, \frac{A}{4\bar{r}}\, \Sigma_I \qquad (3.1.39)$$

where

$$\bar{r} = \frac{1}{4}\, \Sigma_I r_I \qquad (3.1.40)$$

We now see that the internal force vector for the axisymmetric case, Equation (3.1.39), is the same as that for the plane strain case, Equation (3.1.14), with the addition of the hoop stress and covected forces.

The velocity gradient in cylindrical coordinates is

$$\dot{u}_{\alpha}\big|_{\beta} = \dot{u}_{\alpha,\beta} - \Gamma^{\gamma}_{\alpha\beta}\,\dot{u}_{\gamma} \tag{3.1.41}$$

Substituting Equation (3.1.34) into the above equation, and enforcing axisymmetry leaves only five nonzero components; the four in-plane components, and the physical hoop strain rate D_{33}. This additional strain rate component is defined conjugate to Equation (3.1.37) as

$$D_{33} = \frac{1}{r^2}\,\dot{u}_3\big|_3 = \frac{\dot{u}_1}{r} \tag{3.1.42}$$

We evaluate this quantity with one point integration as follows:

$$\bar{D}_{33} = \frac{\dot{\bar{u}}_1}{\bar{r}} \tag{3.1.43}$$

where $\bar{r}$ is given by Equation (3.1.40) and

$$\dot{\bar{u}}_1 = \frac{1}{4}\,\Sigma_I\,\dot{u}_{1I} \tag{3.1.44}$$

3.1.3 Lumped Mass Matrix

One of the aforementioned advantages of using the Petrov-Galerkin method for the axisymmetric case is that the inertial terms in the variational statement of the boundary value problem are identical for both the plane strain, Equation (3.1.8), and axisymmetric, Equation (3.1.35), cases. Therefore, we can treat both cases at one time.

In order to reap the benefits of an explicit architecture, we must diagonalize the mass matrix. We do this by integrating the inertial energy variation as follows:

$$\int_A \rho u_i \delta u_i dA = u_{iI} m_{IJ} \delta u_{iJ} \tag{3.1.45}$$

where

$$m_{IJ} = \rho A \delta_{IJ} \tag{3.1.46}$$

and δ_{IJ} is the kroneker delta. Clearly the assembly process for the global mass matrix from the individual element matrices results in a global mass matrix which is diagonal and can be expressed as a vector, M_I.

3.2 Explicit Time Integration

PRONTO uses a modified central difference scheme to integrate the equations of motion through time. By this we mean that the velocities are integrated with a forward difference, while the displacements are integrated with a backward difference. The integration scheme for a node is expressed as

$$\mathbf{u}_t = \mathbf{f}_t^{EXT} - \mathbf{f}_t^{INT} \; / \; M \tag{3.2.1}$$

$$\dot{\mathbf{u}}_{t+\Delta t} = \dot{\mathbf{u}}_t + \Delta t \; \mathbf{u}_t \tag{3.2.2}$$

and

$$\mathbf{u}_{t+\Delta t} = \mathbf{u}_t + \Delta t \; \dot{\mathbf{u}}_{t+\Delta t} \tag{3.2.3}$$

where $\mathbf{f}_t^{EXT}$ and $\mathbf{f}_t^{INT}$ are the external and internal nodal forces, respectively, M is the nodal point lumped mass, and Δt is the time increment.

The central difference operator is conditionally stable. It can be shown that the Courant stability limit for the operator is given in terms of the highest eigenvalue in the system [41]:

$$\Delta t \leq \frac{2}{\omega_{max}} \tag{3.2.4}$$

In Section 3.5, we discuss how the highest eigenvalue is approximated and how we determine a stable time increment.

3.3 Finite Rotation Algorithm

We stated in Section 2.2 that one of our fundamental numerical challenges in the development of an accurate algorithm for finite rotations was the determination of **R**, the rotation tensor defined by the polar decomposition of the deformation gradient **F**. We developed an incremental algorithm for reasons of computational efficiency and numerical accuracy. The validity of the unrotated reference frame is based on the orthogonal transformation given by Equation (2.2.1). Therefore the crux of integrating Equation (2.1.6) for **R** is to maintain the orthogonality of **R**. If one integrates $\dot{\mathbf{R}} = \boldsymbol{\Omega}\mathbf{R}$ via a forward difference scheme, the orthogonality of **R** degenerates rapidly no matter how fine the time increments. We instead adapted the algorithm of Hughes and Winget [19] for integrating incremental rotations as follows.

A rigid body rotation over a time increment Δt may be represented by

$$\mathbf{x}_{t+\Delta t} = \mathbf{Q}_{\Delta t}\, \mathbf{x}_t \tag{3.3.1}$$

where $\mathbf{Q}_{\Delta t}$ is a proper orthogonal tensor with the same rate of rotation as **R** given by Equation (2.1.6). The total rotation **R** is updated via the highly accurate expression below.

$$\mathbf{R}_{t+\Delta t} = \mathbf{Q}_{\Delta t}\, \mathbf{R}_t \tag{3.3.2}$$

For a constant rate of rotation, the midpoint velocity and the midpoint coordinates are related by

$$\frac{1}{\Delta t}\,(\mathbf{x}_{t+\Delta t} - \mathbf{x}_t) = \frac{1}{2}\,\boldsymbol{\Omega}(\mathbf{x}_{t+\Delta t} + \mathbf{x}_t)\ . \tag{3.3.3}$$

Combining Equations (3.3.1) and (3.3.3) yields

$$(\mathbf{Q}_{\Delta t} - \mathbf{I})\,\mathbf{x}_t = \frac{\Delta t}{2}\,\boldsymbol{\Omega}(\mathbf{Q}_{\Delta t} + \mathbf{I})\,\mathbf{x}_t\ . \tag{3.3.4}$$

Since $\mathbf{x}_t$ is arbitrary in Equation (3.3.4), t may be eliminated. We then solve for $\mathbf{Q}_{\Delta t}$. The result is

$$\mathbf{Q}_{\Delta t} = \left(\mathbf{I} - \frac{\Delta t}{2}\,\boldsymbol{\Omega}\right)^{-1} \left(\mathbf{I} + \frac{\Delta t}{2}\,\boldsymbol{\Omega}\right) . \tag{3.3.5}$$

The accuracy of this integration scheme is dependent upon the accuracy of the midpoint relationship of Equation (3.3.3). The rate of rotation must not vary significantly over the time increment. Furthermore, Hughes and Winget [19] showed that the conditioning of Equation (3.3.5) degenerates as $\Delta t\boldsymbol{\Omega}$ grows.

Our complete numerical algorithm for a single time step is as follows:

1. Calculate $\mathbf{D}$ and $\mathbf{W}$.

2. Compute $z_i = e_{ijk}\, V_{jm}\, D_{mk}$,

$$\boldsymbol{\omega} = \mathbf{w} - 2[\mathbf{V} - \mathbf{I}\ \mathrm{tr}(\mathbf{V})]^{-1}\ \mathbf{z} \text{ , and}$$

$$\Omega_{ij} = \frac{1}{2}\, e_{ijk}\, \omega_k .$$

3. Solve $\left(\mathbf{I} - \frac{\Delta t}{2}\,\boldsymbol{\Omega}\right) \mathbf{R}_{t+\Delta t} = \left(\mathbf{I} + \frac{\Delta t}{2}\,\boldsymbol{\Omega}\right) \mathbf{R}_t$

4. Calculate $\dot{\mathbf{V}} = (\mathbf{D} + \mathbf{W})\ \mathbf{V} - \mathbf{V}\boldsymbol{\Omega}$.

5. Update $\mathbf{V}_{t+\Delta t} = \mathbf{V}_t + \Delta t\ \dot{\mathbf{V}}_{\Delta t}$.

6. Compute $\mathbf{d} = \mathbf{R}^{\mathsf{T}}\ \mathbf{D}\ \mathbf{R}$.

7. Integrate $\dot{\boldsymbol{\sigma}} = \mathbf{f}(\mathbf{d}, \boldsymbol{\sigma})$

8. Compute $\mathbf{T} = \mathbf{R}\ \boldsymbol{\sigma}\ \mathbf{R}^{\mathsf{T}}$.

This algorithm requires that the tensors $\mathbf{V}$ and $\mathbf{R}$ be stored in memory for each element.

3.4 Determination of Effective Moduli

Algorithms for calculating the stable time increment, hourglass control, bulk viscosity, and nonreflecting boundaries require dilitational and shear moduli. In PRONTO we use an algorithm for adaptively determining the effective dilatational and shear moduli of the material.

Since PRONTO uses an explicit integration algorithm, the constitutive response over a time step can be recast aposteriori as a hypoelastic relationship. We approximate this relationship as isotropic. This defines effective moduli, $\hat{\lambda}$ and $\hat{\mu}$ in terms of the hypoelastic stress increment and strain increment as follows:

$$\Delta\sigma_{ij} = \Delta t(\hat{\lambda} d_{kk}\ \delta_{ij} + 2\hat{\mu}\ d_{ij}) \tag{3.4.1}$$

Equation 3.4.1 can be rewritten in terms of volumetric and deviatoric parts as

$$\Delta\sigma_{kk} = \Delta t(3\hat{\lambda} + 2\hat{\mu})\ d_{kk} \tag{3.4.2}$$

and

$$s_{ij} = \Delta t\ 2\hat{\mu}\ e_{ij} \tag{3.4.3}$$

where

$$s_{ij} = \Delta\sigma_{ij} - \frac{1}{3}\ \Delta\sigma_{kk}\ \delta_{ij} \tag{3.4.4}$$

and

$$e_{ij} = d_{ij} - \frac{1}{3}\ d_{kk}\ \delta_{ij}\ . \tag{3.4.5}$$

The effective bulk modulus follows directly from Equation (3.4.2) as

$$3\hat{K} = 3\hat{\lambda} + 2\hat{\mu} = \frac{\Delta\sigma_{kk}}{\Delta t d_{mm}} \tag{3.4.6}$$

Taking the inner product of Equation (3.4.3) with the deviatoric strain rate and solving for the effective shear modulus $2\hat{\mu}$ gives

$$2\hat{\mu} = \frac{s_{ij}\, e_{ij}}{\Delta t\, e_{mn}\, e_{mn}} \tag{3.4.7}$$

Using the result of Equation (3.4.6) with Equation (3.4.7), we can calculate the effective dilatational modulus $\hat{\lambda} + 2\hat{\mu}$:

$$\hat{\lambda} + 2\hat{\mu} = \frac{1}{3}\left(3\hat{K} + 2\cdot(2\hat{\mu})\right) \tag{3.4.8}$$

If the strain increments are insignificant, Equations (3.4.6) and (3.4.7) will not yield numerically meaningful results. In this circumstance, PRONTO sets the dilatational modulus to an initial estimate, $\lambda_o + 2\mu_o$. An initial estimate of the dilatational modulus is, therefore, the only parameter which every constitutive model is required to provide to the time step control algorithm.

In a case where the volumetric strain increment is significant, but the deviatoric increment is not, the effective shear modulus can be estimated by rearranging Equation (3.4.8) as follows:

$$2\hat{\mu} = \frac{1}{2}\left(3(\lambda_o + 2\mu_o) - 3\hat{K}\right) \tag{3.4.9}$$

If neither strain increment is significant, PRONTO sets the effective shear modulus to the initial dilatational modulus.

The algorithm that PRONTO follows to estimate the effective dilatational and shear moduli is summarized in Table 3.2 Note that either of effective moduli calculated via this algorithm may be zero or negative. These degenerate cases must be taken into account whenever these moduli are used.

TABLE 3.2

$\Delta t d_{kk} > 10^{-6}$	$\Delta t^2 e_{ij} e_{ij} > 10^{-12}$	$\hat{\lambda} + 2\hat{\mu}$	$2\hat{\mu}$
Yes	Yes	(3.4.8)	(3.4.7)
Yes	No	$\lambda_o + 2\mu_o$	(3.4.9)
No	Yes	$\lambda_o + 2\mu_o$	(3.4.7)
No	No	$\lambda_o + 2\mu_o$	$\lambda_o + 2\mu_o$

3.5 Determination of the Stable Time Increment

Flanagan and Belytschko [20] provided eigenvalue estimates for the uniform strain quadrilateral described in Section 3.1. They showed that the maximum eigenvalue was bounded by

$$4 \frac{\lambda + 2\mu}{\rho} \frac{B_{1I} B_{1I}}{A^2} \geq \omega_{max}^2 \geq 2 \frac{\lambda + 2\mu}{\rho} \frac{B_{1I} B_{1I}}{A^2} \tag{3.5.1}$$

Using the effective dilatational modulus from Section 3.4 with the eigenvalue estimates of Equation (3.5.1) allows us to write the stability criteria of Equation (3.2.4) as

$$\Delta \hat{t}^2 \leq \frac{(\rho_o A_o)\, A}{(\lambda + 2\mu)\, B_{1I} B_{1I}} \,. \tag{3.5.2}$$

The stable time increment is determined from Equation (3.5.2) as the minimum over all elements.

Equation (3.5.2) is numerically invalid if the effective dilatational modulus is less than or equal to zero. A negative modulus indicates a strain softening situation (the Damage Model, Section 4.4, is the only currently supported constitutive model which allows strain softening), which

renders the central difference operator unconditionally unstable. In practice, however, strain softening is generally short lived, so that the calculations can continue in a stable manner once the softening energy has been dissipated. To aid the user in controlling an unstable strain softening situation, we adjust the effective dilatational modulus with the strain softening scale factor (Appendix A, command 9) as follows:

$$\text{If } \hat{\lambda} + 2\hat{\mu} < 0 \quad ; \quad \lambda + 2\mu = \frac{\lambda_o + 2\mu_o}{(\text{ssft})^2} \tag{3.5.3}$$

To avoid dividing by zero in Equation (3.5.2), we then enforce the following condition:

$$\lambda + 2\mu \geq (\lambda_o + 2\mu_o)\ 10^{-6} \tag{3.5.4}$$

The estimate of the critical time increment given in the preceding equation is for the case where there is no damping present in the system. If we define ϵ as the fraction of critical damping in the highest element mode, the stability criteria of Equation (3.5.2) becomes

$$\Delta t \leq \Delta\hat{t} \left(\sqrt{1 + \epsilon^2} - \epsilon \right) \tag{3.5.5}$$

Conventional estimates of the critical time increment size have been based on the transit time of a dilatational wave over the shortest dimension of an element or zone. For the undamped case this gives

$$\Delta t = \ell / c \tag{3.5.6}$$

where c is the dilitational wave speed.

There are two fundamental and important differences between the time increment limits given by Equations (3.5.2) and (3.5.6). First, our time increment limit is dependent on a characteristic element dimension, which is based on the finite element gradient operator and does not require an ad hoc guess of this dimension. This characteristic element dimension, ℓ, is defined by inspection of Equation (3.5.2) as

$$\ell = A / \sqrt{B_{1I}\ B_{1I}} \tag{3.5.7}$$

Second, the sound speed used in the estimate is based on the current response of the material and not on the original elastic sound speed. For materials which experience a reduction in stiffness due to plastic flow, this can result in significant increases in the critical time increment.

It should be noted that the stability analysis performed at each time step predicts the critical time increment for the next step. Our assumption is that the conservativeness of this estimate compensates for any reduction in the stable time increment over a single time step.

3.6 Hourglass Control Algorithm

The mean stress-strain formulation of the uniform strain element considers only a fully linear velocity field. The remaining portion of the nodal velocity field is the so-called hourglass field. Excitation of these modes may lead to severe, unresisted mesh distortion. The hourglass control algorithm described here is taken directly from Flanagan and Belytschko [17]. The method isolates the hourglass modes so that they may be treated independently of the rigid body and uniform strain modes.

A fully linear velocity field for the quadrilateral can be described by

$$\dot{u}_i^{LIN} = \dot{\bar{u}}_i + \dot{\bar{u}}_{i,J} \, (x_J - \bar{x}_J) \tag{3.6.1}$$

The mean coordinates $\bar{x}_i$ correspond to the center of the element and are defined as

$$\bar{x}_i = \frac{1}{4} \, x_{iI} \, \Sigma_I \tag{3.6.2}$$

The mean translational velocity is similarly defined by

$$\dot{\bar{u}}_i = \frac{1}{4} \, \dot{u}_{iI} \, \Sigma_I \tag{3.6.3}$$

The linear portion of the nodal velocity field may be expressed by specializing Equation (3.6.1) to the nodes as follows:

$$\dot{u}_{iI}^{LIN} = \dot{\bar{u}}_i \, \Sigma_I + \dot{\bar{u}}_{i,j}(x_{jI} - \bar{x}_j \, \Sigma_I) \tag{3.6.4}$$

where Σ_I is used to maintain consistent index notation and indicates that $\dot{\bar{u}}_i$ and $\bar{x}_j$ are independent of position within the element. From Equations (3.1.16) and (3.6.4), and the orthogonality of the base vectors, it follows that

$$\dot{u}_{iI} \, \Sigma_I = \dot{u}_{iI}^{LIN} \, \Sigma_I = 4\dot{\bar{u}}_i \tag{3.6.5}$$

and

$$\dot{u}_{iI} \, B_{jI} = \dot{u}_{iI}^{LIN} \, B_{jI} = A\dot{\bar{u}}_{i,j} \tag{3.6.6}$$

The hourglass field $\dot{u}_{iI}^{HG}$ may now be defined by removing the linear portion of the nodal velocity field:

$$\dot{u}_{iI}^{HG} = \dot{u}_{iI} - \dot{u}_{iI}^{LIN} \tag{3.6.7}$$

Equations (3.6.5) through (3.6.7) prove that Σ_I and B_{jI} are orthogonal to the hourglass field:

$$\dot{u}_{iI}^{HG} \, \Sigma_I = 0 \tag{3.6.8}$$

$$\dot{u}_{iI}^{HG} \, B_{jI} = 0 \tag{3.6.9}$$

Furthermore, it can be shown that the B matrix is a linear combination of the volumetric base vectors, Λ_I, so Equation (3.6.9) can be written as

$$\dot{u}_{iI}^{HG} \, \Lambda_I = 0 \tag{3.6.10}$$

Equations (3.6.8) and (3.6.10) show that the hourglass field is orthogonal to all the base vectors in Table 3.1 except the hourglass base vectors. Therefore, $\dot{u}_{iI}^{HG}$ may be expanded as a linear combination of the hourglass base vectors as follows:

$$\dot{u}_{iI}^{HG} = \frac{1}{2} \dot{q}_i \Gamma_I \tag{3.6.11}$$

The hourglass nodal velocities are represented by $\dot{q}_i$ above (the leading constant is added to normalize Γ_I). We now define the hourglass shape vector γ_I such that

$$\dot{q}_i = \frac{1}{2} \dot{u}_{iI} \gamma_I \tag{3.6.12}$$

By substituting Equations (3.6.4), (3.6.7), and (3.6.12) into (3.6.11), then multiplying by Γ_I and using the orthogonality of the base vectors, we obtain the following:

$$\dot{\bar{u}}_{iI} \Gamma_I - \dot{\bar{u}}_{i,J} x_{jI} \Gamma_I = \dot{u}_{iI} \gamma_I \tag{3.6.13}$$

With the definition of the mean velocity gradient, Equation (3.1.13), we can eliminate the nodal velocities above. As a result, we can compute γ_I from the following expression:

$$\gamma_I = \Gamma_I - \frac{1}{A} B_{iI} x_{iJ} \Gamma_J \tag{3.6.14}$$

The difference between the hourglass base vectors Γ_I and the hourglass shape vectors γ_I is very important. They are identical if and only if the quadrilateral is a parallelogram. For a general shape, Γ_I is orthogonal to B_{jI} while γ_I is orthogonal to the linear velocity field $\dot{u}_{iI}^{LIN}$. While Γ_I defines the hourglass pattern, γ_I is necessary to accurately detect hourglassing. Equation (3.6.14) is simple enough for the quadrilateral that it can be written explicitly as

$$\gamma_I = \frac{1}{A} \begin{bmatrix} x_2(y_3 - y_4) + x_3(y_4 - y_2) + x_4(y_2 - y_3) \\ x_3(y_1 - y_4) + x_4(y_3 - y_1) + x_1(y_4 - y_3) \\ x_4(y_1 - y_2) + x_1(y_2 - y_4) + x_2(y_4 - y_1) \\ x_1(y_3 - y_2) + x_2(y_1 - y_3) + x_3(y_2 - y_1) \end{bmatrix} \tag{3.6.15}$$

For the purpose of controlling the hourglass modes, we define generalized forces Q_i, which are conjugate to $\dot{q}_i$ so that the rate of work is

$$\dot{u}_{iI} \, f_{iI}^{HG} = \frac{1}{2} \, Q_i \, \dot{q}_i \tag{3.6.16}$$

for arbitrary $\dot{u}_{iI}$. Using Equation (3.6.12), it follows that the contribution of the hourglass resistance to the nodal forces is given by

$$f_{iI}^{HG} = \frac{1}{2} \, Q_i \, \gamma_I \tag{3.6.17}$$

Two types of hourglass resistance are used in PRONTO: artificial stiffness and artificial damping. We express this combination as

$$Q_i = Q_i^K + Q_i^V \tag{3.6.18}$$

In terms of the tuneable stiffness (κ) and viscosity (ϵ) factors, these restinances are given by

$$\dot{Q}_i^K = \frac{\kappa}{2} \, 2\hat{\mu} \, \frac{B_{iI} \, B_{iI}}{A} \, \dot{q}_i \tag{3.6.19}$$

$$Q_i^V = \epsilon \, \sqrt{\min(0, 2\hat{\mu}) \, m} \, \dot{q}_i \tag{3.6.20}$$

Note that the stiffness expression must be integrated, which further requires that this resistance be stored in a global array.

Observe that the nodal antihourglass forces of Equation (3.6.17) have the shape of γ_I rather than Γ_I. This fact is essential since the antihourglass forces should be orthogonal to the linear velocity field, so that no energy is transferred to or from the rigid body and uniform strain modes by the antihourglassing scheme.

We would prefer to use only hourglass stiffness and, in fact, this is what is used for the plane strain case (κ = .05 and ϵ = 0.0). Unfortunately, the nonstrain terms in the Petrov-Galerkin formulation give rise to an instability which is best stabilized using hourglass viscosity. For the axisymmetric case, values of κ = .01 and ϵ = .03 are used.

3.7 Artificial Bulk Viscosity

Artificial viscosity is applied to the numerical solution for two reasons. First is to prevent high velocity gradients from collapsing an element before it has a chance to respond. The second reason is to quiet truncation frequency "ringing".

Ideally, one would like to add viscosity only to the highest mode of the element, but isolating this mode is impractical. The standard technique is to simply add viscosity to the volumetric or "bulk" response. This generates a viscous pressure in terms of the volume strain rate as follows:

$$q = b_1 \rho c \ell \frac{\dot{V}}{V} - \rho \left(b_2 \ell \frac{\dot{V}}{V} \right)^2 \qquad (3.7.1)$$

The quadratic term in Equation (3.7.1) is more important and is designed to "smear" a shock front across several elements. This term yields a jump in energy as a smeared shock passes, which simulates the shock heating. As a result, the smeared shock front can be propagated as a steady wave.

The linear term is intended to dissipate truncation frequency oscillations. Note that the quadratic term is only applied to compressive strain rates since an element cannot collapse in expansion.

The preceding expression is simplified if we use the undamped stable time increment defined by Equation (3.5.2) and write

$$\Delta\hat{t} = \frac{\ell}{c} = \sqrt{\frac{A^2}{B_{iI}B_{iI}} \cdot \frac{\rho}{\lambda + 2\mu}} \tag{3.7.2}$$

or

$$\Delta\hat{t} = \sqrt{\frac{m}{\lambda + 2\mu} \cdot \frac{A}{B_{iI}B_{iI}}} \tag{3.7.3}$$

We now define the factor ϵ such that the quadratic viscosity term vanishes in expansion

$$\epsilon = b_1 - b_2^2 \, \Delta\hat{t} \, \min(0, D_{kk}) \tag{3.7.4}$$

This quantity is required for the damped stability criteria of Equation (3.5.5). Note that the condition imposed by Equation (3.5.4) prevents Equation (3.7.4) from yielding so large a value of ϵ that Equation (3.5.5) would numerically yield a zero value.

We will show below that ϵ is an estimate of the fraction of critical damping in the highest element mode. Using Equation (3.7.4) in Equation (3.7.3) allows us to write the viscous pressure as

$$q = (b_1 - b_2^2 \, \Delta\hat{t} \, D_{kk})(\lambda + 2\mu) \, \Delta\hat{t} \, D_{kk} \tag{3.7.5}$$

The bulk viscosity pressure is appended to the stresses during the internal force calculations to yield the following forces:

$$f_{iI} = q \, B_{iI} \tag{3.7.6}$$

The above expression can be expanded using Equations (3.7.3) and (3.7.4) to yield

$$f_{iI} = \epsilon\rho c \ell \, \frac{1}{A} \, B_{jJ} \, B_{iI} \quad u_{jJ} \tag{3.7.7}$$

This form indicates that if B_{1I} is an eigenvector, the modal damping is

$$\epsilon \rho \frac{cA}{\ell} \tag{3.7.8}$$

The critical damping estimate of the maximum element frequency is

$$2m\omega = 2 \frac{\rho A}{4} \frac{2c}{\ell} = \rho \frac{Ac}{\ell} \tag{3.7.9}$$

The two expressions above show that ϵ is indeed a good estimate of the fraction of critical damping in the highest mode.

3.8 Adaptive Element Deletion

The adaptive element deletion option was added to PRONTO 2D to provide the capability to model catastrophic material failure. This option should not be confused with the element block deletion option (Appendix B, command 35) which can be used to remove an entire block of material from the analysis at some predetermined time. The keyword here is "adaptive". We allow the user to specify criteria which define when the material fails within an element. This criteria is defined at the element level and PRONTO checks every time step to determine whether material failure has occurred.

Currently, the user can define failure in terms of energy per unit volume, Von Mises stress, pressure, or maximum principal stress. Also, failure criteria can be defined in terms of any internal state variable. Note that the pressure is positive in compression, $p = -tr(\sigma)$. The adaptive element death capability requires a very mature user who understands how his/her material behaves. The capability built into the code is quite general, and it is possible for the user to define a nonsensical failure criteria. We allow the user to specify the failure in terms of a particular variable, a prescribed value of the variable at failure, and what we refer to as the mode of failure. By mode we mean minimum, maximum, or absolute value.

The adaptive element deletion capability is completely vectorized and does not add any appreciable computational penalty. We define a status array (length=NUMEL) which has a value of one or zero. If an element is "alive", the status array contains a value of one for that element. When PRONTO detects that the element has "died", the value of the status array for that element is reduced to zero over five time steps. We use the status array to wipe out any contribution that a deleted element makes in step 1 of Section 8.2. Each deleted element undergoes all the calculations which it would if it were not deleted, but its contributions are not included in the timestep control algorithm nor the stress divergence. This is accomplished by a few multiplications of critical results by the status array. If the element is not deleted, the results are multiplied by a one and the results are unchanged. If the element is deleted, the results are multiplied by a zero and the results are neutralized. Hence, the overall cost of this algorithm is a few multiplications per element.

When the element is deleted, its contribution to the nodal point lumped mass is still retained. When all the elements connected to a particular node are deleted, the node then becomes a free nodal mass, whose motion we continue to calculate.

It is more convenient for post-processing to define the status array exactly opposite to our convention. For this reason, we flip each value as we write the status array to the post-processing data base. Note that if the adaptive element deletion and/or the element block deletion options are used, the element status array is automatically written to the data base.

4.0 CONSTITUTIVE MODELS

One of the primary reasons for developing PRONTO was to have a numerical testbed for developing constitutive models. As a result, considerable effort was directed to write a flexible material interface subroutine which allows a constitutive model to be added to the code with minimal effort. The MATINT subroutine in PRONTO allows a constitutive modeler to add a new material model to the program by filling in a handful of numbers in data statements which tell the program how to set up the internal data base for the new model. Consequently, the constitutive modeler does not have to understand the inner workings of PRONTO and does not have to write any format statements or juggle the memory allocation in the code. The comments in the FORTRAN explain in great detail how to add the new model. See Appendix C for the steps to be taken to add a new constitutive model.

Currently there are nine material models in the code. Since models can be added with such ease, this number is expected to increase as applications requiring new materials arise.

All material models are written in terms of the unrotated Cauchy stress, $\boldsymbol{\sigma}$, and the deformation rate in the unrotated configuration, $\mathbf{d}$.

For each of the materials described below, we give a list of the internal state variables used in that particular material model. We also give a list of the material constants which are stored in the PROP array (Appendix B, Section 5.0). In the list of material properties, the items denoted by an "*" are material properties which are calculated internally. The remaining material properties are the actual values read from the input data.

The relationship between the material models described in this chapter and the equations of state described in Chapter 5 must be understood in order to properly use the equations of state. We have structured PRONTO so that material models can act as a host to an equation of state. Not all of the constitutive models described below in this chapter do so. The equations of state cannot be used except in conjunction with a material model.

The equation of state can only give the volumetric material response. The hydrodynamic material model (Section 4.7) has only volumetric response and just calls the specified equation of state. The elastic plastic hydrodynamic material model (Section 4.9) uses classical J_2 plasticity theory to determine the deviatoric material response and calls the specified equation of state for the volumetric material response. Any of the other constitutive models in this chapter could be restructured to use an equation of state for the volumetric material response if required.

We have followed the historical convention used for each material model for the sign of a positive pressure. We inherited most of these constitutive models from previous codes and did not wish to change what has come to be accepted conventions for a positive pressure. This means that for some models the pressure is positive in tension and for some it is positive in compression. Table 4.1 shows the convention used for each of the material models described in this chapter.

TABLE 4.1

Model	Tension	Compression
Elastic	positive	negative
Elastic Plastic	positive	negative
Viscoplastic	positive	negative
Soils and Crushable Foams	negative	positive
Low Density Foams	positive	negative
Hydrodynamic	negative	positive
Rate and Temperature Dependent Plasticity	positive	negative
Elastic Plastic Hydrodynamic	negative	positive

4.1 Elastic Material, Hooke's Law

A linear elastic material is defined using Hooke's Law. In a rate form, this is written as

$$\dot{\boldsymbol{\sigma}} = \lambda(\text{tr } \mathbf{d})\boldsymbol{\delta} + 2\mu\, \mathbf{d} \tag{4.1.1}$$

where λ and μ are the elastic Lame material constants.

This model has no internal state variables.

The PROP array for this material contains the following entries:

PROP(1) – Young's Modulus, E
PROP(2) – Poisson's Ratio, ν
*PROP(3) – λ
*PROP(4) – 2μ

4.2 Elastic Plastic Material with Combined Hardening

The elastic plastic model is based on a standard Von Mises type yield condition and uses combined kinematic and isotropic hardening. This model is widely used in many finite element and finite difference computer programs and the many details of its derivation are scattered throughout the literature. Here, we present the model in detail because we feel that many users of the model are not familiar with its underlying assumptions and numerical approximations.

4.2.1 Basic Definitions and Assumptions

Some definitions and assumptions are outlined here. Referring to Figure 4.2.1, which shows the yield surface in deviatoric stress space, we define the backstress (the center of the yield surface) by the tensor, $\boldsymbol{\alpha}$.

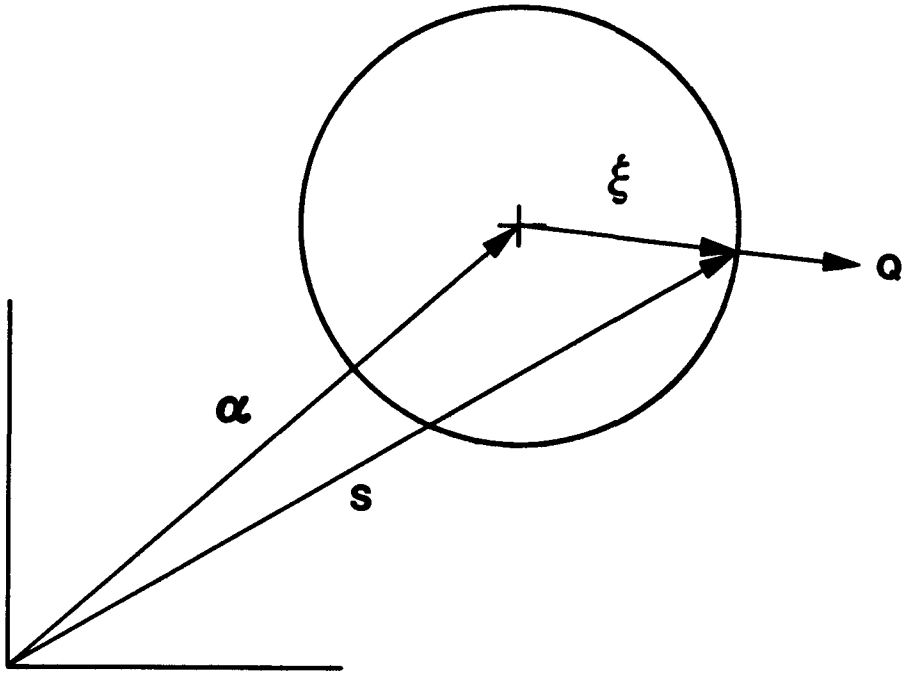


Figure 4.2.1. Yield Surface in Deviatoric Stress Space

If $\boldsymbol{\sigma}$ is the current value of the stress, we define the deviatoric part of the current stress by

$$\mathbf{S} = \boldsymbol{\sigma} - \frac{1}{3}\,\mathrm{tr}\boldsymbol{\sigma}\,\boldsymbol{\delta} \ . \tag{4.2.1}$$

We define the stress difference measured by subtracting the backstress from the deviatoric stress by

$$\boldsymbol{\xi} = \mathbf{S} - \boldsymbol{\alpha} \tag{4.2.2}$$

The magnitude of the deviatoric stress, R, is defined by

$$R = |\boldsymbol{\xi}| = \sqrt{\boldsymbol{\xi}:\boldsymbol{\xi}} \ , \tag{4.2.3}$$

where we denote the inner product of second order tensors by $\mathbf{S}:\mathbf{S} = S_{ij}\,S_{ij}$. Note that if the backstress is zero (isotropic hardening case) the stress difference is equal to the deviatoric part of the current stress, $\mathbf{S}$.

The Von Mises yield surface is defined as

$$f(\boldsymbol{\sigma}) = \frac{1}{2}\,\boldsymbol{\xi}:\boldsymbol{\xi} = \kappa^2 \ . \tag{4.2.4}$$

The Von Mises effective stress, $\bar{\sigma}$, is defined by

$$\bar{\sigma} = \sqrt{\frac{3}{2}\,\boldsymbol{\xi}:\boldsymbol{\xi}} \ . \tag{4.2.5}$$

Since R is the magnitude of the deviatoric stress tensor when $\boldsymbol{\alpha} = 0$, it follows that

$$R = \sqrt{2}\,\kappa = \sqrt{\frac{2}{3}}\,\bar{\sigma} \ . \tag{4.2.6}$$

The normal to the yield surface can be determined from Equation (4.2.4)

$$\mathbf{Q} = \frac{\partial f}{\partial \boldsymbol{\sigma}} \Big/ \left| \frac{\partial f}{\partial \boldsymbol{\sigma}} \right| = \boldsymbol{\xi}/R \ . \tag{4.2.7}$$

We assume that the strain rate can be decomposed into elastic and plastic parts by an additive decomposition

$$\mathbf{d} = \mathbf{d}^{el} + \mathbf{d}^{pl} \tag{4.2.8}$$

and assume that the plastic part of the strain rate is given by a normality condition

$$\mathbf{d}^{pl} = \gamma \mathbf{Q} \ . \tag{4.2.9}$$

when the scalar multiplier, γ, must be determined.

A scalar measure of equivalent plastic strain rate is defined by

$$\bar{d}^{pl} = \sqrt{\frac{2}{3} \mathbf{d}^{pl} : \mathbf{d}^{pl}} \tag{4.2.10}$$

which is chosen such that

$$\bar{\sigma}\, \bar{d}^{pl} = \boldsymbol{\sigma} : \mathbf{d}^{pl} \ . \tag{4.2.11}$$

The stress rate is assumed to be purely due to the elastic part of the strain rate and is expressed in terms of Hooke's law by

$$\dot{\boldsymbol{\sigma}} = \lambda \ \mathrm{tr}\ \mathbf{d}^{el}\ \boldsymbol{\delta} + 2\mu\ \mathbf{d}^{el} \ . \tag{4.2.12}$$

where λ and μ are the Lame constants for the material.

Below, we develop the theory for the cases of isotropic hardening, kinematic hardening and combined hardening separately so that the reader can see the details of each case.

4.2.2 Isotropic Hardening

In the isotropic hardening case, the backstress is zero and the stress difference is equal to the deviatoric stress, S. We write a consistency condition by taking the rate of Equation (4.2.4)

$$\dot{f}(\boldsymbol{\sigma}) = 2\,\kappa\,\dot{\kappa}\ . \tag{4.2.13}$$

By "consistency" we mean that the state of stress must remain on the yield surface at all times. We use the chain rule and the definition of the normal to the yield surface given by Equation (4.2.7) to obtain

$$\dot{f}(\boldsymbol{\sigma}) = \frac{\partial f}{\partial \boldsymbol{\sigma}} : \dot{\boldsymbol{\sigma}} = \left| \frac{\partial f}{\partial \boldsymbol{\sigma}} \right| \mathbf{Q} : \dot{\boldsymbol{\sigma}} \tag{4.2.14}$$

and from Equations (4.2.3) and (4.2.4)

$$\left| \frac{\partial f}{\partial \boldsymbol{\sigma}} \right| = |\mathbf{S}| = R\ . \tag{4.2.15}$$

Combining Equations (4.2.13), (4.2.14), and (4.2.15)

$$\frac{1}{R}\,\mathbf{S} : \dot{\boldsymbol{\sigma}} = \dot{R}\ . \tag{4.2.16}$$

We note that because $\mathbf{S}$ is deviatoric, $\mathbf{S}:\dot{\boldsymbol{\sigma}} = \mathbf{S}:\dot{\mathbf{S}}$ and

$$\mathbf{S}:\dot{\mathbf{S}} = \frac{d}{dt}\left(\frac{1}{2}\,\mathbf{S}:\mathbf{S}\right) = \frac{d}{dt}\left(\frac{\bar{\sigma}^2}{3}\right) = \frac{2}{3}\,\bar{\sigma}\,\dot{\bar{\sigma}}\ . \tag{4.2.17}$$

Then Equation (4.2.16) can be written as

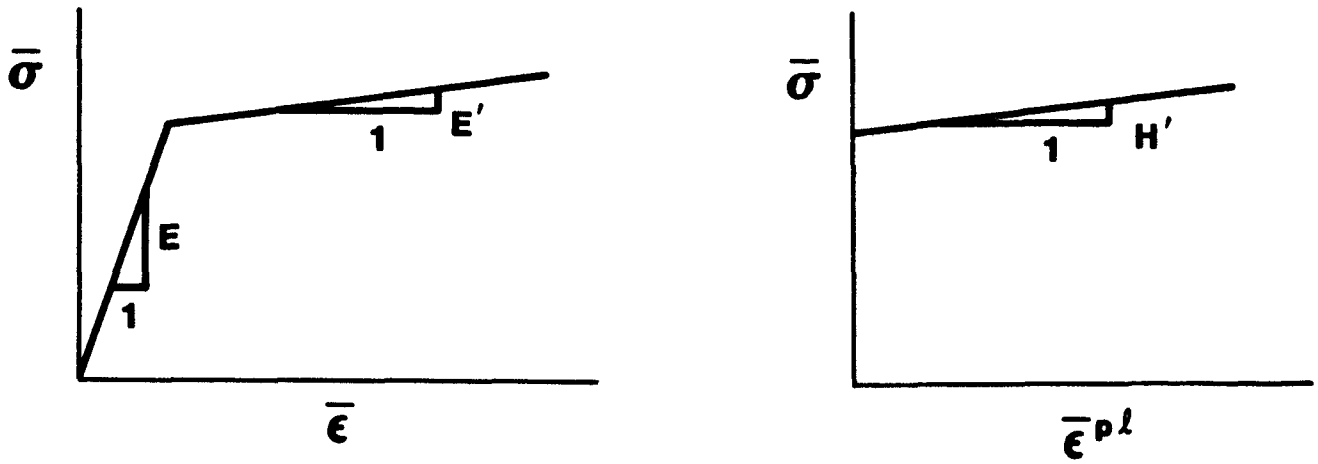
$$\dot{R} = \sqrt{\frac{2}{3}}\,\dot{\bar{\sigma}} = \sqrt{\frac{2}{3}}\ H'\ \bar{d}^{pl} \tag{4.2.18}$$

where H' is the slope of the yield stress versus equivalent plastic strain ($\bar{\sigma}$ versus $\bar{\epsilon}^{pl}$). This is derivable from the data from a uniaxial tension test as shown in Figure 4.2.2.

The consistency condition, Equation (4.2.16) and Equation (4.2.18), result in

$$\sqrt{\frac{2}{3}}\ H'\ \bar{d}^{pl} = \mathbf{Q}:\dot{\boldsymbol{\sigma}}\ . \tag{4.2.19}$$

We define the trial elastic stress rate $\dot{\boldsymbol{\sigma}}^{TR}$ by



$$H' = \frac{EE'}{E - E'}$$

Figure 4.2.2. Conversion of Data From a Uniaxial Tension Test to Equivalent Plastic Strain Versus Von Mises Stress

$$\dot{\boldsymbol{\sigma}}^{TR} = \mathbf{C}:\mathbf{d} \tag{4.2.20}$$

where $\mathbf{C}$ is the fourth order tensor of elastic coefficients defined by Equation (4.2.12). Combining the strain rate decomposition defined in Equation (4.2.8) with Equations (4.2.19) and (4.2.20) yields

$$\sqrt{\frac{2}{3}}\, H'\, \bar{d}^{pl} = \mathbf{Q}:\dot{\boldsymbol{\sigma}}^{TR} - \mathbf{Q}:\mathbf{C}:\mathbf{d}^{pl} \ . \tag{4.2.21}$$

We note that because $\mathbf{Q}$ is deviatoric, $\mathbf{C}:\mathbf{Q} = 2\mu\,\mathbf{Q}$ and $\mathbf{Q}:\mathbf{C}:\mathbf{Q} = 2\mu$. Then using the normality condition, Equation (4.2.9), the definition of equivalent plastic strain, Equation (4.2.10), and Equation (4.2.21)

$$\frac{2}{3}\, H'\, \gamma = \mathbf{Q}:\dot{\boldsymbol{\sigma}}^{TR} - \gamma\, 2\mu \tag{4.2.22}$$

and since $\mathbf{Q}$ is deviatoric ($\mathbf{Q}:\dot{\boldsymbol{\sigma}}^{TR} = 2\mu\,\mathbf{Q}:\mathbf{d}$) we can determine γ from Equation (4.2.22) as

$$\gamma = \frac{1}{(1 + H'/3\mu)}\, \mathbf{Q}:\mathbf{d} \ . \tag{4.2.23}$$

The current normal to the yield surface, $\mathbf{Q}$, and the total strain rate, $\mathbf{d}$, are known quantities. Hence, from Equation (4.2.23), γ can be determined which can be used in Equation (4.2.9) to determine the plastic part of the strain rate which, with the additive strain rate decomposition and the elastic stress rate of Equations (4.2.8) and (4.2.12), completes the definition of the rate equations.

We still must explain how to integrate the rate equations subject to the constraint that the stress must remain on the yield surface. We will show how that is accomplished in Section 4.2.5.

4.2.3 Kinematic Hardening

Just as before with the isotropic hardening case, we write a Von Mises yield condition but now in terms of the stress difference

$$f(\xi) = \frac{1}{2}\,\xi:\xi = \kappa^2 \ . \tag{4.2.24}$$

It is important to remember that α and ξ are deviatoric tensors. The consistency condition is written for kinematic hardening as

$$\dot{f}(\xi) = 0 \tag{4.2.25}$$

because the size of the yield surface does not grow with kinematic hardening, therefore, $\dot{\kappa} = 0$. Using the chain rule on Equation (4.2.25)

$$\frac{\partial f}{\partial \xi} :\dot{\xi} = 0 \tag{4.2.26}$$

and

$$\frac{\partial f}{\partial \xi} = \left| \frac{\partial f}{\partial \xi} \right| Q = R\,Q \ . \tag{4.2.27}$$

Combining Equations (4.2.26) and (4.2.27) and assuming that $R \neq 0$

$$Q:\dot{\xi} = 0 \tag{4.2.28}$$

or

$$Q:(\dot{S} - \dot{\alpha}) = 0 \ . \tag{4.2.29}$$

A geometric interpretation of Equation (4.2.29) is shown in Figure 4.2.3 where it can be seen that the backstress moves in a direction parallel to the normal to the yield surface.

We must now decide how $\dot{\alpha}$ is defined. Recall that for the isotropic hardening case, Equation (4.2.19)

$$Q:\dot{\sigma} = \sqrt{\frac{2}{3}}\,H'\,\bar{d}^{pl} = \frac{2}{3}\,H'\,\gamma \ . \tag{4.2.30}$$

The kinematic hardening condition assumes that

$$\dot{\alpha} = \phi\,d^{pl} = \phi\gamma\,Q \tag{4.2.31}$$

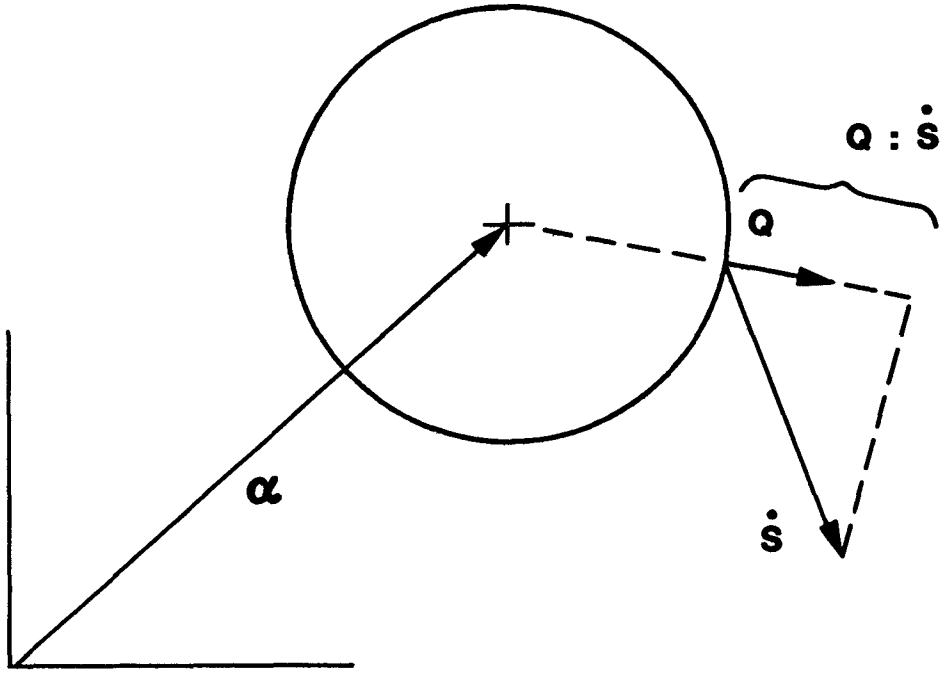


Figure 4.2.3. Geometric Interpretation of the Consistency Condition for Kinematic Hardening

where ϕ is a material parameter. Equation (4.2.31) combined with Equation (4.2.29) gives a result identical to the isotropic hardening case, Equation (4.2.30), if ϕ is chosen to be $\frac{2}{3}$ H'. Hence, either Equation (4.2.30) or (4.2.31) gives us a scalar condition on $\dot{\boldsymbol{\alpha}}$. Note that both of these are assumptions and must be shown to be reasonable. Of course, experience with material models based on these assumptions has proven them to be reasonable representations of material behavior.

Using Equation (4.2.30), the strain rate decomposition, Equation (4.2.8), and the elastic strain rate, Equation (4.2.12), in the consistency condition for kinematic hardening, Equation (4.2.29) gives

$$\frac{2}{3} H' \gamma \mathbf{Q} = \dot{\boldsymbol{\sigma}}^{TR} - \mathbf{C}:\mathbf{d}^{pl} \quad . \tag{4.2.32}$$

Taking the tensor inner product of both sides of Equation (4.2.32) with $\mathbf{Q}$ gives

$$\mathbf{Q}: \frac{2}{3} H' \gamma \mathbf{Q} = \mathbf{Q}:(\dot{\boldsymbol{\sigma}}^{TR} - 2\mu\gamma \mathbf{Q}) \tag{4.2.33}$$

Again, because $\mathbf{Q}$ is deviatoric; $\mathbf{C}:\mathbf{Q} = 2\mu \mathbf{Q}$ and $\mathbf{Q}:\mathbf{C}:\mathbf{Q} = 2\mu$.

Solving Equation (4.2.33) for γ gives

$$\gamma = \frac{1}{(1 + H'/3\mu)} \mathbf{Q}:\mathbf{d} \tag{4.2.34}$$

which is the same result as was obtained for the isotropic hardening case.

4.2.4 Combined Isotropic and Kinematic Hardening

For the combined hardening case we define a scalar parameter, β, which determines the amount of each type of hardening. We require that

$$0 \leq \beta \leq 1 \quad . \tag{4.2.35}$$

Figure 4.2.4 illustrates the uniaxial response which will be computed for $\bar{\sigma}$ for different choices of β. When $\beta = 0$ we have only kinematic hardening and when $\beta = 1$ we have only isotropic hardening.

We use the results derived before for the independent hardening cases and multiply by the appropriate fraction for each type of hardening. Equations (4.2.18) and (4.2.31) are rewritten as

$$\dot{R} = \sqrt{\frac{2}{3}}\, H'\, \bar{d}^{pl}\, \beta \tag{4.2.36}$$

and

$$\dot{\alpha} = \frac{2}{3}\, H'\, \mathbf{d}^{pl}\, (1 - \beta) = \frac{2}{3}\, H'\, \gamma \mathbf{Q} (1 - \beta)\ . \tag{4.2.37}$$

As before, we write a consistency condition

$$\mathbf{Q}:\dot{\boldsymbol{\xi}} = \dot{R} \tag{4.2.38}$$

or

$$\mathbf{Q}:(\dot{\mathbf{S}} - \dot{\boldsymbol{\alpha}}) = \sqrt{\frac{2}{3}}\, H'\, \bar{d}^{pl}\, \beta\ . \tag{4.2.39}$$

Using the elastic stress rate and the additive strain rate decomposition with Equation (4.2.39) and taking the tensor product with the normal, $\mathbf{Q}$

$$\mathbf{Q}:\dot{\boldsymbol{\sigma}}^{TR} - \gamma\, \mathbf{Q}:\mathbf{C}:\mathbf{Q} - \mathbf{Q}:\left[\frac{2}{3}\, H'\, \gamma(1 - \beta)\right]:\mathbf{Q} = \mathbf{Q}:\left[\sqrt{\frac{2}{3}}\, H' \sqrt{\frac{2}{3}}\, \beta\gamma\right]:\mathbf{Q}\ . \tag{4.2.40}$$

Solving for γ

$$\gamma = \frac{1}{(1 + H'/3\mu)}\, \mathbf{Q}:\mathbf{d} \tag{4.2.41}$$

which is the same result as we obtained for each of the independent cases.

We summarize the governing equations for the combined theory:

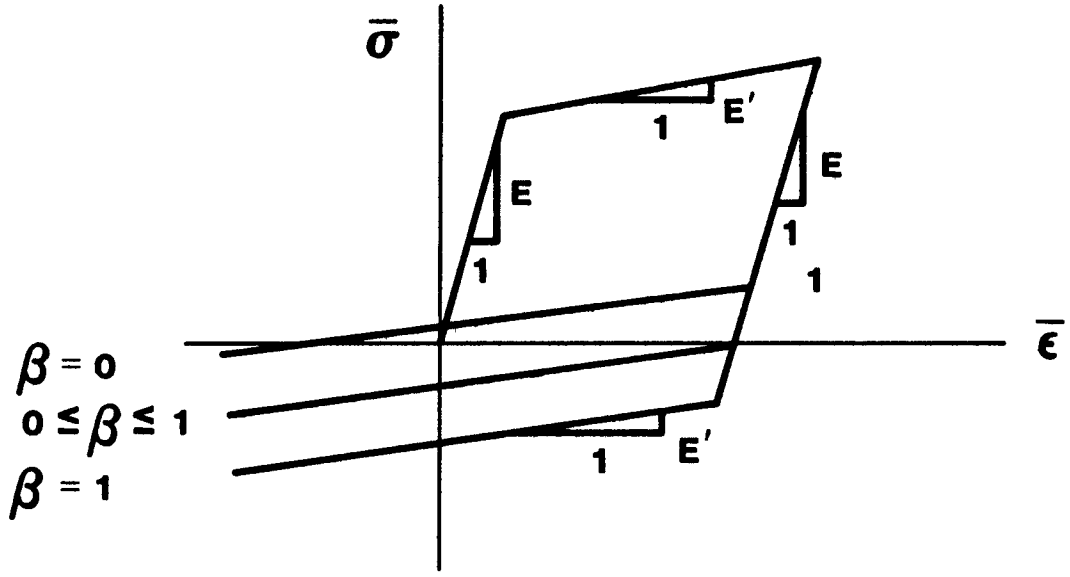


Figure 4.2.4. Effect of the Choice of the Hardening Parameter, β, on the Computed Uniaxial Response

$$\dot{\boldsymbol{\sigma}} = \mathbf{C}:(\mathbf{d} - \mathbf{d}^{pl}) = \dot{\boldsymbol{\sigma}}^{TR} \tag{4.2.42}$$

$$\dot{R} = \beta \sqrt{\frac{2}{3}}\, H' \, \bar{d}^{pl} = \beta \, \frac{2}{3} \, H' \, \gamma \tag{4.2.43}$$

$$\dot{\boldsymbol{\alpha}} = (1 - \beta) \, \frac{2}{3} \, H' \, \mathbf{d}^{pl} \tag{4.2.44}$$

$$\mathbf{d}^{pl} = \begin{cases} 0, & \text{elastic; } f(\xi) < \kappa^2 \\ \gamma \mathbf{Q}, & \text{plastic; } f(\xi) \geq \kappa^2 \end{cases} \tag{4.2.45}$$

$$\gamma = \frac{1}{(1 + H'/3\mu)} \, \mathbf{Q}:\mathbf{d} \tag{4.2.46}$$

$$\mathbf{Q} = \frac{\partial f}{\partial \xi} \Big/ \left| \frac{\partial f}{\partial \xi} \right| = \xi / R \tag{4.2.47}$$

4.2.5 Numerical Implementation

Our finite element algorithm requires an incremental form of Equations (4.2.41) through (4.2.43). Additionally, we must have an algorithm which integrates the incremental equations subject to the constraint that the stress remains on the yield surface.

The incremental analogs of Equations (4.2.42) through (4.2.44) are

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}^{TR}_{n+1} - \Delta\gamma \; 2\mu \;\; \mathbf{Q} \tag{4.2.48}$$

$$R_{n+1} = R_n + \frac{2}{3} \, \beta H' \, \Delta\gamma \tag{4.2.49}$$

and

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + (1 - \beta) \, \frac{2}{3} \, H' \, \Delta\gamma \mathbf{Q} \; . \tag{4.2.50}$$

where $\Delta\gamma$ represents the product of the time increment and the equivalent plastic strain rate ($\Delta\gamma = \Delta t \; \gamma$).

The subscripts n and n+1 refer to the beginning and end of a time step, respectively.

We also need an incremental analog to the rate forms of the consistency condition given by Equations (4.2.13), (4.2.25), and (4.2.39). At the end of the time step, we insist that the stress state must be on the yield surface. Hence the incremental consistency condition is

$$\boldsymbol{\alpha}_{n+1} + R_{n+1}\,\mathbf{Q} = \mathbf{S}_{n+1} \ . \tag{4.2.51}$$

Equation (4.2.51) is shown graphically in Figure 4.2.5.

Substituting the definitions given by Equations (4.2.48) through (4.2.50) into the consistency condition of Equation (4.2.51)

$$\left[\boldsymbol{\alpha}_n + (1-\beta)\,\frac{2}{3}\,H'\,\Delta\gamma\mathbf{Q}\right] + \left[R_n + \frac{2}{3}\,\beta H'\,\Delta\gamma\right]\mathbf{Q} = \left[\mathbf{S}^{TR}_{n+1} - \Delta\gamma 2\mu\,\mathbf{Q}\right]. \tag{4.2.52}$$

Taking the tensor product of both sides of Equation (4.2.52) with $\mathbf{Q}$ and solving for $\Delta\gamma$

$$\Delta\gamma = \frac{1}{2\mu}\,\frac{1}{(1 + H'/3\mu)}\,\left(\left|\,\boldsymbol{\xi}^{TR}_{n+1}\,\right| - R_n\right) \tag{4.2.53}$$

It follows from Equation (4.2.53) that the plastic strain increment is proportional to the magnitude of the excursion of the elastic trial stress past the yield surface (see Figure 4.2.6).

Using the result of Equation (4.2.53) in Equations (4.2.48) through (4.2.50) completes the algorithm. In addition we can compute

$$\Delta\mathbf{d}^{pl} = \mathbf{Q}\,\Delta\gamma \tag{4.2.54}$$

and

$$\Delta\bar{d}^{pl} = \sqrt{\frac{2}{3}}\,\Delta\gamma \ . \tag{4.2.55}$$

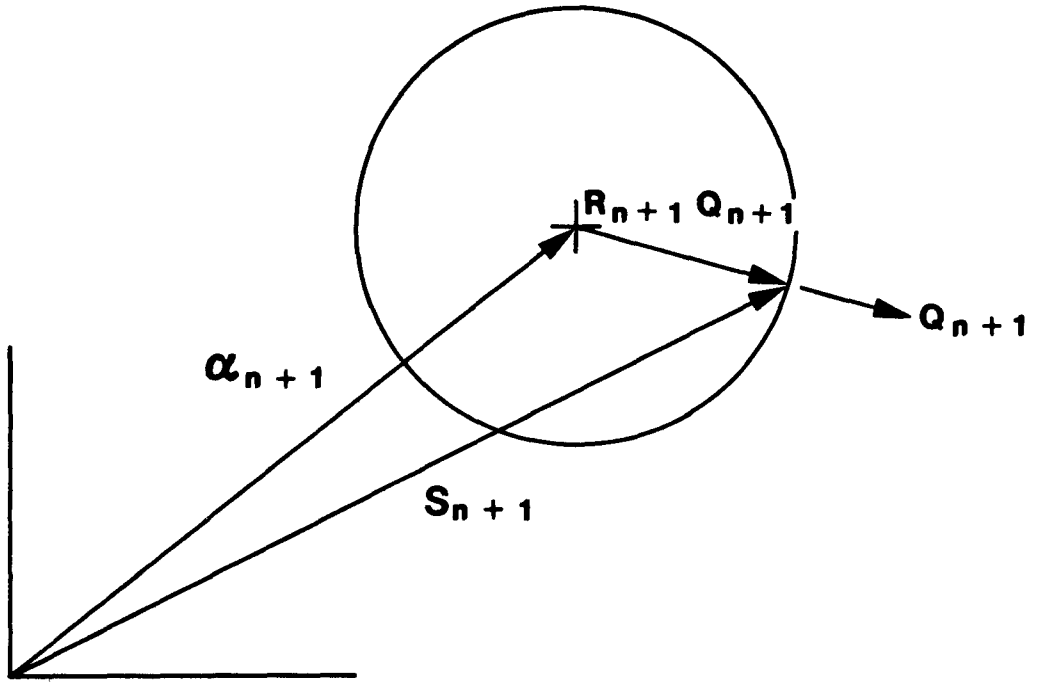


Figure 4.2.5. Geometric Interpretation of the Incremental Form of the Consistency Condition for Combined Hardening

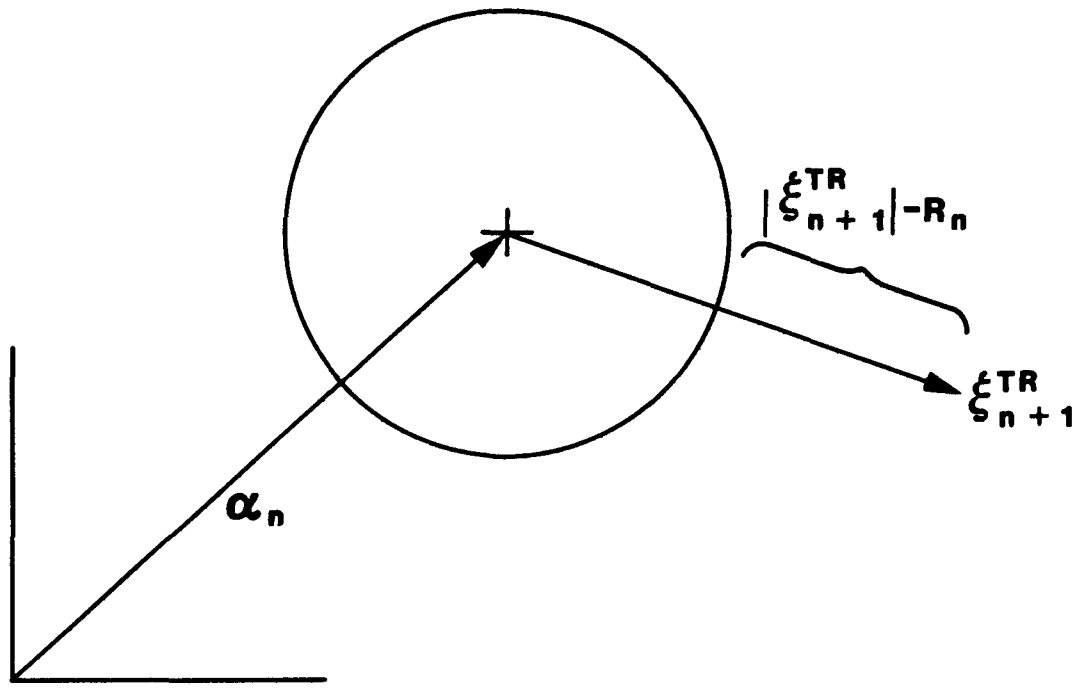


Figure 4.2.6. Geometric Interpretation of the Radial Return Correction

The results of Equation (4.2.53) applied to Equation (4.2.48) show that the final stress is calculated by returning the elastic trial stress radially to the final yield surface at the end of the time step. (Hence the derivation of the name Radial Return Method.) Estimates of the accuracy of this method and other methods for similarly integrating the rate equations are available in Krieg and Krieg [21] and Schreyer, et al. [22]. Note that the last term in Equation (4.2.48) (the radial return correction) is purely deviatoric.

The elastic plastic material model uses six internal state variables:

EQPS - equivalent plastic strain
RADIUS - current radius of yield surface
ALPHA11 - 1,1 component of backstress in unrotated configuration
ALPHA22 - 2,2 component of backstress in unrotated configuration
ALPHA33 - 3,3 component of backstress in unrotated configuration
ALPHA12 - 1,2 component of backstress in unrotated configuration

The PROP array for this material contains the following entries:

PROP(1) - Young's Modulus, E
PROP(2) - Poisson's Ratio, ν
PROP(3) - Yield Stress, σ_{yd}
PROP(4) - Hardening Modulus, H
PROP(5) - β
*PROP(6) - 2μ
*PROP(7) - 3μ
*PROP(8) - $1/(2\mu*(1 + H'/3\mu))$ (Note: $H' = H/(1 - E/H)$
*PROP(9) - λ
*PROP(10) - $2\beta \cdot H'/3$
*PROP(11) - $2(1 - \beta)H'/3$

4.3 Viscoplastic Material Model

The viscoplastic material model presented here represents a simple rate dependent plasticity model. The model is intended for relatively low strain rate ($|\mathbf{d}| < 200$) and is not recommended for high rates of impact. More details of the model can be found in Taylor and Becker [23] and Perzyna [24]. The model assumes an additive strain rate decomposition identical to the elastic plastic model

$$\mathbf{d} = \mathbf{d}^{el} + \mathbf{d}^{pl} \ . \tag{4.3.1}$$

The stress rate is assumed to be given by the elastic part of the strain rate using Hooke's law

$$\dot{\boldsymbol{\sigma}} = \mathbf{C}:\mathbf{d}^{el} = \mathbf{C}:(\mathbf{d} - \mathbf{d}^{pl}) \tag{4.3.2}$$

which can be written more clearly in index notation as

$$\dot{\sigma}_{ij} = \lambda \mathbf{d}^{el}_{kk}\, \delta_{ij} + 2\mu \mathbf{d}^{el}_{ij} \ . \tag{4.3.3}$$

We define the Von Mises equivalent stress by

$$\bar{\sigma} = \sqrt{\frac{3}{2}\, \mathbf{S}:\mathbf{S}} = \sqrt{\frac{3}{2}\, S_{ij}\, S_{ij}} \tag{4.3.4}$$

where $\mathbf{S}$ is the deviatoric part of $\boldsymbol{\sigma}$.

For this isothermal model, we use isotropic hardening only. Hence, we can write the yield stress as

$$\sigma_0 = \sigma_0(\bar{\epsilon}^{pl}) \tag{4.3.5}$$

where $\bar{\epsilon}^{pl}$ is the equivalent plastic strain. In this model, we assume isotropic hardening with a control hardening modulus, H. This is defined by identifying an equivalent plastic strain rate by

$$\bar{\sigma}\,\bar{d}^{pl} = \boldsymbol{\sigma}:\mathbf{d}^{pl} \tag{4.3.6}$$

and

$$\bar{\epsilon}^{pl} = \int_0^t \sqrt{\frac{2}{3}\,\mathbf{d}^{pl}:\mathbf{d}^{pl}}\;dt\ . \tag{4.3.7}$$

We define the yield function as

$$f(\boldsymbol{\sigma},\ \sigma_0) = \bar{\sigma} - \sigma_0(\bar{d}^{pl})\ . \tag{4.3.8}$$

The plastic strain rate is assumed to be given by a stress potential as

$$\mathbf{d}^{pl} = \partial g(\boldsymbol{\sigma})/\partial\boldsymbol{\sigma} \tag{4.3.9}$$

and we assume an associated flow rule which implies that

$$g(\boldsymbol{\sigma}) = \bar{g}(f) = g(\bar{\sigma},\ \sigma_0)\ . \tag{4.3.10}$$

Then Equation (4.3.9) can be written as

$$\mathbf{d}^{pl} = \frac{\partial g}{\partial\bar{\sigma}}\,\frac{\partial\bar{\sigma}}{\partial\boldsymbol{\sigma}}\ . \tag{4.3.11}$$

We use a power law for $\partial g/\partial\bar{\sigma}$

$$\frac{\partial g}{\partial\bar{\sigma}} = \begin{cases} \gamma\left(\dfrac{\bar{\sigma}}{\sigma_0} - 1\right)^p & \bar{\sigma} \geq \sigma_0 \\ 0 & \bar{\sigma} < \sigma_0 \end{cases} \tag{4.3.12}$$

where γ and p are material parameters.

Equation (4.3.12) indicates that the plastic strain rate is proportional to the overstress above the current value of the yield stress. Hence, the higher the overstress, the greater the plastic strain, which

leads to a reduction in the stress rate given by Equation (4.3.2) and an increase in strain hardening given by Equation (4.3.5).

Consider a uniaxial tension test. The solid curve shown in Figure 4.3.1 shows the locus of apparent yield strengths $\hat{\sigma}_y$ for mild steel at different strain rates. The apparent yield strength is the measured yield strength for a specimen from a tension test at a given strain rate. At different strain rates, different yield strengths are found. If the elastic strain rate is assumed negligible, using Equation (4.3.11) and (4.3.12), the uniaxial strain rate is

$$d = \gamma(\bar{\sigma}/\sigma_0 - 1)^p \ . \tag{4.3.13}$$

Solving Equation (4.3.13) for the Von Mises equivalent stress gives a relation for the apparent yield stress

$$\hat{\sigma}_y = \sigma_0[1 + (d/\gamma)^{1/p}] \ . \tag{4.3.14}$$

If the rate of deformation is very slow (e.g., $d \to 0$), or the fluidity constant is very large (e.g., $\gamma \to \infty$), then the yield stress given by Equation (4.3.14) is equal to the static yield stress and the static yield condition of a rate-independent constitutive theory is satisfied. If the motion is very rapid (e.g., $d \to \infty$), or the fluidity constant is zero, the response is elastic, since the value of yield stress is not restricted by Equation (4.3.14). Values of effective yield stress as a function of strain rate for different choices of the flow parameters, γ and p, are shown in Figure 4.3.1.

It follows that

$$\bar{d}^{pl} = \frac{\partial g}{\partial \bar{\sigma}} \tag{4.3.15}$$

and

$$\frac{\partial \bar{\sigma}}{\partial \sigma} = \frac{1}{\bar{\sigma}} S \ . \tag{4.3.16}$$

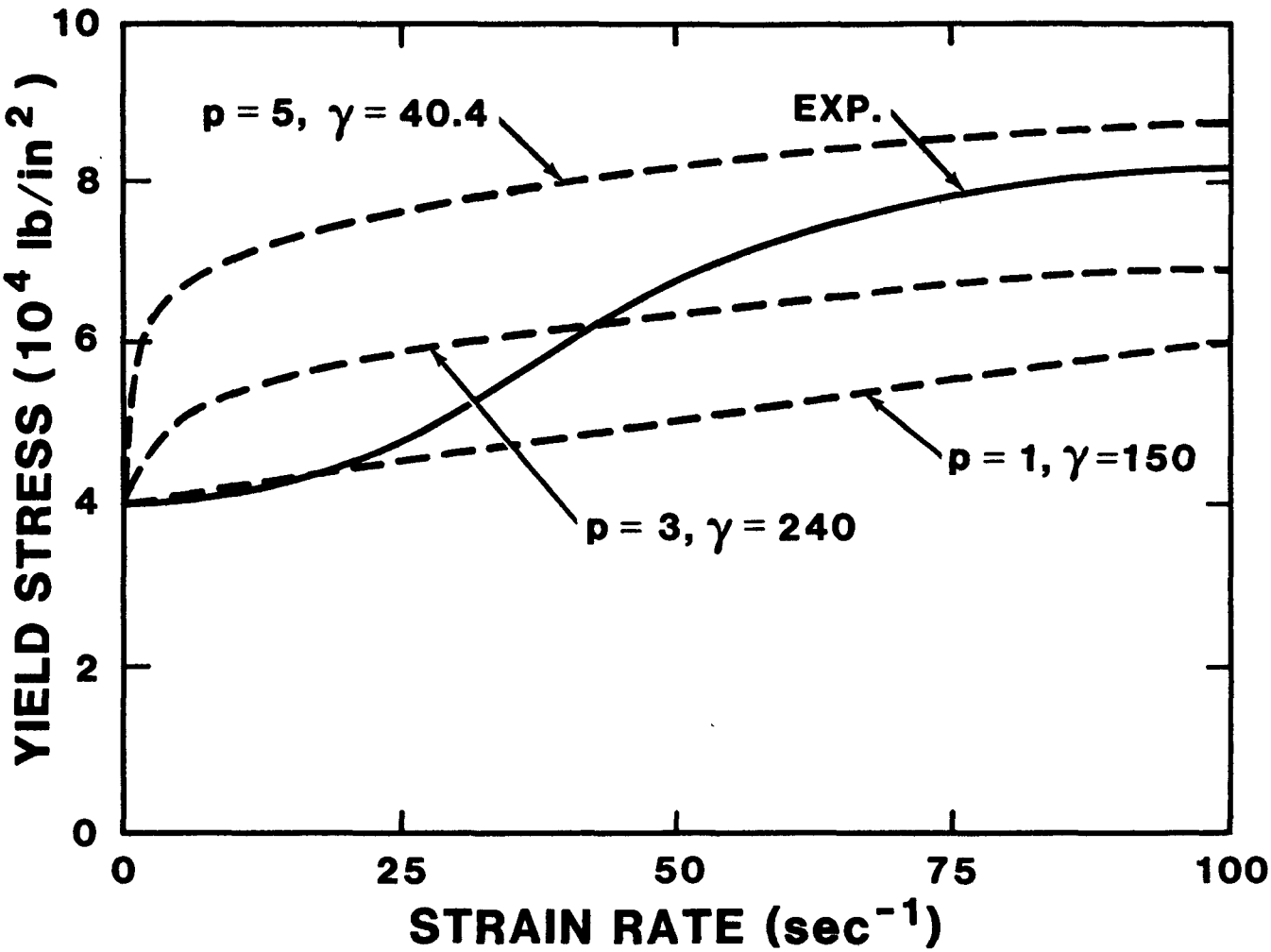


Figure 4.3.1. Yield Stress as a Function of Strain Rate as Defined by the Viscoplastic Material Model

Using these definitions of the flow potential in Equation (4.3.9) yields

$$\mathbf{d}^{pl} = \bar{d}^{pl} \frac{1}{\bar{\sigma}} \mathbf{S} \ . \qquad (4.3.17)$$

The numerical algorithm used in this model consists of a backward difference integration of the rate equations. The algorithm proceeds as follows:

1. Calculate the elastic trial stress

$$\boldsymbol{\sigma}_{n+1}^{TR} = \boldsymbol{\sigma}_n + \mathbf{C}\,\Delta t\mathbf{d}$$

2. Calculate the equivalent trial stress

$$\bar{\sigma}^{TR} = \sqrt{\frac{3}{2}\,\mathbf{S}^{TR}:\mathbf{S}^{TR}}$$

3. Check for yield

$$\bar{\sigma} - \sigma_o(\bar{d}^{pl}) \leq 0 \quad ; \quad \text{skip step 4}$$

$$\bar{\sigma} - \sigma_o(\bar{d}^{pl}) > 0 \quad \text{continue below with step 4}$$

4. Yield exceeded, calculate

$$\Delta\mathbf{d}^{pl} = \gamma\left(\frac{\bar{\sigma}^{TR}}{\sigma_o} - 1\right)^p \frac{\Delta t}{\bar{\sigma}}\,\mathbf{S}^{TR}$$

$$\Delta\boldsymbol{\sigma} = \mathbf{C}\,(\Delta t\mathbf{d} - \Delta\mathbf{d}^{pl}) = \boldsymbol{\sigma}^{TR} - 2\mu\,\Delta\mathbf{d}^{pl}$$

$$\Delta\bar{d}^{pl} = \gamma\left(\frac{\bar{\sigma}^{TR}}{\sigma_o} - 1\right)^p$$

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}_n + \Delta\boldsymbol{\sigma}$$

$$\sigma_{o\ n+1} = \sigma_o\left(\bar{d}_n^{pl} + \Delta\bar{d}_{n+1}^{pl}\right)$$

The viscoplastic material model uses two internal state variables:

EQPS – equivalent plastic strain

SIGYLD – current value of yield stress

The PROP array for this material contains the following entries:

PROP(1) – Young's Modulus, E

PROP(2) – Poisson's Ratio, ν

PROP(3) – Yield Stress, σ_{yd}

PROP(4) – Hardening Modulus, H

PROP(5) – γ

PROP(6) – p

*PROP(7) – 2μ

*PROP(8) – λ

*PROP(9) – H' = H/(1 – H/E)

4.4 Damage Model

The damage model in PRONTO is based on the work of Taylor, et al. [25] and simulates the dynamic fracture behavior of brittle rock. The essential feature of this model is the treatment of the dynamic fracture process in rock as a continuous accrual of damage where the damage mechanism is attributed to microcracking in the rock medium. The fundamental assumption of the damage model is that the material is permeated by an array of randomly distributed microcracks which grow and interact with one another under tensile loading.

The compressive response of the material is assumed to be elastic-perfectly plastic and follows the theory of Section 4.2 when the hardening modulus, H', is set to zero. In this section, we present the equations governing the tensile response of the material.

Following the work of Budiansky and O'Connell [26], we write the effective bulk modulus of a cracked medium as

$$\bar{K}/K = 1 - \frac{16}{9}\left(\frac{1 - \bar{\nu}^2}{1 - 2\bar{\nu}}\right) C_d \ . \qquad (4.4.1)$$

In Equation (4.4.1), the barred quantities represent degraded or effective quantities in the fractured medium. We denote the undegraded bulk modulus and Poisson's ratio by K and ν, respectively. The crack density, C_d, represents the volume fraction of the material made of flaws and is given by

$$C_d = \frac{45}{16} \frac{(\nu - \bar{\nu})(2 - \bar{\nu})}{(1 - \bar{\nu}^2)[10\nu - \bar{\nu}(1 + 3\nu)]} \ . \qquad (4.4.2)$$

A Wiebull distribution is used to determine the number of flaws per unit volume, N, active at a given pressure level, p (positive in tension)

$$N = k \left(\frac{p}{3K}\right)^m \ . \qquad (4.4.3)$$

where k and m are material constants.

The nominal fragment size, a, is given by an expression derived by Grady [27]

$$a = \left(\frac{\sqrt{20}\ K_{IC}}{\rho c\ \dot{\epsilon}_{max}}\right)^{2/3} \qquad (4.4.4)$$

where K_{IC} is the fracture toughness of the material, ρ is the material density, c is wave speed $\sqrt{E/\rho}$, and $\dot{\epsilon}_{max}$ is the maximum positive (tensile) strain rate the material has ever experienced.

The crack density is proportional to the number of flaws per unit volume and the nominal fragment size,

$$C_d = \beta\ N\ a^3 \ , \qquad (4.4.5)$$

where β is a proportionality constant. Combining Equations (4.4.3) through (4.4.5) gives an expression for the crack density in terms of the current pressure level and the maximum previous deviatoric strain rate, $\dot{\epsilon}_{max}$,

$$C_d = \frac{5}{2} \frac{k}{(3K)^m} \left(\frac{K_{IC}}{\rho c}\right)^2 P^m \dot{\epsilon}_{max}^{-2} \tag{4.4.6}$$

where we have absorbed the proportionality constant, β, into the material constant, k.

Combining Equations (4.4.6) and (4.4.2) gives

$$\frac{5}{2} \frac{k}{(3K)^m} \left(\frac{K_{IC}}{\rho c}\right)^2 P^m \dot{\epsilon}_{max}^{-2} = \frac{45}{16} \frac{(\nu - \bar{\nu})(2 - \bar{\nu})}{(1 - \bar{\nu}^2)[10\nu - \bar{\nu}(1 + 3\nu)]} \tag{4.4.7}$$

Equation (4.4.7) gives a relation which can be solved for the effective Poisson's ration of the degraded material. Unfortunately, for given values of C_d and ν, the equation is a cubic in $\bar{\nu}$ and the determination of $\bar{\nu}$ is a nontrivial numerical exercise. As a simplification, Equation (4.4.7) has been approximated with a linear, analytic function for $\bar{\nu}$ in terms of ν and C_d,

$$\bar{\nu} = \nu \left(1 - \frac{16}{9} C_d\right) . \tag{4.4.8}$$

The error associated with using Equation (4.4.8) instead of (4.4.7) to determine $\bar{\nu}$ is generally less than 5 percent. Once $\bar{\nu}$ is known, it is used in Equation (4.4.1) to determine the effective bulk modulus of the material. The error of the bulk modulus due to using Equation (4.4.8) instead of Equation (4.4.7) is less than 1 percent.

It is convenient to define a damage parameter, D, where $0 < D < 1$ as

$$D = \frac{16}{9} f_1(\bar{\nu}) C_d \tag{4.4.9}$$

where

$$f_1(\bar{\nu}) = \frac{(1 - \bar{\nu}^2)}{(1 - 2\bar{\nu})} . \tag{4.4.10}$$

This definition of damage follows directly from inspection of Equation (4.4.1) and results in an expression for the total mean stress or pressure as

$$P = 3K\ (1 - D)\ \epsilon_v\ , \tag{4.4.11}$$

where ϵ_v is the mean volumetric strain $\left(\int \frac{1}{3}\ \mathrm{tr}\ \mathbf{d}\ dt \right)$.

We assume that the deviatoric response of the material is degraded in the same manner as the bulk response,

$$\mathbf{S} = 2\mu\ (1 - D)\ \mathbf{e}\ , \tag{4.4.12}$$

where $\mathbf{S}$ is the deviatoric part of the stress and $\mathbf{e}$ is the deviatoric part of the strain.

Taking the rate of Equations (4.4.9) through (4.4.12);

$$\dot{D} = \frac{16}{9}\ f_1(\bar{\nu})\ \dot{C}_d + \frac{16}{9}\ C_d\ \dot{f}_1(\bar{\nu}), \tag{4.4.13}$$

$$\dot{P} = 3K\ (1 - D)\ \dot{\epsilon}_v - 3K\ \epsilon_v\ \dot{D}\ , \tag{4.4.14}$$

$$\dot{\mathbf{S}} = 2\mu\ (1 - D)\ \dot{\mathbf{e}} - 2\mu\ \mathbf{e}\ \dot{D}\ , \tag{4.4.15}$$

and

$$\dot{f}_1(\bar{\nu}) = \frac{\partial f_1}{\partial \bar{\nu}}\ \frac{\partial \bar{\nu}}{\partial C_d}\ \dot{C}_d\ . \tag{4.4.16}$$

It follows that

$$\dot{D} = \frac{16}{9}\left[f_1(\bar{\nu}) - \frac{16}{9}\ \nu\ f_2(\bar{\nu})\ C_d \right] \dot{C}_d \tag{4.4.17}$$

where

$$f_2(\bar{\nu}) = \frac{2(1 - \bar{\nu} + \bar{\nu}^2)}{(1 - 2\bar{\nu})^2} \tag{4.4.18}$$

and

$$\dot{C}_d = \frac{5}{2} \frac{km}{(3K)^{m-1}} \left(\frac{K_{IC}}{\rho c}\right)^2 p^{m-1} \dot{\epsilon}_v \dot{\epsilon}_{max}^{-2} . \qquad (4.4.19)$$

Equations (4.4.14), (4.4.15), (4.4.17), and (4.4.19) represent seven coupled ordinary differential equations to be integrated over each time step. In PRONTO, we use a simple forward difference integration operator.

The damage material model requires six material constants to characterize the material. Young's modulus, Poisson's ratio, the yield stress in compression, and the fracture toughness of the material are all conventional material properties readily obtained from standard material tests. The remaining two material constants are k and m for the Wiebull distribution of Equation (4.4.3). Determining these material constants requires data relating the fracture stress of the material to the strain rate. If this data is available, the two constants can be determined as follows. The logarithm of the fracture stress versus the logarithm of the uniaxial strain rate is generally a straight line,

$$\ln(\sigma_F) = C_o + \frac{2}{m} \ln(\dot{\epsilon}) . \qquad (4.4.20)$$

Two points on the fracture stress versus strain rate curve can be used with Equation (4.4.20) to determine m. The other constant, k, is determined from

$$\ln(k) = m \left[\ln\left(\frac{3Km}{\sigma_F}\right) - \frac{m+1}{m} \ln(m+1) \right] - \ln\left[\frac{40}{9} \left(\frac{K_{IC}}{\rho c}\right)^2 \dot{\epsilon}^{-2} \right]. \qquad (4.4.21)$$

If laboratory data for fracture stress versus strain rate are not available, it is possible to generate this data using an expression derived by Kipp, et al. [28],

$$\sigma_F = \left(\frac{9\pi \, E \, K_{IC}^2}{16 \, N_S^2 \, c_S} \right)^{1/3} \dot{\epsilon}^{1/3} \qquad (4.4.22)$$

where N_S is a shape factor (1.12 for penny shaped cracks) and c_S is the shear wave velocity of the material. Equation (4.4.22) has been shown to be a reasonable approximation for a number of rock types.

The damage material model uses five internal state variables:

EQPS – equivalent plastic strain
DAMAGE – damage (Equation (4.4.9))
EVMAX – maximum volumetric tensile strain experienced by the material
FRAGSIZ – average fragment diameter (Equation (4.4.4))
CRAKDENS – crack density (Equation (4.4.2))

The PROP array for this material type contains the following entries:

PROP(1) – Young's Modulus, E
PROP(2) – Poisson's Ratio, ν
PROP(3) – Yield Stress, σ_{yd}
PROP(4) – m
PROP(5) – k
PROP(6) – Fracture Toughness, K_{IC}
*PROP(7) – Bulk Modulus, K
*PROP(8) – μ
*PROP(9) – m – 1
*PROP(10) – COND = $\frac{5}{2}$ k m $K_{IC}^2/(\rho c)^2$ (Note: $c = \sqrt{E/\rho}$)
*PROP(11) – CONA = $\frac{1}{2}$ $(\sqrt{20}\ K_{IC}/\rho c)^{2/3}$

4.5 Soils and Crushable Foams Model

The soils and crushable foams model in PRONTO is a direct descendent of the model developed by Krieg [29]. Reference [29] is an unpublished Sandia National Laboratories report and is not readily available. The model was described in detail by Swenson and Taylor [30] as it was incorporated into a tensile failure model. One major difficulty with the original version of

this material model which has confounded users is that the pressure dependence of the yield stress is expressed in terms of J^2, the second invariant of the stress tensor. We have reformulated the model so that the yield stress is written directly in terms of the pressure. NOTE: this means that old data must be converted.

The yield surface assumed is a surface of revolution about the hydrostat in principal stress space as shown in Figure 4.5.1. In addition, a planar end cap on the normally open end is assumed. The yield stress is specified as a polynomial in pressure, p (positive in compression)

$$\sigma_{yd} = a_o + a_1 P + a_2 P^2 . \qquad (4.5.1)$$

The determination of the yield stress from Equation (4.5.1) places severe restrictions on the admissible values of a_o, a_1, and a_2. There are three valid cases as shown in Figure 4.5.2. First, the user may specify a positive a_o, and a_1 and a_2 equal to zero as shown in Figure 4.5.2a. This gives an elastic–perfectly plastic deviatoric response, and the yield surface is a cylinder oriented along the hydrostat in principal stress space. Second, a conical yield surface (Figure 4.5.2b) is given by setting a_2 to zero and entering appropriate values of $a_o$and a_1. The program checks the users input to determine whether a valid (negative) tensile fracture pressure, P_{fr}, results from the input data. The third case results when all three constants are nonzero and the program detects that a valid negative tensile failure pressure can be derived from the data. This case is shown in Figure 4.5.2c. A valid set of constants for the third case results in a parabola as shown in Figure 4.5.2.c. We have drawn the descending portion of the curve with a dashed line indicating that the program does not use that portion of the curve. Instead, when the pressure exceeds P*, the yield stress is held constant as shown at the maximum value.

The plasticity theories for the volumetric and deviatoric parts of the material response are completely uncoupled. The volumetric response is computed first. The mean pressure, P, is assumed to be positive in compression and a yield function is written for the volumetric response as

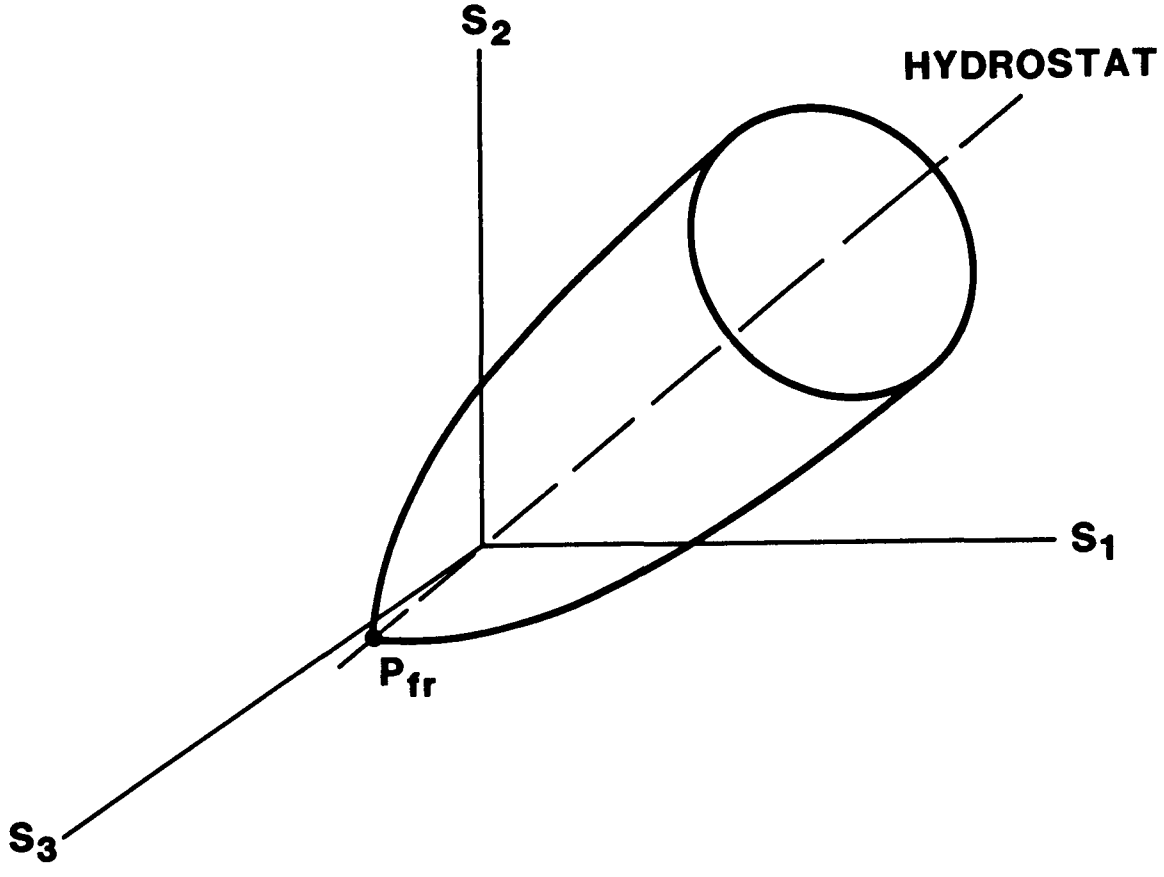


Figure 4.5.1. Pressure Dependent Yield Surface for the Soils and Crushable Foams Material Model

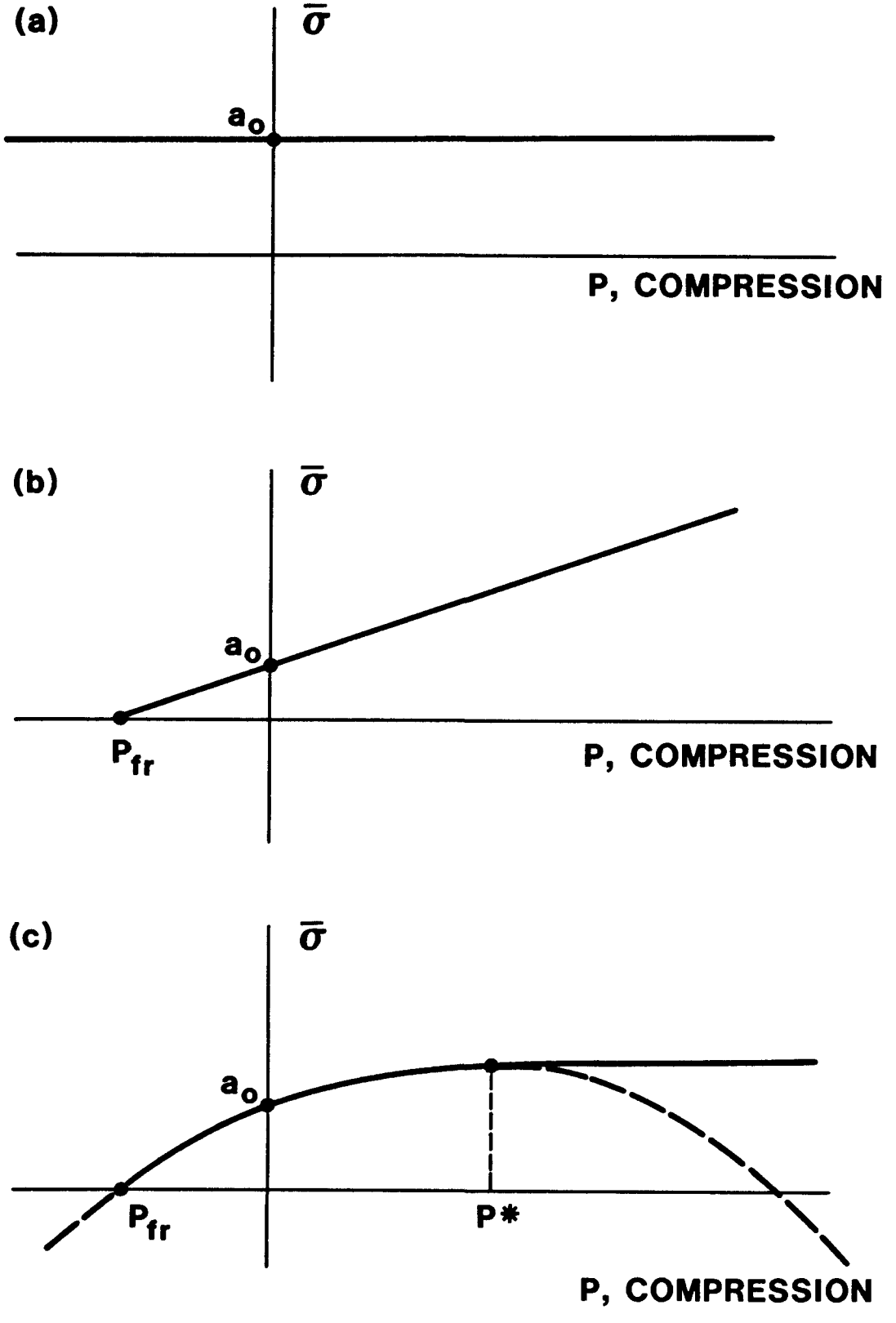


Figure 4.5.2. Forms of Valid Yield Surface Which can be Defined for the Soils and Crushable Foams Material Model

$$\phi_p = P - f_p(\epsilon_v) \tag{4.5.2}$$

where $f_p(\epsilon_v)$ defines the volumetric stress–strain curve for the pressure as shown in Figure 4.5.3. This function is defined by the user with the restriction that the slope of the function must be less than or equal to the unloading bulk modulus, K_o, everywhere. If the user wishes the volumetric response to be purely elastic, he simply specifies no function identification (e.g., FUNCTION ID = 0). The yield function, ϕ_p, determines the motion of the end cap along the hydrostat.

The mean volumetric strain is updated as

$$\epsilon_v^{n+1} = \epsilon_v^n + \Delta t \cdot \dot{\epsilon}_v \tag{4.5.3}$$

where $\dot{\epsilon}_v$ is the volumetric part of the strain rate $\left(\dot{\epsilon}_v = \frac{1}{3} \text{ tr } \mathbf{d}\right)$.

There are three possible regimes of the pressure–volumetric strain response. Tensile failure is assumed to occur if the pressure becomes smaller (more negative) than P_{fr}. The quantity ϵ_{fr} is initialized to $-P_{fr}/K_o$ by the program. If tensile failure is detected, the pressure is set to $-P_{fr}$. Remember, pressure is negative in tension! Failure by monotonic tensile loading is shown in Figure 4.5.4a. As long as $\epsilon_v < \epsilon_{fr}$, the pressure will remain equal to $-P_{fr}$.

If the volumetric strain exceeds ϵ_{fr}, a check is then made to see if

$$\epsilon_v < \epsilon_u \tag{4.5.4}$$

where ϵ_u is the most positive (compressive) volumetric strain previously experienced by the material, set initially to zero by the program. If Equation (4.5.4) is satisfied, the step is elastic and,

$$P^{n+1} = P^n - K_o \, \Delta\epsilon_v \, . \tag{4.5.5}$$

This elastic response is shown in Figure 4.5.4b.

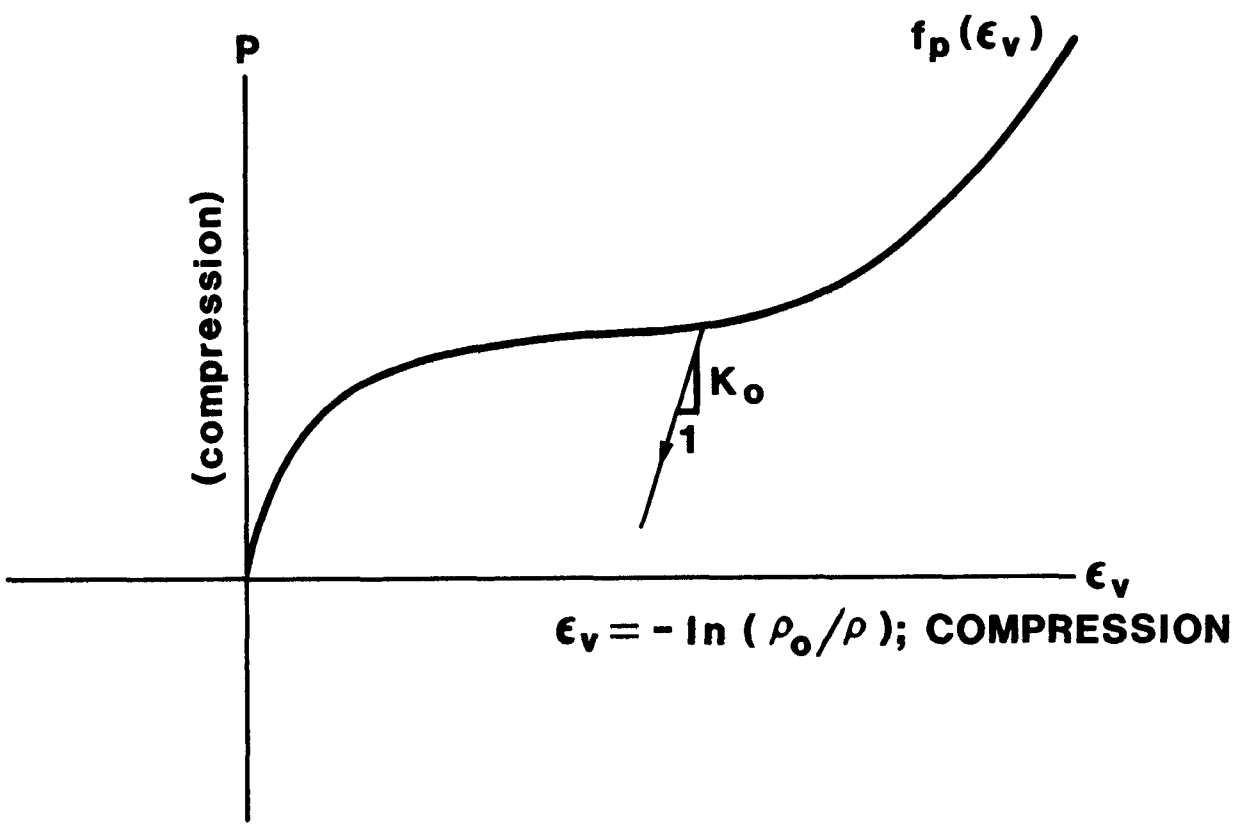


Figure 4.5.3. Pressure Versus Volumetric Strain Curve in Terms of a User Defined Curve, $F(\epsilon_v)$, for the Soils and Crushable Foams Material Model

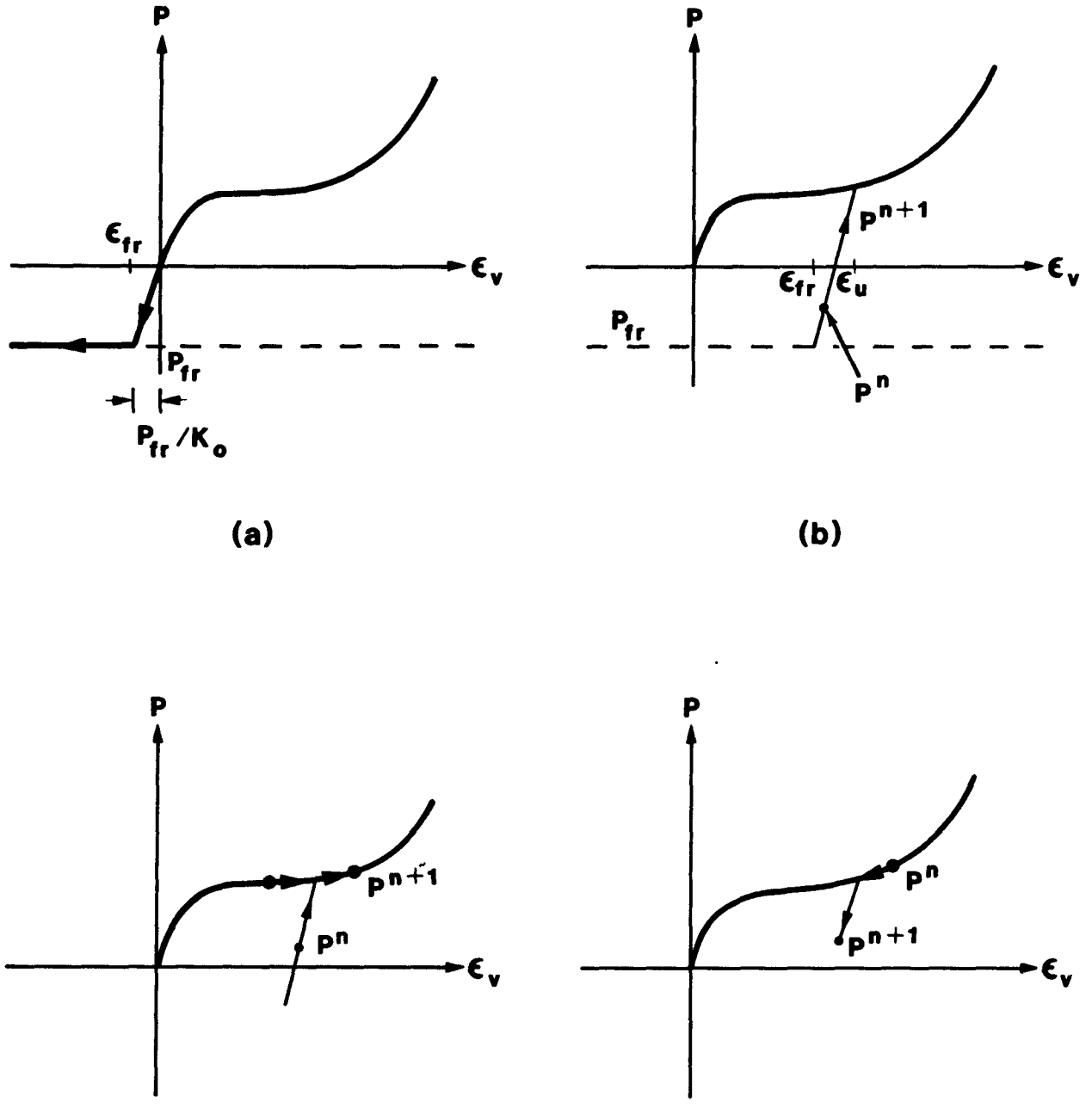


Figure 4.5.4. Possible Loading Cases for the Pressure Versus Volumetric Strain Response Using the Soils and Crushable Foams Material Mode

If Equation (4.5.4) is not satisfied, the volumetric response is along the curve defined by $f_p(\epsilon_v)$ and

$$p^{n+1} = f_p\left(\epsilon_v^{n+1}\right) \tag{4.5.6}$$

and we set

$$\epsilon_u = \epsilon_v^{n+1} \ . \tag{4.5.7}$$

This response is shown in Figure 4.5.4c. Note, that if Equation (4.5.5) is used to determine P, we also drag ϵ_{fr} along so that if we unload from the curve, $f_p(\epsilon_v)$, we will fracture at the appropriate strain level as shown in Figure 4.5.4d.

The deviatoric part of the response is computed next and uses a conventional plasticity theory with radial return. See Krieg and Krieg [21]. The trial elastic deviatoric stresses are computed as

$$S^{TR} = S_n + 2G\ \Delta t\ \dot{e} \tag{4.5.8}$$

where $\dot{e}$ is the deviatoric part of the strain rate. The current value of yield stress is calculated using Equation (4.5.1) and the Von Mises effective stress, $\bar{\sigma}$, is computed as

$$\bar{\sigma} = \sqrt{\frac{3}{2}\ S{:}S} \ . \tag{4.5.9}$$

The yield condition is checked to determine whether $\bar{\sigma} < \sigma_{yd}$. If this is the case, the trial stress is the correct deviatoric stress at the end of the time step, $S_{n+1} = S^{TR}$. If yield is exceeded, a simple radial return is performed to calculate the deviatoric stress at the end of the time step

$$S_{n+1} = \frac{\sigma_{yd}}{\bar{\sigma}}\ S^{TR} \ . \tag{4.5.10}$$

Finally, the total stress is determined by

$$\sigma_{n+1} = S_{n+1} + P_{n+1}\,\delta \ . \qquad (4.5.11)$$

The Soils and Crushable Foams model uses four internal state variables:

EVMAX – maximum compressive volumetric strain experienced (always positive)

EVFRAC – current value of volumetric fracture strain (positive in compression)

EV – current value of volumetric strain (positive in compression)

NUM – integer pointing to the last increment in the pressure function where the interpolate was found

The PROP array contains the following entries for this material:

PROP(1) – 2μ

PROP(2) – Bulk Modulus, K

PROP(3) – a_0

PROP(4) – a_1

PROP(5) – a_2

PROP(6) – Function Id number

4.6 Low Density Foams

The low density foams model presented here was developed by Morgan, Krieg, and Neilsen [31] and is based on results from experimental tests on low density, closed-cell polyurethane foams. These foams having densities ranging from 2 to 10 pounds per cubic foot have been proposed for use as energy absorbers in nuclear waste shipping containers. Representative responses of closed-cell polyurethane foams for various hydrostatic, uniaxial and triaxial laboratory test conditions are shown in Figures 4.6.1 and 4.6.2. These results indicate that the volumetric response of the foam is highly dependent on load history. This implies that typical decompositions of total foam response into an independent volumetric part and a mean stress (pressure) dependent deviatoric part are not valid for this class of

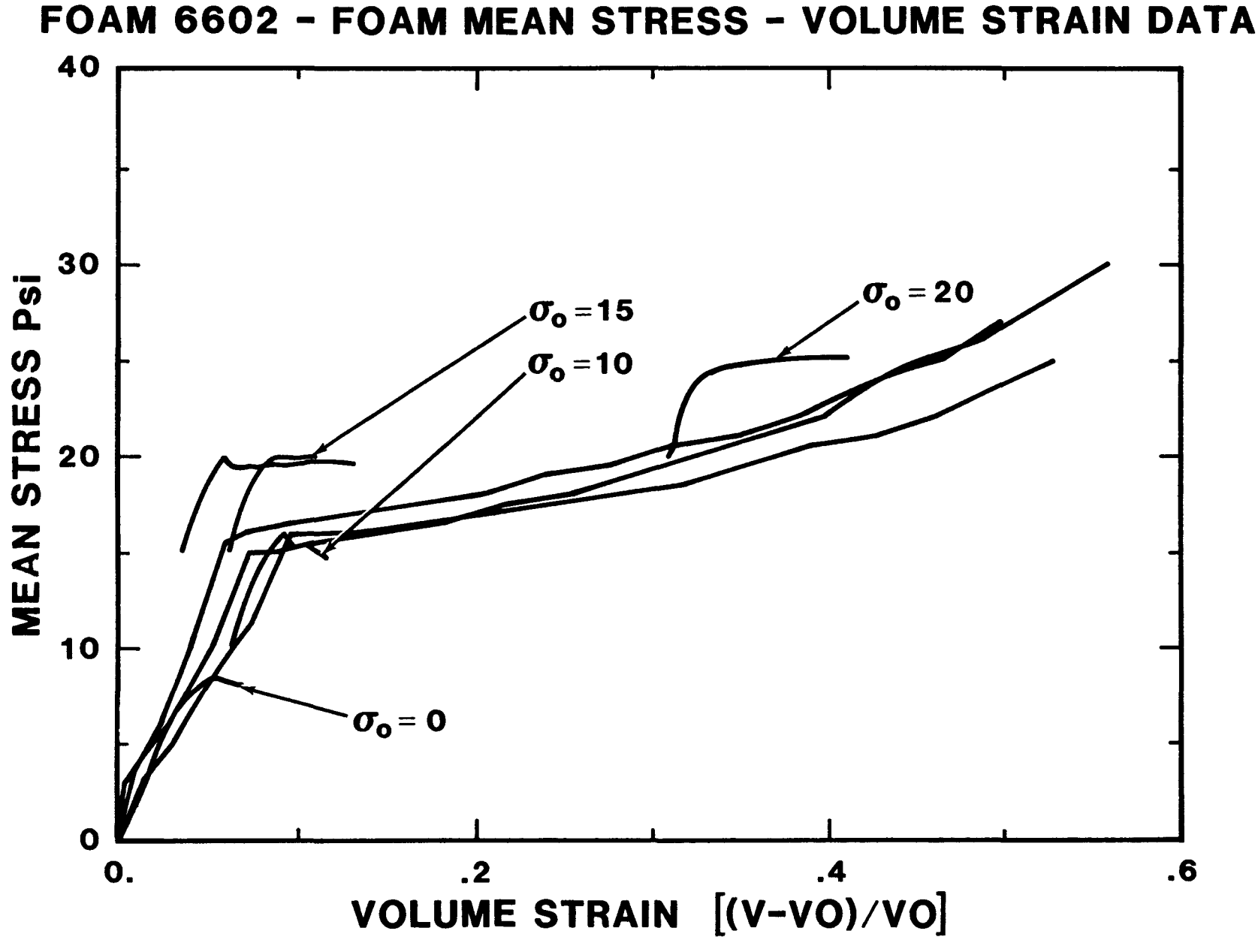


Figure 4.6.1. Foam Volume Strain Versus Mean Stress for 6602 Foam at Various Confining Pressures

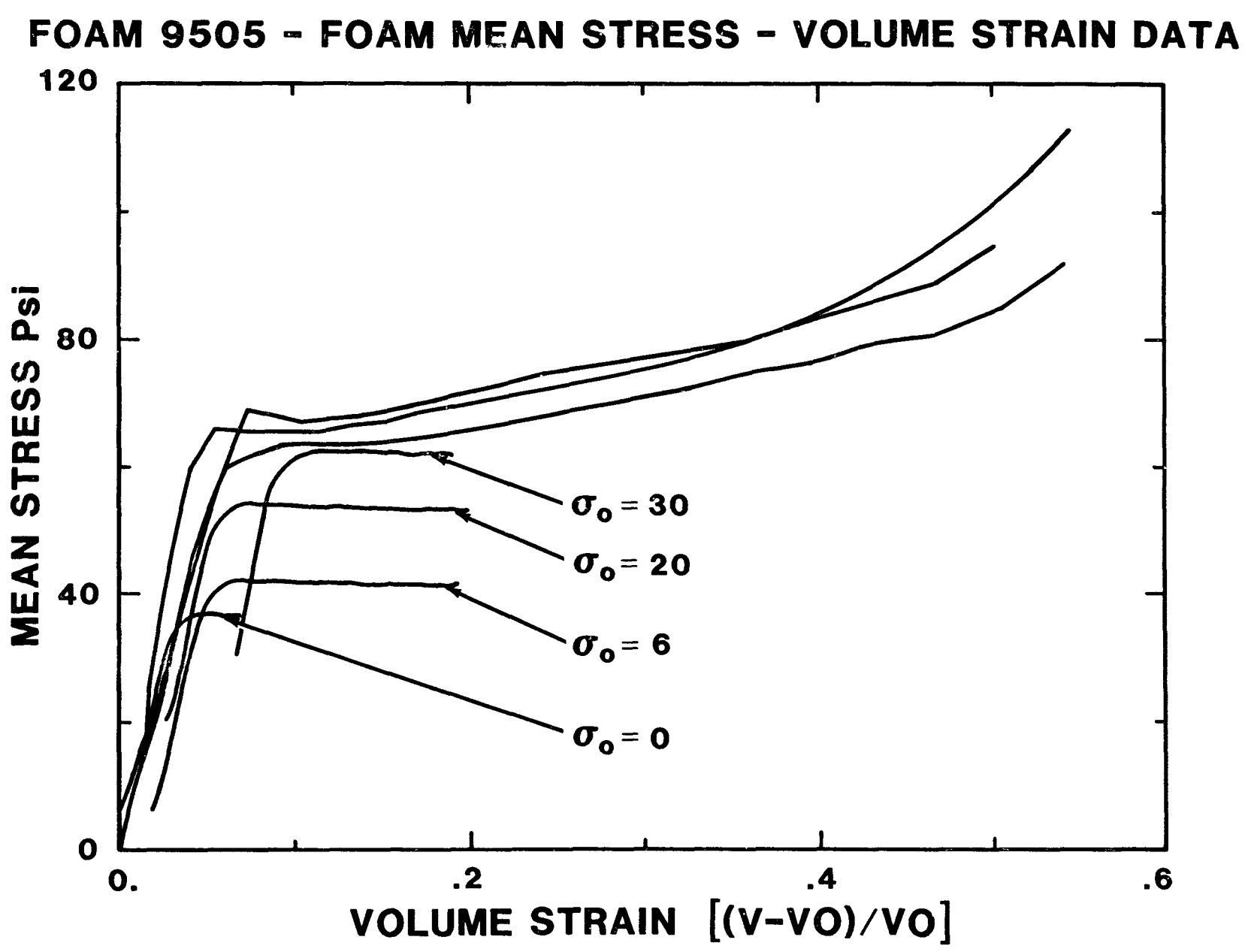


Figure 4.6.2. Foam Volume Strain Versus Mean Stress for 9505 Foam at Various Confining Pressures

foam. Many "soil and crushable foam" models, including the other foam model described in Section 4.5, use such decompositions and hence are not valid for low density closed-cell polyurethane foams. The model presented here reproduces experimental test responses more accurately for this class of foams than the model in Section 4.5.

The experimental tests on which this model is based were performed by the Civil Engineering Research Facility of the University of New Mexico with the results reported in [31]. Foam samples were subjected to static, compressive stresses during these tests. In most of the tests, air was trapped in the closed cells of the foams and could not escape because the samples were jacketed with an impervious material. In this constitutive model, the total foam response is decomposed into contributions from the skeleton and from air trapped in the closed cells of the foam. The contribution of the air to the total foam response is dependent on the application. If the foam is used in a vented application where the air can escape, the contribution of the air is zero and the foam and skeleton responses are identical. If the foam is used in an application where the air cannot escape, such as a sealed shipping container, the foam pressure is considered to be the sum of pressure carried by the skeleton and the air pressure. That is,

$$P_F = P_{sk} + P_{air} \tag{4.6.1}$$

where P_F and P_{sk} are the mean stresses (first invariants of the stress tensor divided by three) of the foam and skeleton, respectively. The mean stresses and air pressure are assumed positive in tension. The air pressure is determined from

$$P_{air} = \frac{p_o \gamma}{1 + \gamma - \phi} \tag{4.6.2}$$

where γ is the engineering volume strain (first invariant of the total strains) which is positive in tension and p_o and ϕ are model parameters. The parameter p_o is the initial foam pressure (usually atmospheric pressure of 14.7 psi), and ϕ is the ratio of the foam density to the polymer density from which the foam is produced.

Test data indicate that the skeleton response in any principal stress direction is independent of loading in any other principal stress direction. Thus, Poisson's ratio for the skeleton is equal to zero. Test data also indicate that the yield strength of the skeleton in any principal stress direction can be expressed in terms of the engineering volume strain and the second invariant of the deviatoric strains with the following relationship

$$f_1 = \begin{cases} A + B(1 + C\gamma); & II'_\epsilon > 0 \\ B(1 + C\gamma); & II'_\epsilon = 0 \end{cases} \qquad (4.6.3)$$

is the second invariant of the deviatoric strain tensor; γ is the engineering volume strain as in Equation (4.6.2); and A, B, and C are constants determined from fitting Equation (4.6.3) to the laboratory data. Constants B and C are determined from hydrostatic test data where II'_ϵ is zero, and A is determined from any test where the loading is deviatoric.

Numerical implementation of the model is as follows. Foam stresses and strains from the previous time increment are saved. At the beginning of the next time increment, the old skeleton stresses are computed from the old foam stresses and the old air pressure. The strain rates for the new time increment are used to determine new strain increments and trial elastic stress increments for the skeleton. These stress increments are added to the old skeleton stresses to produce new trial stresses for the skeleton. The trial skeleton stresses are then rotated to principal stress directions and compared with the yield stress determined from Equation (4.6.3). If yield occurs, the skeleton stresses are set to the yield stress. If yield does not occur, the trial skeleton principal stresses become the final skeleton principal stresses. The final skeleton stresses are obtained by rotating the final skeleton principal stresses back to the unrotated configuration. Then, the final foam stresses are obtained by adding the air pressure contribution for the new strain state to the new skeleton stresses.

Input parameters for the model are the constants E, p_0, ϕ, A, B, and C which are defined above. If the foam is used in an application where the

air can escape, p_o should be input as zero. Otherwise, p_o is the atmospheric pressure at the beginning of the simulation.

There are no internal state variables for this model.

The PROP array contains the following entries for this material type:

PROP(1) – Young's Modulus, E
PROP(2) – A
PROP(3) – B
PROP(4) – C
PROP(5) – NAIR
PROP(6) – P_o
PROP(7) – ϕ

4.7 Hydrodynamic Materials

All of the equations of state described in Chapter 5 are used by specifying a hydrodynamic material type. This material type has only volumetric or mean stress response and no deviatoric response. The pressure is calculated in the equation of state and the stress is set as

$$\sigma = -p\ \delta \qquad (4.7.1)$$

Note that the pressure is assumed positive in compression in the equations of state.

The hydrodynamic material requires the user to input a pressure cutoff which is positive in compression.

There are no internal state variables for this material type.

The PROP array contains only one entry for this material:

PROP(1) – Pressure cutoff.

4.8 Rate and Temperature Dependent Plasticity

In this section, we discuss briefly the formulation and numerical implementation of a unified creep plasticity model which was proposed by Bammann [32], and later modified and cast into a form which allows efficient implementation into a finite element code by Bammann and Johnson [33]. The description here follows the same notation as that for the rate independent combined hardening plasticity model described in Section 4.2.

The kinematics and thermodynamics of finite deformation as well as the numerical algorithm developed in reference [32] are valid for any of the class of unified creep plasticity models discussed by Bammann and Krieg [34]. Unified creep plasticity models are formulated without the introduction of a yield surface. This is accomplished by proposing a constitutive equation for the plastic part of the strain rate which is near zero when response is elastic and increases rapidly at yield, thus simulating an elastic limit. This model represents a combined hardening model where the history dependence is characterized by two internal state variables, a scalar, κ, for the isotropic hardening and a second order tensor, $\boldsymbol{\alpha}$, for the kinematic hardening. These internal state variables have the same definition as for the isotropic and kinematic hardening cases described in Sections 4.2.2 and 4.2.3 for the rate independent plasticity model. The crucial difference is that the evolution equations for the internal state variables are motivated by the specific dislocation processes which they represent.

The specific model proposed in [32] and [33] is summarized as,

$$\dot{\boldsymbol{\sigma}} = \hat{\lambda} \mathrm{tr}(\mathbf{d})\, \boldsymbol{\delta} + 2\mu(\mathbf{d} - \mathbf{d}^{pl}) \tag{4.8.1}$$

$$\mathbf{d}^{pl} = r(\theta)\, \sinh\left[\frac{|\boldsymbol{\xi}| - \kappa - Y(\theta)}{V(\theta)}\right] \frac{\boldsymbol{\xi}}{|\boldsymbol{\xi}|} \tag{4.8.2}$$

$$\dot{\boldsymbol{\alpha}} = (1 - \beta)\, k(\theta)\, \mathbf{d}^{pl} - \frac{g(\theta) + h(\theta)|\mathbf{d}^{pl}|}{1 - \beta} |\boldsymbol{\alpha}| \boldsymbol{\alpha} \tag{4.8.3}$$

$$\dot{\kappa} = \beta k(\theta)\, |\mathbf{d}^{pl}| - \frac{g(\theta) + h(\theta)\, |\mathbf{d}^{pl}|}{\beta} \kappa^2 \tag{4.8.4}$$

where θ is temperature and the functions, $r(\theta)$, $Y(\theta)$, $V(\theta)$, $k(\theta)$, $g(\theta)$, and $h(\theta)$ will be defined below. In Equation (4.8.2), $\boldsymbol{\zeta}$ is a deviatoric effective stress defined by

$$\boldsymbol{\zeta} = \frac{3}{2}\,\mathbf{S} - \boldsymbol{\alpha} \tag{4.8.5}$$

where $\mathbf{S}$ is the deviatoric part of the unrotated Cauchy stress defined by Equation (4.2.1). Note that $\boldsymbol{\zeta}$ is a different measure of effective stress than that defined by the backstress, $\boldsymbol{\xi}$, of Equation (4.2.2).

The assumption of a nonconducting temperature change is given by

$$\dot{\theta} = \frac{1}{\rho C_v}\left[\boldsymbol{\zeta}:\mathbf{d}^{pl} + \frac{h(\theta)\,|\mathbf{d}^{pl}| + g(\theta)}{k(\theta)}\,(|\boldsymbol{\alpha}|^3 + \kappa^3)\right] \tag{4.8.6}$$

where C_v is specific heat.

This can be reasonably approximated for moderate strains by

$$\dot{\theta} = \frac{.95}{\rho C_v}\,(\boldsymbol{\sigma}:\mathbf{d}^{pl}) \tag{4.8.7}$$

To implement this model, the flow rule defined by Equation (4.8.2) is inverted and the dependence upon the plastic stretching is replaced by a dependence upon the total stretching. This results in a rate dependent Mises type flow surface of the form

$$f(\boldsymbol{\zeta},\mathbf{d},k,\theta) = |\boldsymbol{\zeta}| - \kappa - \chi(|\mathbf{d}|,\theta) \tag{4.8.8}$$

where

$$\chi(|\mathbf{d}|,\theta) = Y(\theta) + V(\theta)\,\sinh^{-1}\left[\frac{|\mathbf{d}|}{r(\theta)}\right] \tag{4.8.9}$$

The flow rule defined by Equation (4.8.8) is similar to that defined by Equation (4.2.4) for the rate independent case except that it depends on strain rate and temperature. A similar substitution for the plastic strain rate is utilized in the dynamic recovery terms resulting in a simplified

flow rule. More specifically, $|\mathbf{d}^{pl}|$ is replaced by $|\mathbf{d}|$ in the recovery terms in Equations (4.8.3) and (4.8.4) (see Equations (4.8.11) and (4.8.12)). While this is convenient, it is not essential as discussed in reference [33].

The model is now in a form permitting implementation utilizing the radial return method described in general in Section 4.2.4 and in particular for this model in reference [33]. The nonconducting temperature change is included by use of an operator split, that is, the stress is updated assuming isothermal conditions and the temperature change is then calculated based upon the new values for the stress, plastic strain rate, and internal state variables. We begin by assuming elastic behavior and calculate a trial deviatoric stress and trial internal state variables,

$$S^{TR} = S_n + \int_{t_n}^{t_{n+1}} 2\mu \dot{e} dt \tag{4.8.10}$$

$$\boldsymbol{\alpha}^{TR} = \boldsymbol{\alpha}_n - \int_{t_n}^{t_{n+1}} \frac{[h(\theta)\ |\mathbf{d}| + g(\theta)]\ |\boldsymbol{\alpha}|\boldsymbol{\alpha}}{1-\beta}\, dt \tag{4.8.11}$$

$$\kappa^{TR} = \kappa_n - \int_{t_n}^{t_{n+1}} \frac{[h(\theta)\ |\mathbf{d}| + g(\theta)]\ \kappa^2}{\beta}\, dt \tag{4.8.12}$$

where $\dot{\mathbf{e}}$ is the deviatoric part of $\mathbf{d}$.

The elastic assumption is now checked by substitution into the yield condition of Equation (4.8.8)

$$\phi = \left| \frac{3}{2} S^{TR} - \boldsymbol{\alpha}^{TR} \right| - \kappa^{TR} - \chi(|\mathbf{d}|,\theta) \tag{4.8.13}$$

If $\phi \leq 0$, the assumption of elasticity is verified and we proceed to the next time step with the trial values of the variables

$$\sigma_{n+1} = S^{TR} + \frac{3\hat{\lambda} + 2\mu}{3} \int_{t_n}^{t_{n+1}} tr(\mathbf{d})\ \boldsymbol{\delta}\ dt \tag{4.8.14}$$

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}^{TR} \tag{4.8.15}$$

$$\kappa_{n+1} = \kappa^{TR} \tag{4.8.16}$$

If, however, the stress point lies outside the flow surface, $\phi > 0$, the values of the variables at the end of the time step must have the terms associated with the plastic part of the strain rate

$$S_{n+1} = S^{TR} - \int_{t_n}^{t_{n+1}} 2\mu \mathbf{d}^{pl}\ dt \tag{4.8.17}$$

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}^{TR} + \int_{t_n}^{t_{n+1}} (1 - \beta)\ k(\theta)\ \mathbf{d}^{pl}\ dt \tag{4.8.18}$$

$$\kappa_{n+1} = \kappa^{TR} + \int_{t_n}^{t_{n+1}} \beta k(\theta)\ |\mathbf{d}^{pl}|\ dt \tag{4.8.19}$$

The obvious problem, just as in the rate independent case, is how to determine $\Delta\gamma$. We take a similar approach as in the rate independent case. We assume normality for the plastic flow and assume that the direction of the flow at the end of the step is the same as the trial (elastic) direction,

$$\int_{t_n}^{t_{n+1}} \mathbf{d}^{pl}\ dt \simeq \Delta\gamma\ \frac{\boldsymbol{\xi}^{TR}}{|\boldsymbol{\xi}^{TR}|} \tag{4.8.20}$$

We calculate the flow rule from the numerical consistency equation just as in Section 4.2.5, except we use the flow rule defined by Equation (4.8.8). Solving for $\Delta\gamma$,

$$\Delta\gamma = \frac{|\boldsymbol{\xi}^{TR}| - \chi(|\mathbf{d}|,\theta)}{3\mu + k(\theta)} - \kappa^{TR} \tag{4.8.21}$$

The values for the stress and internal state variables at the end of the time step are then given by

$$S_{n+1} = S^{TR} - 2\mu\Delta\gamma \frac{\xi^{TR}}{|\xi^{TR}|} \tag{4.8.22}$$

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}^{TR} + \frac{3\hat{\lambda} + 2\mu}{3} \int_{t_n}^{t_{n+1}} \mathrm{tr}(\mathbf{d})\, \boldsymbol{\delta} dt \tag{4.8.23}$$

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}^{TR} + (1 - \beta)\, k(\theta)\, \Delta\gamma \frac{\xi^{TR}}{|\xi^{TR}|} \tag{4.8.24}$$

$$\kappa_{n+1} = \kappa^{TR} + \beta k(\theta)\, \Delta\gamma \tag{4.8.25}$$

The temperature is updated based upon these values of the variables,

$$\theta_{n+1} = \theta_n + \dot{\theta}\Delta t \tag{4.8.26}$$

where

$$\dot{\theta} = \frac{.95}{\rho C_v} (\boldsymbol{\sigma} : \mathbf{d}^{pl}) \tag{4.8.27}$$

or $\dot{\theta}$ can be computed using Equation (4.8.6).

In the case of uniaxial stress, the yield function takes the form

$$\phi = |\sigma - \alpha| - \chi(\dot{\epsilon}, \theta), \tag{4.8.28}$$

where,

$$\chi(\dot{\epsilon}, \theta) = Y(\theta) + V(\theta)\, \sinh^{-1}\left[\frac{|\sigma - \alpha|}{r(\theta)}\right]. \tag{4.8.29}$$

and σ is the true uniaxial (Cauchy) stress and $\dot{\epsilon}$ is the true stretching or strain rate. The function χ is the same as in Equation (4.8.9), except that $|\xi|$ is replaced by the uniaxial value, $|\sigma - \alpha|$ as in the yield surface.

This formulation allows a general fit for many metals over many decades of strain rate and a large temperature range. The temperature dependence in Equation (4.8.28) would, in general, take different forms depending upon the temperature range considered. However, this does not complicate the model or the determination of parameters. The parameter $V(\theta)$ determines the slope of the yield stress versus logarithm of strain rate at the higher strain rates as shown in Figure 4.8.1. This slope increases sharply with increasing temperature consistent with the observed increased rate sensitivity with temperature of most metals.

The rate independent yield stress is given by $Y(\theta)$, while $r(\theta)$ determines the strain rate at which the rate independent limit is reached as shown in Figure 4.8.2. The back-extrapolated yield stress is determined from a series of tension or compression tests at various temperatures. It is important to use the back-extrapolated yield stress since we are neglecting the "knee" in the post yield regime and desire an accurate representation at strains larger than 2 or 3 percent. The final parameters in the function $\chi(\dot{\epsilon},\theta)$ can then be determined with a nonlinear least squares analysis utilizing the data at all temperatures simultaneously. If the form of the functions $r(\theta)$, $Y(\theta)$, and $V(\theta)$ is not known, these can be determined by fitting Equation (4.8.28) at various temperatures, giving values of these parameters at various temperatures. From this data, the best form for $r(\theta)$, $Y(\theta)$, and $V(\theta)$ can be determined.

The dynamic recovery term, $h(|\sigma - \alpha|,\theta)$, is defined as,

$$h(|\sigma - \alpha|,\theta) = \hat{h}(\theta)\ r(\theta)\ \sinh\left(\frac{|\sigma - \alpha| - Y(\theta)}{V(\theta)}\right) . \qquad (4.8.30)$$

The form for this equation was chosen to give an accurate description of large strain compression behavior at different strain rates as well as to simplify the system of equations during loading. This form predicts that the stress reaches a constant value in compression or tension independent of the strain rate. While this is not precisely true for all metals due to the influence of micromechanisms which are not considered, it is a reasonable assumption for stainless steel and is easily modified if the experimental

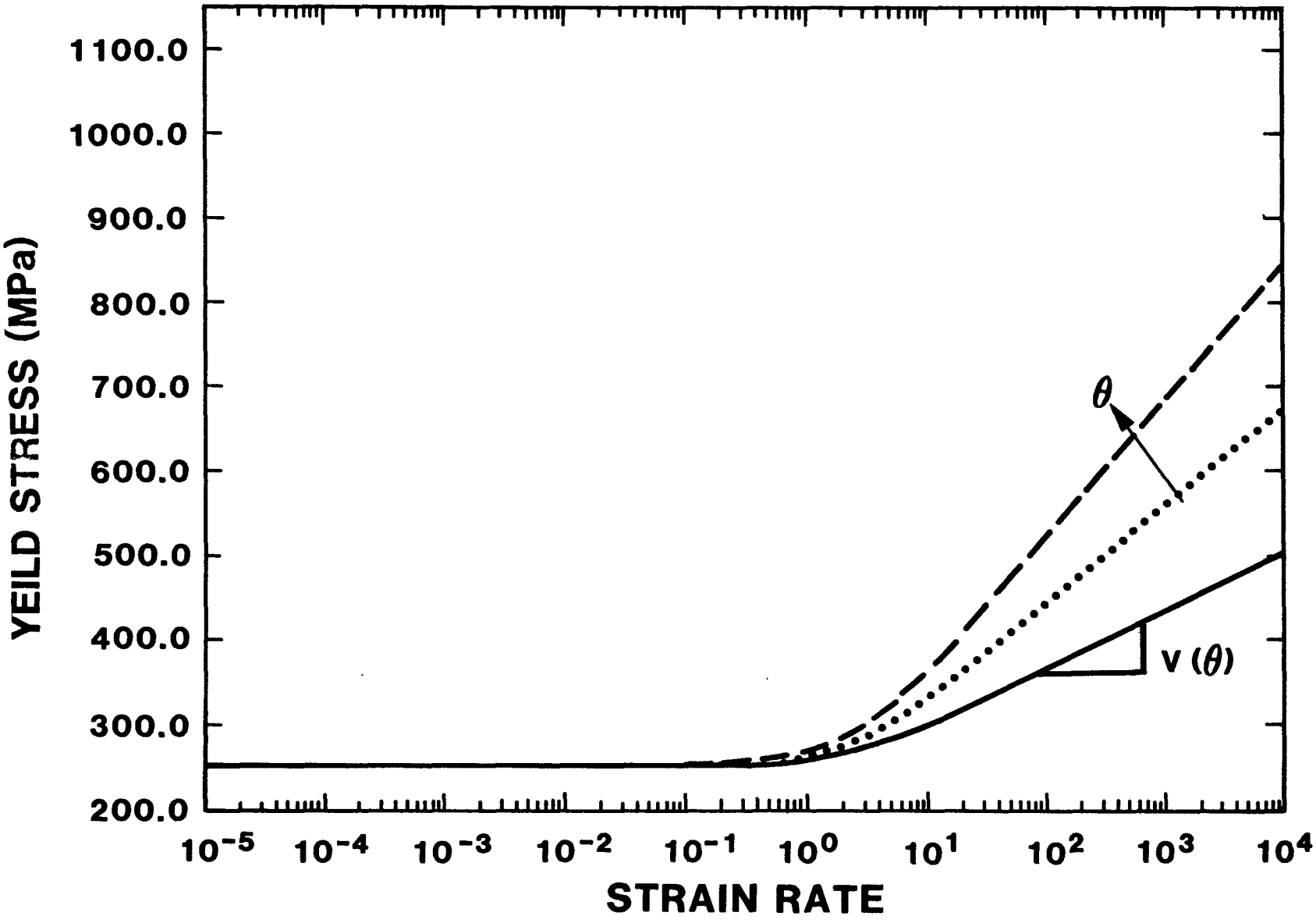


Figure 4.8.1. Effect of Increasing Temperature on the Function V(θ)

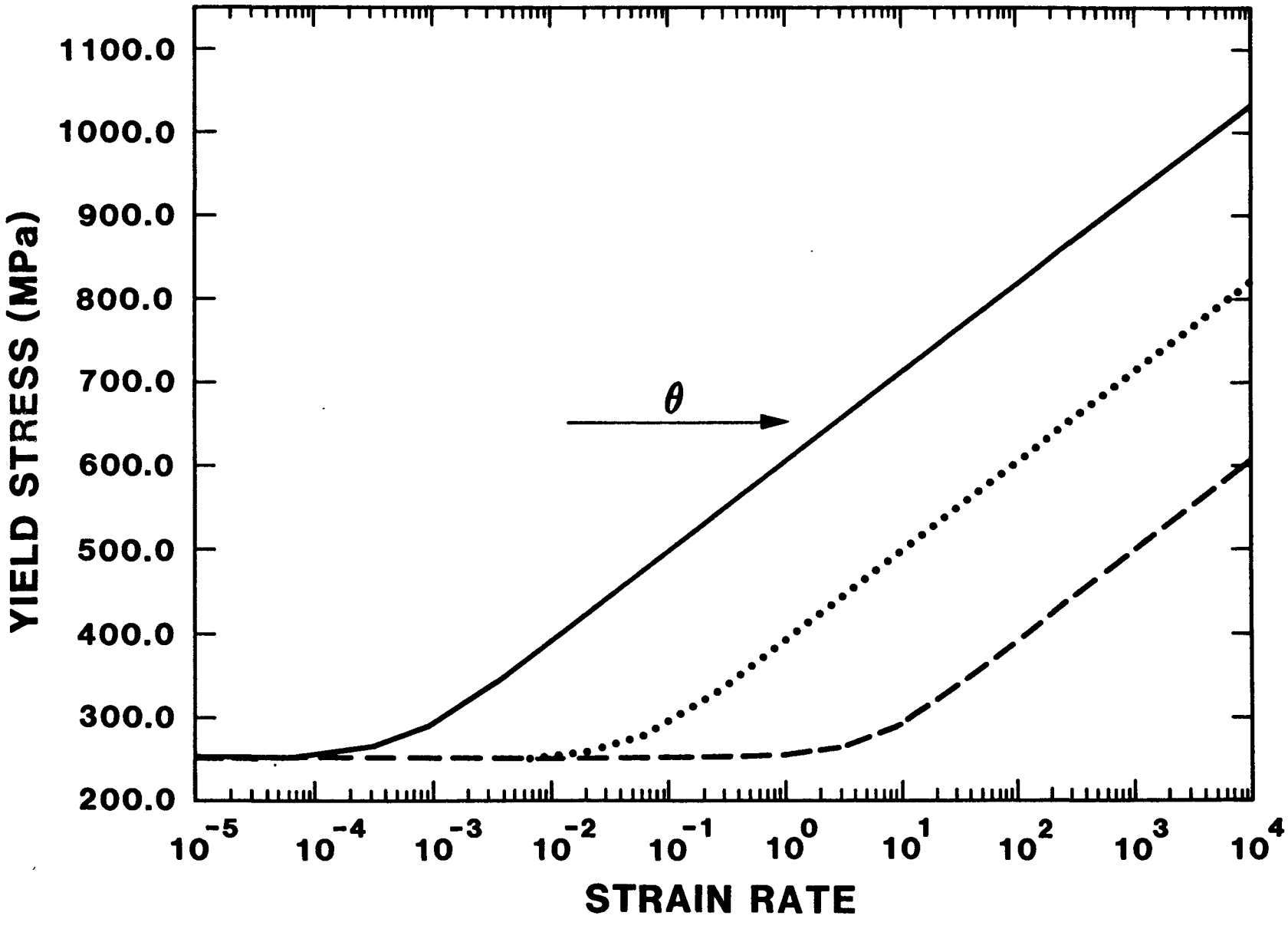


Figure 4.8.2. Effect on the Yield Stress of Increasing the Function r(θ)

results require otherwise. During loading, the dynamic recovery function reduces to,

$$h(|\sigma - \alpha|,\theta) = \hat{h}(\theta)\dot{\epsilon} \ . \tag{4.8.31}$$

Now consider the case of a tensile or compressive test conducted at constant true strain rate and at strain rates small enough to approximate isothermal conditions. For this case, the system of constitutive equations can be integrated in closed form to yield,

$$\sigma = \chi + \sqrt{\frac{\kappa\dot{\epsilon}}{h\dot{\epsilon} + g}} \tanh \sqrt{\frac{\kappa[h\dot{\epsilon} + g]}{\dot{\epsilon}}}\, \epsilon \ . \tag{4.8.32}$$

This equation can be examined to determine the differences between dynamic and static recovery in the case of uniaxial stress at a constant true strain rate. Notice that after an initial strain rate dependent yield, the stress is predicted to approach the asymptote,

$$\sigma = \chi + \sqrt{\frac{\kappa\dot{\epsilon}}{h\dot{\epsilon} + g}} \ , \tag{4.8.33}$$

At high strain rates this reduces to,

$$\sigma \simeq \chi + \sqrt{\frac{\kappa}{\hat{h}}} \ . \tag{4.8.34}$$

Under these conditions the model predicts that the steady state value of the stress occurs at the same value of strain, hence at high strain rates the strain rate dependence of the stress is entirely due to the rate dependence of the yield function χ. This assumption is difficult to check since at the strain rates where this occurs, the tests are not isothermal due to the temperature change from the plastic work. At small strain rates, Equation (4.8.32) reduces to

$$\sigma \simeq \chi + \sqrt{\frac{\kappa\dot{\epsilon}}{g}} \ . \tag{4.8.35}$$

The parameters $k(\theta)$, $g(\theta)$, and $h(\theta)$ can now be determined from Equation (4.8.31). This is accomplished by consideration of tensile or compressive tests performed at strain rates small enough for the isothermal assumption to be reasonably approximated. Then by performing these tests at different temperatures, the parameters $k(\theta)$, $g(\theta)$, and $h(\theta)$ are determined from a nonlinear least squares analysis. The parameter β is determined by considering reverse loading data exactly as in the case for the rate independent, combined hardening model. For $\beta = 0$, the hardening is kinematic, while for $\beta = 1$, the hardening is fully isotropic. Axial stress data in a large strain torsion test can also be used to determine this parameter. For metals which exhibit an axial stress of approximately one-third the shear stress, the response is best modeled by choosing $\beta = 0$. If the axial stress is approximately two orders of magnitude smaller than the shear stress, $\beta = 1$ is a more appropriate choice. Users interested in this effect are referred to reference [33] for details.

The temperature dependent parameters, $Y(\theta)$, $V(\theta)$, $g(\theta)$, $k(\theta)$, $h(\theta)$, and $r(\theta)$ are chosen for this study as,

$$V(\theta) = C_1 \exp\left(-\frac{C_2}{\theta}\right),$$

$$Y(\theta) = C_3 + \frac{C_4}{\theta},$$

$$r(\theta) = C_5 \exp\left(-\frac{C_6}{\theta}\right),$$

$$h(\theta) = C_7 \exp\left(-\frac{C_8}{\theta}\right),$$

$$k(\theta) = C_9 \exp\left(-\frac{C_{10}}{\theta}\right),$$

$$g(\theta) = C_{11} \exp\left(-\frac{C_{12}}{\theta}\right),$$

It is important to note that the form of these temperature dependent parameters can be as complex as necessary to accurately predict behavior over large temperature ranges. It is not the number of material parameters,

but rather the number of independent tests required to determine the parameter which is important.

The elastic plastic temperature dependent material uses seven internal state variables:

EQPS - equivalent plastic strain
TEMP - temperature
RADIUS - current radius of the yield surface
ALPHA11 - 1,1 component of the backstress
ALPHA22 - 2,2 component of the backstress
ALPHA33 - 3,3 component of the backstress
ALPHA12 - 1,2 component of the backstress

The PROP array for this material contains the following entries:

PROP(1) - Young's Modulus, E
PROP(2) - Poisson's Ratio, ν
PROP(3) - C1
PROP(4) - C2
PROP(5) - C3
PROP(6) - C4
PROP(7) - C5
PROP(8) - C6
PROP(9) - C7
PROP(10) - C8
PROP(11) - C9
PROP(12) - C10
PROP(13) - C11
PROP(14) - C12
PROP(15) - β
PROP(16) - ρC_v
PROP(17) - initial temperature
*PROP(18) - 2μ
*PROP(19) - 3μ
*PROP(20) - λ

TABLE 4.8.1. VALUES OF PARAMETERS FOR 21-6-9 SS

$E = 207$ GPa

$\nu = .3$

$\beta = 0$

$C_V = 4.60 \times 10^2$ [J/kg·K]

$\rho = 7.83 \times 10^3$ [Kg/m^3]

$C1 = 5.58 \times 10^1$ [MPa]

$C2 = 8.67 \times 10^1$ [K]

$C3 = 2.448 \times 10^1$ [MPa]

$C4 = 1.07 \times 10^5$ [MPa*K]

$C5 = 7.28 \times 10^2$ [s^{-1}]

$C6 = 2.44 \times 10^3$ [K]

$C7 = 2.47 \times 10^{-3}$ [1/MPa]

$C8 = 1.1 \times 10^2$ [K]

$C9 = 1.81 \times 10^3$ [MPa]

$C10 = 5.23 \times 10^1$ [K]

$C11 = 0.$ [1/MPa·s]

$C12 = 1.$ [K]

4.9 Elastic Plastic Hydrodynamic Material

The elastic plastic hydrodynamic material model is a combination of the elastic plastic combined hardening model described in Section 4.2 and the purely hydrodynamic material model described in Section 4.7. In this material model, we uncouple the volumetric and deviatoric response. The volumetric response is determined using one of the equations of state defined in Chapter 5, and the deviatoric response is determined using the equations of Section 4.2.

First, we calculate the deviatoric response. This is accomplished in a manner almost identical to Section 4.2.5. We calculate a trial deviatoric stress

$$S^{TR} = S_n + 2\mu \dot{\mathbf{e}} \tag{4.9.1}$$

where $\dot{\mathbf{e}}$ is the deviatoric part of the strain rate, $\mathbf{d}$, and 2μ is the usual Lame constant. We then proceed exactly as in Section 4.2.5 with an incremental consistency condition and determine the increment in equivalent plastic strain, update the radius of the yield surface, and update the backstress. The same radial return correction is applied to the DEVIATORIC part of the stress tensor.

We then update the energy to include the increment due to the deviatoric energy contribution in Equation (5.1.9).

Next we call the appropriate equation of state to calculate the new pressure at the end of the time increment. Once this is known, we determine the total stress by

$$\sigma_{n+1} = S_{n+1} - p\delta \tag{4.9.2}$$

The elastic plastic hydrodynamic material model uses six internal state variables:

EQPS – equivalent plastic strain

RADIUS – current radius of yield surface

ALPHA11 – 1,1 component of backstress in unrotated configuration

ALPHA22 – 2,2 component of backstress in unrotated configuration

ALPHA33 – 3,3 component of backstress in unrotated configuration

ALPHA12 – 1,2 component of backstress in unrotated configuration

The PROP array for this material contains the following entries:

PROP(1) – Young's Modulus, E

PROP(2) – Poisson's Ratio, ν

PROP(3) – Yield Stress, σ_{yd}

PROP(4) – Hardening Modulus, H

PROP(5) – β

PROP(6) – pressure cutoff

*PROP(7) – 2μ

*PROP(8) – $1/(2\mu*(1 + H'/3\mu))$ (Note: $H' = H/(1 - E/H)$

*PROP(9) – λ

*PROP(10) – $2\beta \cdot H'/3$

*PROP(11) – $2(1 - \beta)H'/3$

5.0 EQUATIONS OF STATE

The discussion of the hydrodynamic equations of state incorporated in PRONTO follows closely the development of the theory found in WONDY [3] and TOODY [4].

5.1 Introduction

The equation for conservation of energy equates the increase in internal energy per unit volume to the rate at which work is being done by the stresses and the rate at which heat is being added. In the absence of heat conduction

$$\dot{E}_v = \rho \frac{\partial E_m}{\partial t} = (p - q) \frac{1}{\rho} \frac{\partial \rho}{\partial t} + (\boldsymbol{\sigma} - p\boldsymbol{\delta}):\dot{\mathbf{e}} + \rho\dot{Q} \ . \qquad (5.1.1)$$

We note that in Equation (5.1.1), p is the pressure measured as **POSITIVE IN COMPRESSION**, and q is the pressure due to the bulk viscosity, from Equation (3.7.1), which is **NEGATIVE IN COMPRESSION**. Also in Equation (5.1.1), E_v is the energy per unit volume, E_m is the energy per unit mass, and $\dot{Q}$ is the heat rate per unit mass.

The continuity equation can be written as

$$\frac{1}{\rho} \frac{\partial \rho}{\partial t} = -\text{tr } \mathbf{d} = -d_{kk} \ . \qquad (5.1.2)$$

The deviatoric part of the strain rate is

$$\dot{\mathbf{e}} = \mathbf{d} - \frac{1}{3} \text{tr } \mathbf{d}\ \boldsymbol{\delta} \ . \qquad (5.1.3)$$

and the pressure is given by

$$p = -\frac{1}{3} \text{tr } \boldsymbol{\sigma} \ . \qquad (5.1.4)$$

Rewriting Equation (5.1.1) in terms of the pressure and the deviatoric part of the stresses, S,

$$\dot{E}_v = (q - p)\, d_{kk} + \mathbf{S}:\dot{\mathbf{e}} + \rho\dot{Q} \ . \qquad (5.1.5)$$

An equation of state is assumed for the pressure as a function of density and energy per unit mass

$$p = f(\rho, E_m) \ . \qquad (5.1.6)$$

In PRONTO we use equations of state linear in internal energy of the form

$$p = f_1(\rho) + f_2(\rho)\, E_m \ . \qquad (5.1.7)$$

We find it convenient in the numerical implementation to work with energy per unit volume instead of energy per unit mass and rewrite Equation (5.1.7) as

$$p = f_1(\rho) + f_3(\rho)\, E_v \ . \qquad (5.1.8)$$

where we have defined a new function, $f_3(\rho) = f_2(\rho)/\rho$.

Assuming there are no heat sources and the strain rates are constant over the step, we can integrate Equation (5.1.5) to obtain the following discrete form of the energy equation:

$$E_v^{t+\Delta t} = E_v^t + \frac{\Delta t}{2}\,(q^t + q^{t+\Delta t} - p^t - p^{t+\Delta t})\, d_{kk} + \frac{\Delta t}{2}\,(\mathbf{S}^t + \mathbf{S}^{t+\Delta t}):\dot{\mathbf{e}} \ . \qquad (5.1.9)$$

Equations (5.1.9) and (5.1.8) represent two linear equations in two unknowns: $E_v^{t+\Delta t}$ and $p^{t+\Delta t}$.

Defining E_v^* by

$$E_v^* = E_v^t + \frac{\Delta t}{2}\,(q^t + q^{t+\Delta t} - p^t)\, d_{kk} + \frac{\Delta t}{2}\,(\mathbf{S}^t + \mathbf{S}^{t+\Delta t}):\dot{\mathbf{e}} \qquad (5.1.10)$$

we rewrite Equation (5.1.9) as

$$E_v^{t+\Delta t} = E_v^* - \frac{\Delta t}{2}\, p^{t+\Delta t}\, d_{kk} \ . \qquad (5.1.11)$$

If we have a completely hydrodynamic equation of state (see Section 4.5), there are no deviatoric terms in Equation (5.1.10) (i.e., $\mathbf{S}$ and $\dot{\mathbf{e}}$ are both zero). For the elastic plastic hydrodynamic material (see Section 4.9), the deviatoric and volumetric response are uncoupled. We first determine the deviatoric response and calculate the deviatoric strain energy in Equation (5.1.10). Then we proceed with the equation of state calculation.

The bulk viscosity pressure as defined in Equation (3.7.5) contains a linear and a quadratic part. Careful inspection of Equation (3.7.5) along with the definition of the stable time increment given by Equation (3.7.2) shows that the quadratic part of q is independent of the effective dilitational modulus, while the linear part is not. At the time we must calculate E_v^* in Equation (5.1.10), we do not yet know the effective moduli for the time step since it depends on the new pressure. To avoid the need to interate to solve Equations (5.1.10) and (5.1.11), we do not include the energy due to the linear term in the calculation.

By substituting Equation (5.1.11) into (5.1.8), we can solve for the new pressure;

$$p^{t+\Delta t} = \frac{f_1(\rho) + f_3(\rho)\, E_V^*}{1 + f_3(\rho)\, \frac{\Delta t}{2}\, d_{kk}} \tag{5.1.12}$$

After calculating the new pressure using Equation (5.1.12), the energy can be updated using Equation (5.1.11).

5.2 Mie–Gruneisen Type Equations of State

The designation, Mie–Gruneisen equation of state, refers to any equation of state which is linear in energy. The most general form is

$$p - p_H = \Gamma\rho\, (E_m - E_H) \tag{5.2.1}$$

where p_H and E_H are pressure and energy per unit mass along some reference path and are functions of density only. The Gruneisen ratio, Γ, is also a function of density only. The Hugoniot reference pressure $p_H(\rho)$ is generally defined from fits to experimental data.

The Hugoniot energy is related to the Hugoniot pressure by

$$E_H = \frac{p_H \eta}{2\rho_o} \tag{5.2.2}$$

where

$$\eta = 1 - \rho_o/\rho \ . \tag{5.2.3}$$

The Gruneisen ratio is usually approximated as

$$\Gamma(\rho) = \Gamma_o(1 + h_1\eta + h_2\eta^2 + ...) \ . \tag{5.2.4}$$

Using Equation (5.2.2) in Equation (5.2.1) leads to

$$p = p_H \left[1 + \frac{\Gamma}{2}\left(\frac{\rho}{\rho_o} - 1\right)\right] + \Gamma\rho E_m \ . \tag{5.2.5}$$

Equation (5.2.5) has the form of Equation (5.1.7)

$$p = f_1(\rho) + f_2(\rho)\ E_m \tag{5.2.6}$$

where

$$f_1(\rho) = p_H \left(1 + \frac{\Gamma\mu}{2}\right) \tag{5.2.7}$$

$$f_2(\rho) = \Gamma\rho \tag{5.2.8}$$

and

$$\mu = \left(\frac{\rho}{\rho_o} - 1\right) = \frac{\rho}{\rho_o}\,\eta \ . \tag{5.2.9}$$

The most common form for Equation (5.2.4) is to use $h_1 = -1$ and all other $h_i = 0$ which gives

$$\Gamma = \Gamma_o \frac{\rho_o}{\rho} \ . \qquad (5.2.10)$$

ALL the Mie-Gruneisen equation of states in PRONTO will use the form given by Equation (5.2.10).

5.2.1 Linear Us-Up Hugoniot Form

A common fit to Hugoniot data is given by

$$p_H = \frac{\rho_o c_o^2 \eta}{(1 - s\eta)^2} \qquad (5.2.11)$$

where c_o and s come from the linear shock velocity-partical velocity U_s-U_p fit

$$U_s = c_o + s \cdot U_p \ . \qquad (5.2.12)$$

Equation (5.2.11) follows directly from the relations

$$p_H = U_p \cdot \rho_o \cdot U_s \qquad (5.2.13)$$

and

$$\eta = U_p / U_s \ . \qquad (5.2.14)$$

We see that there is a limiting compression given by the denominator of Equation (5.2.11)

$$\eta_{lim} = 1/s \qquad (5.2.15)$$

or

$$\rho_{lim} = \frac{s\rho_o}{s - 1} \ . \qquad (5.2.16)$$

Also, at $\eta = -1/s$ there is a tensile minimum and thereafter, negative sound speeds are calculated for the material. Since Equation (5.2.11) is intended for use in compression, caution is advised if the model is used in response regimes where large tensions are expected.

For this form of the equation of state we see that

$$f_1(\rho) = \frac{\rho_0 c_0^2 \eta}{(1 - s\eta)^2} \left(1 - \frac{\Gamma\mu}{2}\right) \tag{5.2.17}$$

and

$$f_3(\rho) = \frac{\Gamma_0 \rho_0}{\rho} = \Gamma \ . \tag{5.2.18}$$

5.2.2 Power Series Hugoniot Form

Another common form for the Hugoniot is to express the reference pressure, p_H, in a power series in η,

$$p_H = K_0\eta(1 + K_1\eta + K_2\eta^2 + \ldots .) \ . \tag{5.2.19}$$

In order to match $\frac{dp_H}{d\eta}$ at $\eta = 0$ it is necessary that

$$K_0 = \rho_0 c_0^2 \tag{5.2.20}$$

where K_0 is the adiabatic bulk modulus at zero pressure and room temperature and c_0 is the bulk sound speed.

For this Hugoniot form, the equation of state is defined by

$$f_1(\rho) = K_0\eta(1 + K_1\eta + K_2\eta^2) \left(1 - \frac{\Gamma\mu}{2}\right) \tag{5.2.21}$$

and

$$f_3(\rho) = \frac{\Gamma_0 \rho_0}{\rho} = \Gamma \tag{5.2.22}$$

where we have restricted ourselves to using only three terms in the polynomial in Equation (5.2.19).

5.2.3 Ideal Gas Equation of State

The ideal gas equation of state is given by

$$p = (\gamma - 1)\ \rho E_m \ . \tag{5.2.23}$$

where γ is a material parameter.

Hence, we see that

$$f_1(\rho) = 0 \tag{5.2.24}$$

and

$$f_3(\rho) = (\gamma - 1) \ . \tag{5.2.25}$$

The initial sound speed in the gas, c_o, must be defined by the user. The initial pressure and specific internal energy per unit mass are

$$p_o = \frac{\rho_o c_o^2}{\gamma} \tag{5.2.26}$$

$$E_{m_o} = \frac{c_o^2}{\gamma(\gamma - 1)} \ . \tag{5.2.27}$$

The initial pressure and energy per unit volume

$$E_o = \rho E_{m_o} = \frac{\rho c_o^2}{\gamma(\gamma - 1)} \tag{5.2.28}$$

are initialized inside the code.

5.2.4 JWL High Explosive Equation of State

The Jones-Wilkins-Lee or JWL equation of state [35] provides the pressure generated by the release of chemical energy in an explosive. In PRONTO it is implemented in a form which is usually referred to as a programmed burn. A programmed burn means that the reaction and initiation of the explosive is not determined by the shock in the material, rather the initiation time is determined by a Huygens construction using the detonation wave speed and the distance of the material point from the detonation point(s).

The JWL equation of state is generally written as

$$p = A\left(1 - \frac{\omega\rho}{R_1\rho_o}\right)e^{-R_1\frac{\rho_o}{\rho}} + B\left(1 - \frac{\omega\rho}{R_2\rho_o}\right)e^{-R_2\frac{\rho_o}{\rho}} + \frac{\omega\rho^2}{\rho_o}E_{m_o} \qquad (5.2.29)$$

where A, B, R_1, R_2, ω, and E_{m_o} are material constants. Note that Equation (5.2.29) is written in terms of energy per mass which is the usual form found in the literature. Again, we chose to write our equations of state in terms of energy per unit volume which results in

$$f_1(\rho) = A\left(1 - \frac{\omega\rho}{R_1\rho_o}\right)e^{-R_1\frac{\rho_o}{\rho}} + B\left(1 - \frac{\omega\rho}{R_2\rho_o}\right)e^{-R_2\frac{\rho_o}{\rho}} \qquad (5.2.30)$$

and

$$f_3(\rho) = \frac{\omega\rho}{\rho_o}\ . \qquad (5.2.31)$$

The programmed burn requires the initial calculation of the arrival of the detonation wave at a material point. If there is only one detonation point denoted by $\mathbf{x}_d$, and if the location of the material point is denoted by $\mathbf{x}_n$, then the detonation time is determined by

$$t_d = |\mathbf{x}_d - \mathbf{x}_n|/c_d \qquad (5.2.32)$$

where c_d is the detonation wave speed (a material property supplied by the user) and the symbol $|\cdot|$ indicates the Euclidian norm of a vector. Clearly, if there are multiple detonation points, then Equation (5.2.32) must be applied for each material point for each detonation point and the arrival time is the minimum.

In order to spread the burn wave over several elements, a burn fraction F is computed as

$$F = \min \left[1, \frac{(t - t_d)\, c_d}{B_s \cdot \ell} \right] \tag{5.2.33}$$

where B_s is a constant which controls the width of the burn wave (defaulted to 2.5 in the code) and ℓ is the characteristic length of the element which is calculated internally in the code as the square root of the area of the element. If the time t is less than t_d, the pressure is zero in the explosive. Otherwise, the pressure is given by

$$p = F\,[f_1(\rho) + f_2(\rho) E_v] \tag{5.2.34}$$

When $t < t_d$ the detonation wave speed is used as the sound speed in the material. After the detonation wave has arrived, the sound speed is calculated internally from the hypoelastic stress rates and strain rates just as for all other materials.

This form of the equation of state requires that the internal energy per unit volume be initialized to account for the chemical energy in the explosive. Parameters for a wide variety of explosives have been tabulated by Dobratz [36].

6.0 CONTACT SURFACES

PRONTO currently supports two types of contact surface boundary conditions: a deformable surface against a rigid plane, and two distinct deformable surfaces against each other. The first option requires a far more simple procedure since the constraints on each node are completely uncoupled.

Contact is treated as a kinematic constraint by PRONTO. This means that the final product of the contact algorithm is to modify the accelerations of the nodes along this surface such that the kinematic constraints are satisfied.

PRONTO supports friction for both contact surface options. Either a simple Coulomb friction model or a velocity dependent friction model may be selected.

6.1 Deformable-to-Rigid Surface Contact

The rigid surface option in PRONTO imposes the kinematic constraints of an unyielding plane on a user specified surface of the deformable body. The plane is defined by a point $\bar{\mathbf{x}}$ and the outward unit normal $\mathbf{n}$. The deformable surface can be treated as simply a set of unique nodes. The primary kinematic condition is that the deformable nodes may not penetrate the rigid plane. In addition, the motion of deformable nodes along the plane may be restricted subject to a velocity dependent friction law.

6.1.1 Normal Constraint

We begin by integrating the motion of the deformable nodes without regard to the kinematic constraints required by the rigid surface. For each node, we calculate a predicted kinematic state as follows:

$$\hat{\mathbf{a}} = \mathbf{f} \,/\, m \tag{6.1.1}$$

$$\hat{\mathbf{v}} = \mathbf{v} + \Delta t\, \hat{\mathbf{a}} \tag{6.1.2}$$

$$\hat{\mathbf{x}} = \mathbf{x} + \Delta t\, \hat{\mathbf{v}} \tag{6.1.3}$$

In the above equations, $\mathbf{f}$ is the residual force vector (sum of external forces minus sum of internal forces), m is the nodal mass, $\mathbf{v}$ is the current velocity, $\mathbf{x}$ is the current position, and Δt is the time increment. The predicted kinematic quantities are denoted by a superposed hat. Note that the predicted accelerations and velocities are never stored in a global array.

We now calculate the depth of penetration of each node into the plane as shown below. This depth is zero for nodes which are not in contact.

$$\delta = \max \left(\mathbf{n} \cdot (\tilde{\mathbf{x}} - \hat{\mathbf{x}}),\ 0 \right) \tag{6.1.4}$$

The magnitude of the force which must be applied to enforce the kinematic constraint, i.e., which will cancel the penetration, is given by

$$f_n = \delta\, m \,/\, \Delta t^2 \tag{6.1.5}$$

This force must be applied in the direction of $\mathbf{n}$. Applying this correction to Equation (6.1.1) and eliminating the nodal mass, we can express the new acceleration in the absence of friction as

$$a_n = \delta \,/\, \Delta t^2 \tag{6.1.6}$$

$$\mathbf{a} = \hat{\mathbf{a}} + a_n\, \mathbf{n} \tag{6.1.7}$$

6.1.2 Friction

Friction resists tangential motion of deformable nodes contacting the rigid plane. The unit tangent vector is orthogonal to the outward normal and, therefore, is expressed by

$$\mathbf{s} = \begin{Bmatrix} -n_y \\ n_x \end{Bmatrix} \tag{6.1.8}$$

The tangential component of the predicted velocity of a node is computed as follows:

$$v_s = \mathbf{s} \cdot \hat{\mathbf{v}} \tag{6.1.9}$$

The force which must be applied to cancel the tangential velocity of a node is then given by

$$f_s = - m / \Delta t \; v_s \tag{6.1.10}$$

where the minus sign above reflects that this force would be applied in the direction of $\mathbf{s}$, but opposing the motion.

PRONTO currently supports three options for friction: no friction, Coulomb friction with a constant coefficient of friction, or the velocity dependent friction law found in HONDO II [6]. The coefficient of friction can be expressed by

$$\mu = \mu_\infty + (\mu_0 - \mu_\infty) \; e^{-\gamma v_s} \tag{6.1.11}$$

where μ_0 and μ_∞ are the low and high velocity friction coefficients, respectively, and γ is a decay constant. Clearly, if γ equals zero, the coefficient of friction is the constant μ_0. Futhermore, if μ_0 is also equal to zero, the surface will be frictionless.

The magnitude of the tangential force exerted by the plane on a node cannot exceed the maximum friction force. This constraint is expressed as

$$f_f = \frac{f_s}{|f_s|} \min(\mu f_n \; , \; |f_s|) \tag{6.1.12}$$

Substituting Equations (6.1.5), (6.1.6), and (6.1.10) into the above, then eliminating the nodal mass yields

$$a_s = - \frac{v_s}{|v_s|} \min \left(\mu a_n \; , \; |v_s| \; / \; \Delta t \right) \tag{6.1.13}$$

The total acceleration of the node is then given by

$$\mathbf{a} = \hat{\mathbf{a}} + a_n \mathbf{n} + a_s \mathbf{s} \qquad (6.1.14)$$

6.2 Deformable-to-Deformable Surface Contact

The fundamental condition which must be satisfied between two contact surfaces is that one surface may not penetrate into the other. The algorithmic challenge is to find the set of nodal forces which will maintain kinematic compliance. The classic difficulty of contact algorithms is that elaborate and exhaustive schemes to maintain strict compliance are prohibitively expensive to implement and to execute. Furthermore, because the surface is discretized for the finite element method, it is virtually impossible to pose an algorithm which always yields unique and meaningful results.

The contact algorithm in PRONTO, as in any other code, represents a compromise between robustness and efficiency. Our algorithm is designed to handle very large deformations and moderately high impact velocities. The sample problems in Chapter 9 illustrate a fairly representative, but by no means exhaustive, range of applicability of the PRONTO contact surfaces.

PRONTO uses a partitioned kinematic approach to contact. The partitioning can be adjusted to give a strict master-slave treatment or to balance the master-slave relationship between the two surfaces. In any case, the constraint forces conserve momentum.

The contact algorithm is performed in two passes; first with one surface as the master, then with the other surface as the master. One of these passes will be skipped if a strict master-slave treatment is requested. The following sections described just one such pass.

6.2.1 Surface Topology

A surface in PRONTO must be continuous and simply connected. It may be either an open "string" or a closed "loop". The "inside" of the surface to PRONTO is where the material lies; it is assumed that there is no material on the "outside" of the surface. In order for a surface node to form a valid connection, the two adjoining segments must have material on the same side. By convention, PRONTO orders components of a surface such that the material is to the left and the outward normal is to the right as one progresses along the surface.

Figure 6.2.1 illustrates valid and invalid surface topologies. Frames (a) and (b) show valid open and closed surfaces, respectively. Material is represented by shading and an arrow indicates the ordering direction for each surface. Frames (c) through (d) show examples of surfaces which are invalid to PRONTO. Clearly, frame (c) is not continuous and frame (d) is multiply connected. Frame (e) shows a case which appears ambiguous. This surface is actually simply connected, but not continuous, since the center point is not a valid connection.

PRONTO carefully checks the topology of contact surfaces during initialization. It will print with an appropriate error message for each of the invalid cases described above. If the surface is found to be valid, PRONTO will build the node list data structure which contains the topology of the surface as described below. Note that for a closed surface the number of nodes and segments are equal (NNODES = NSIDES), while for an open surface the number of nodes is one more than the number of sides (NNODES = NSIDES+1).

The node list data structure, NSNODE(NSIDES+1), contains the list of surface nodes in the order in which they appear along the surface. The first NNODES entries always contain unique nodes; the last entry is identical to the first entry (NSNODE(NSIDES+1) = NSNODE(1)) for a closed surface. The surface is made up of a series of segments; each segment is a straight line defined by an element side. The node list also defines the initial and terminal node of each segment, since the terminal point of one segment is

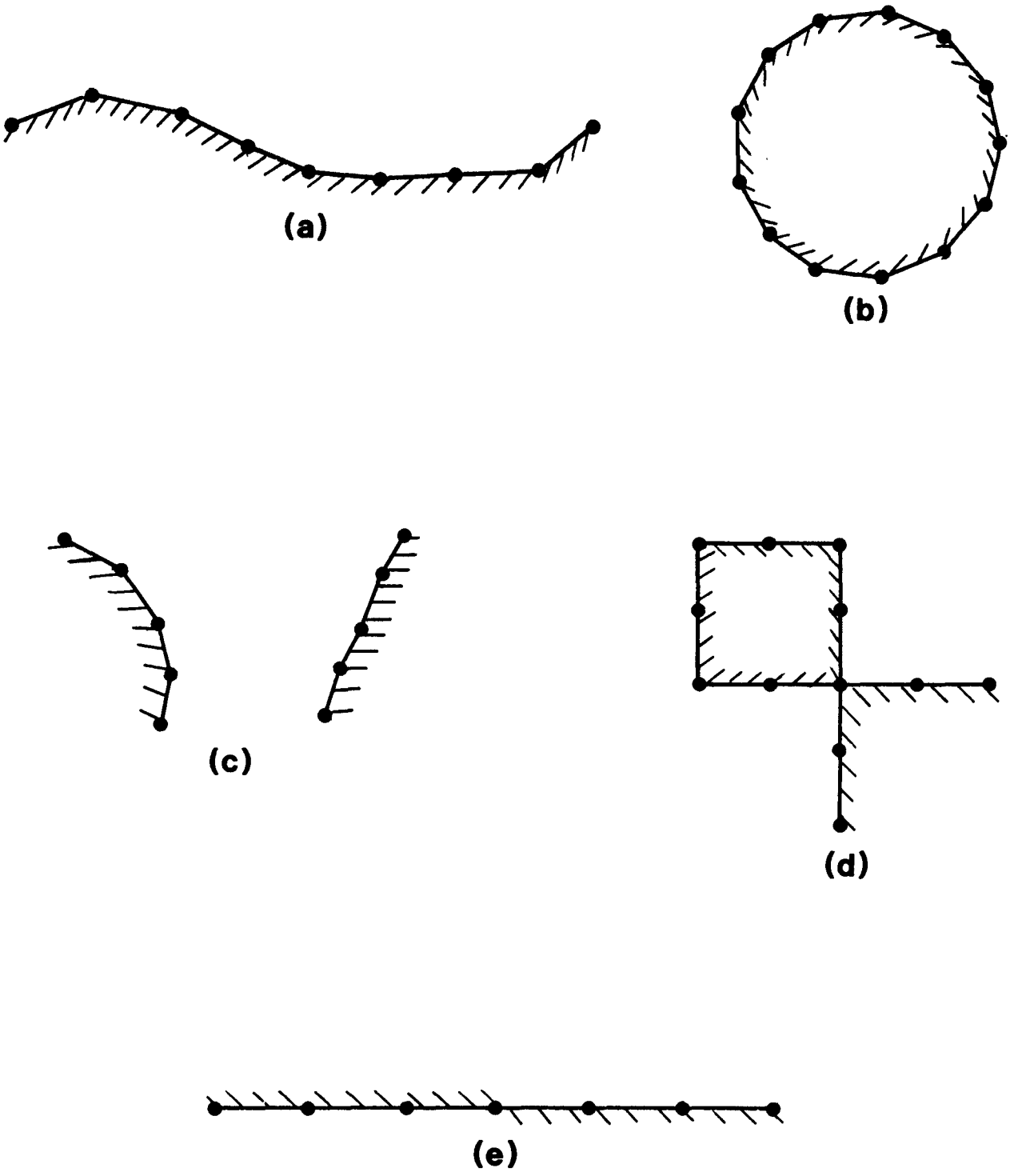


Figure 6.2.1. Valid (a and b) and Invalid (c, d, and e) Surface Topologies for Contact Surfaces

the initial point of the following segment; the segment "N" goes from node NSNODE(N) to node NSNODE(N+1).

6.2.2 Surface Geometry

PRONTO recalculates the geometry of all contact surfaces at each time step. A predicted configuration is computed by integrating the motion without regard to the kinematic constraints required by the contact surfaces. For each node we calculate:

$$\hat{\mathbf{a}} = \mathbf{f} \,/\, m \tag{6.2.1}$$

$$\hat{\mathbf{v}} = \mathbf{v} + \Delta t\, \hat{\mathbf{a}} \tag{6.2.2}$$

$$\hat{\mathbf{x}} = \mathbf{x} + \Delta t\, \hat{\mathbf{v}} \tag{6.2.3}$$

In the above equations, $\mathbf{f}$ is the residual force vector (sum of external forces minus sum of internal forces), m is the nodal mass, $\mathbf{v}$ is the current velocity, $\mathbf{x}$ is the current position, and Δt is the time increment. The predicted kinematic quantities are denoted by a superposed hat. Note that the predicted accelerations and velocities are never stored in a global array.

The outward unit normals of the surface segments are critical to PRONTO's contact algorithm. For convenience of the tracking algorithm (Section 6.2.3), the array which contains the segment normals, P(0:NNODES,2), is structured so that either side attached to a given node can be referenced readily; the precedent segment normal to node "N" is P(N-1,1:2), while the antecedent segment normal is P(N,1:2). The unit normals of the segments (1 through NSIDES) are calculated as follows:

$$\mathbf{N}_I = \begin{Bmatrix} y_{I+1} - y_I \\ x_I - x_{I+1} \end{Bmatrix} \tag{6.2.4}$$

$$\mathbf{n}_I = \mathbf{N}_I \; / \sqrt{\mathbf{N}_I \cdot \mathbf{N}_I} \qquad (6.2.5)$$

The ends of the normal array need to be filled in dependent upon whether the surface is closed or open. For the closed case, we simply copy the last position to the initial position:

$$\mathbf{n}_0 = \mathbf{n}_{NSIDES} \qquad (6.2.6a)$$

For the open case, we construct virtual normals such that right outside corners are formed at the edge nodes:

$$\mathbf{n}_0 = -\mathbf{s}_1 \quad , \quad \mathbf{n}_{NNODES} = \mathbf{s}_{NSIDES} \qquad (6.2.6b)$$

where $\mathbf{s}$ represents the unit tangent of a surface segment. This vector is never stored since it can be readily defined from the unit normal as follows:

$$\mathbf{s} = \begin{Bmatrix} -n_y \\ n_x \end{Bmatrix} \qquad (6.2.7)$$

6.2.3 Surface Tracking

Surface tracking is the process of matching points along one surface to points along its mating surface. For our purposes, it is sufficient to locate the nearest master node to the possible point of contact for each slave node. It is important to understand that in this context the "nearness" of a node is measured in terms of the surficial (along the surface) distance, rather than the spatial (straight line) distance. Once we have determined the nearest master node, all that remains is to determine which of the two master segments the slave node may be contacting. In the next section (6.2.4), we describe how we decide whether the slave node is in contact and which master segment it is contacting.

The tracking algorithm truly governs the cost/benefit of the contact surface capability in PRONTO. The amount of geometric detail that the

tracking algorithm can resolve determines the range of applicability of the contact surfaces. On the other hand, exhaustive checks of every slave node against every master node during every time step will always be prohibitively expensive, and generally unwarranted. The tracking algorithm, therefore, is an area where compromises must be made, but where cleverness will pay off.

The tracking algorithm currently implemented in PRONTO is based on two assumptions regarding the behavior of the contacting surfaces. The first assumption is that the spatially nearest master node is also the surficially nearest master node to the point of contact. This assumption allows us to find the nearest node via simple distance calculations. We initialize the tracking scheme for each slave node S by finding the master node I which has the minimum distance d_I defined by

$$d_I = (\mathbf{x}_I - \mathbf{x}_S)\cdot(\mathbf{x}_I - \mathbf{x}_S) \tag{6.2.8}$$

The second assumption we make to streamline the tracking algorithm is that the nearest master node to a given slave node at one time step will be in the vicinity of the nearest master node at the next time step. This assumption allows us to update the tracking scheme by simply searching for a local minimum in the vicinity of the previously nearest master node. Thus, at each time step, we start at the previously nearest node and search in either the ascending or descending direction along the surface until we find a master node which satisfies:

$$d_{I-1} \geq d_I \leq d_{I+1} \tag{6.2.9}$$

where it is assumed that the search wraps around the node list and

$$d_0 = d_{NNODES} \quad ; \quad d_{NNODES+1} = d_1 \tag{6.2.10}$$

While the tracking algorithm in PRONTO is quite simple, it will solve the majority of contact problems. With varying amounts of user intervention, this algorithm can handle virtually any problem. We shall explain where

limitations of the current algorithm arise and how to circumvent them in the following discussion.

The tracking algorithm may fail when either of the above assumptions is invalidated. The first such condition can only occur near a very sharp corner. Figure 6.2.2 illustrates a case where a master node on the back side of the master surface has the shortest spatial distance to the slave node. The shortest surficial distance to the slave node is clearly to either the corner node or its front side neighbor. The symptom which will occur in this situation is that the slave node will be kicked out of the back side of the master surface and hook there.

The above condition is highly unlikely to occur in any reasonable problem. The are three ways to relieve the symptom for this case. The first, and preferred, method is to refine the mesh at the tip of the master surface (which should be done to achieve decent accuracy anyway). As a rule of thumb, the length of the side segments should always be less than the thickness of the material. The second course of treatment is to set the partition factor (Appendix A, command 27) for the sharp surface so that it acts only as a slave. The final course of treatment would be to divide the master surface at the tip. This must be done with caution since the corner node would then be part of two distinct surfaces, which could lead to a conflict if these two surfaces contact the same master surface.

The second condition which can cause the tracking algorithm to fail is a severely undulating master surface. Figure 6.2.3 shows a situation where a slave node is tracking the master surface on the wrong side of a peak. For this to occur, the master surface must be folding quite rapidly and the slave surface must be moving tangentially to the original master surface. The symptom that occurs in this case is that this slave node will never detect contact. This is also a highly unlikely situation since the tracking information is continuously updated. The remedy for this symptom is quite simple; the calculation should be restarted (Appendix A, commands 8 and 7) after the surface peaks and valleys have formed. PRONTO reinitializes the tracking data when it reads a restart file, which forces an exhaustive search for the nearest master node via Equation (6.2.8).

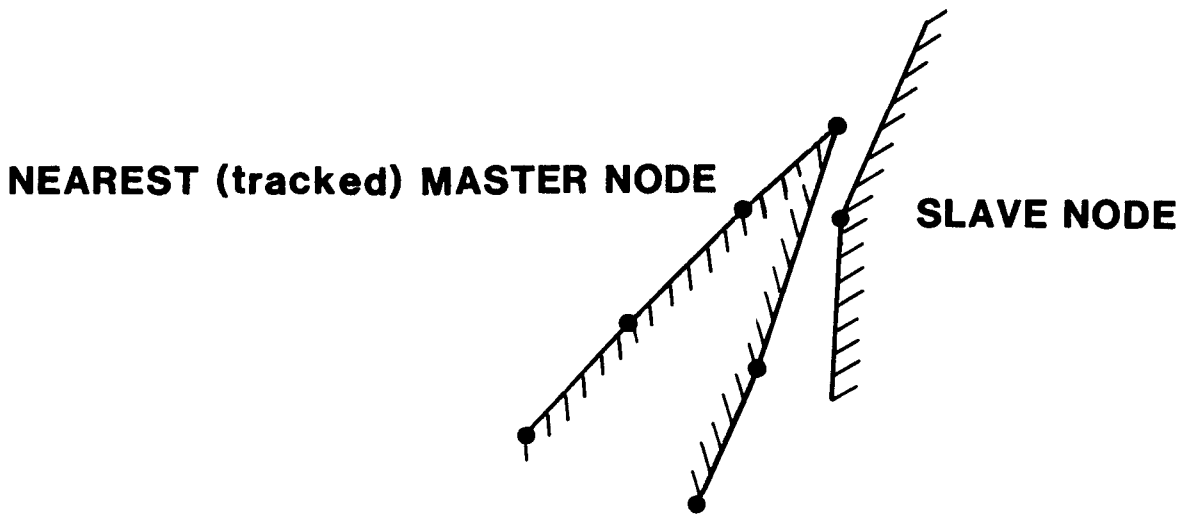


Figure 6.2.2. Case Where a Master Node on the Back Side of the Master Surface has the Shortest Spatial Distance to the Slave Node

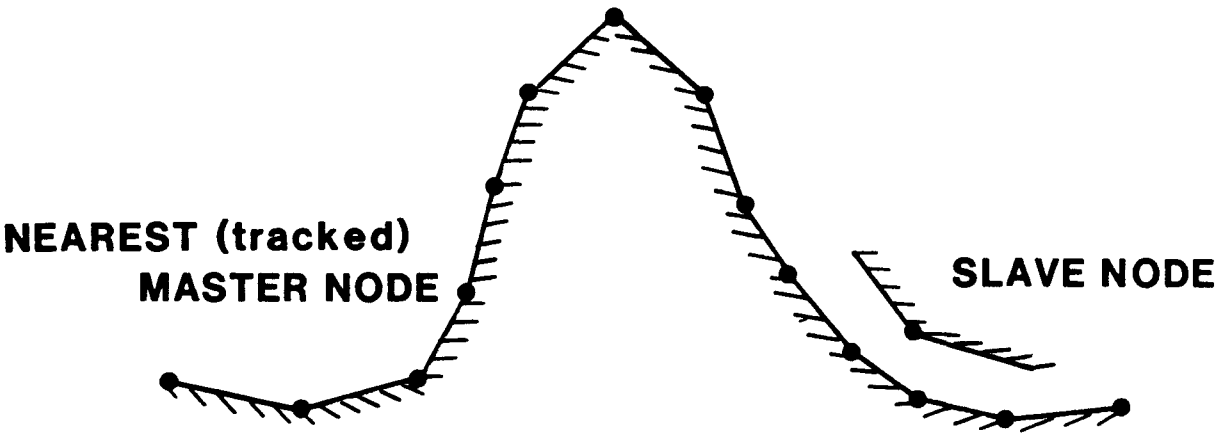


Figure 6.2.3. Case Where a Slave Node is Tracking the Master Surface on the Wrong Side of a Peak

In addition to the pathological cases described above, the tracking algorithm in PRONTO does not yet support a surface contacting itself. This capability would be useful for buckling shells which fold upon themselves, as shown in Figure 6.2.4. The problem in this instance is that the tracking scheme will always find that each node is contacting itself. Presently, the only way to handle this situation is to divide the surface at the crease points. The best approach is to run the calculation with fairly frequent restart dumps (Appendix A, command 8), identify a restart state which occurs after the surface has buckled but before contact, then restart from this state (Appendix A, command 7) with the proper contact surfaces inserted. The greatest difficulty with this technique is that it requires manipulation of the GENESIS mesh file (Appendix D).

6.2.4 Determination of Contact

This section describes how we decide if each slave node is in contact. If a slave node is in contact, we also decide which of the two master segments connected to the nearest master node is in contact.

The first task we perform in order to determine contact is to orient the slave node with respect to the master segments connected to the tracked master node. This entails calculating the local depth and position coordinates for both the precedent and antecedent master segments. Figure 6.2.5 illustrates the geometric interpretation of these quantities which are defined below.

$$\delta_p = \mathbf{n}_{I-1} \cdot (\mathbf{x}_I - \mathbf{x}_s) \tag{6.2.11}$$

$$\chi_p = \mathbf{s}_{I-1} \cdot (\mathbf{x}_I - \mathbf{x}_s) \tag{6.2.12}$$

$$\delta_a = \mathbf{n}_I \cdot (\mathbf{x}_I - \mathbf{x}_s) \tag{6.2.13}$$

$$\chi_a = \mathbf{s}_I \cdot (\mathbf{x}_s - \mathbf{x}_I) \tag{6.2.14}$$

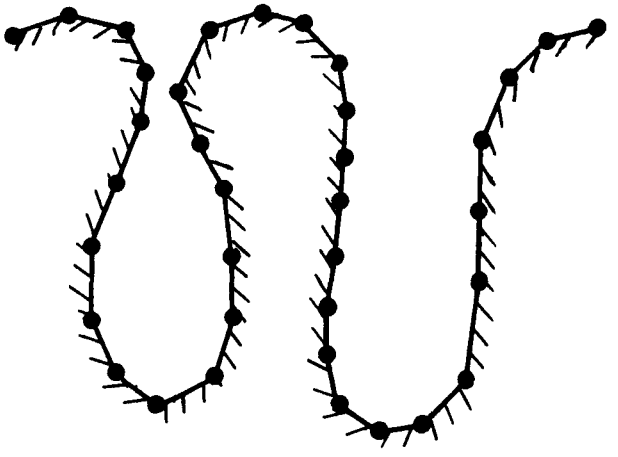


Figure 6.2.4. Case Where a Surface is Contacting Itself

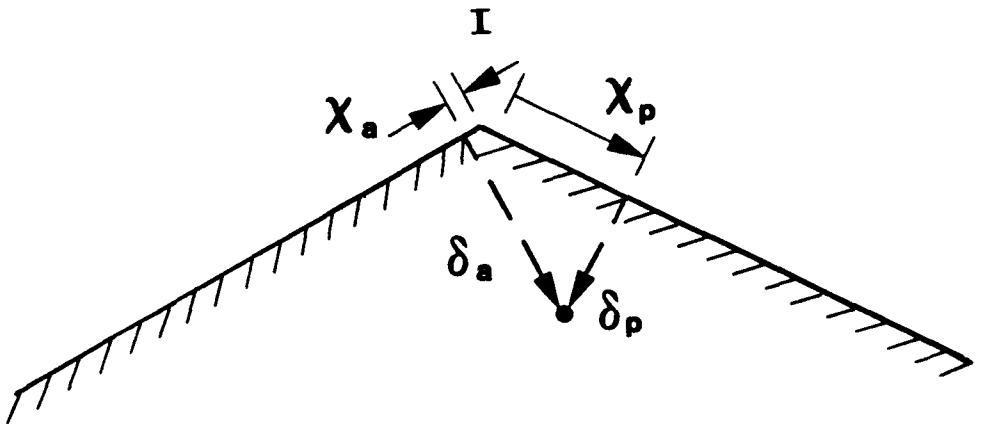


Figure 6.2.5. Definition of Local Depth and Position Coordinates

If its depth (δ) is positive, we say that the slave node is penetrating that segment. If its position (χ) is positive, we say that the slave node is along that segment.

It is helpful to visualize the dichotomy of the two master segments as near and far, rather than precedent and antecedent. The distinction is made by the line which bisects the corner formed at the master node, which is shown as the centerline in Figure 6.2.6. The near segment is the one which lies on the same side of the corner as the slave node. In Figure 6.2.6, the near segment is to the right.

The near segment with regard to the slave node can be determined readily as the segment with the greater position coordinate, as defined in Equations (6.2.12) and (6.2.14), respectively. Figure 6.2.6 illustrates this test.

The ideal condition for determining contact is that the slave node is along and penetrating the contacted segment. Unfortunately, this definition leads to many ambiguous cases because the surface normal is not continuous. One must impose further conditions in order to resolve these ambiguities.

PRONTO's approach to determining contact is based on the premise that a slave node usually contacts the near master segment as defined above. The only exception we make to this rule is when the master surface forms an outside corner. In this instance, as is shown in Figure 6.2.7, it is impossible to determine which master segment is in contact by examining just the depth and position coordinates.

PRONTO detects an outside corner situation when the slave node is along both master segments and penetrating at least one. Figure 6.2.8 shows examples of the range of master surface corners. The hatched areas in frames (b) and (d) indicate the regime where the outside corner ambiguity occurs.

The technique we use to resolve the outside corner ambiguity is to determine if the slave surface near the node in question is more strongly in contact with the far master segment than with the near segment. If this is

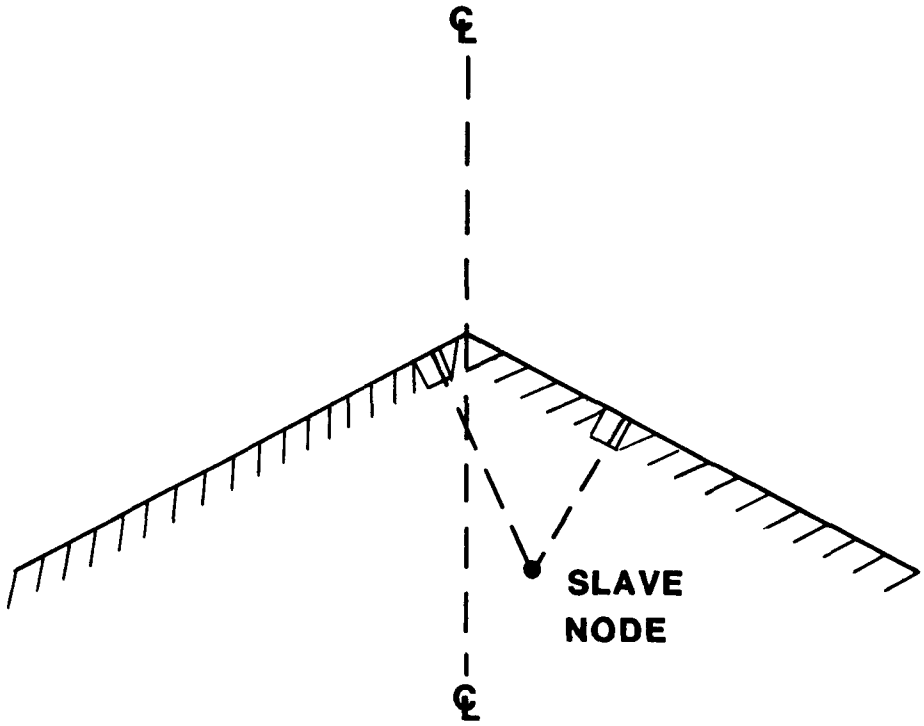


Figure 6.2.6. Definition of Near and Far Master Segments

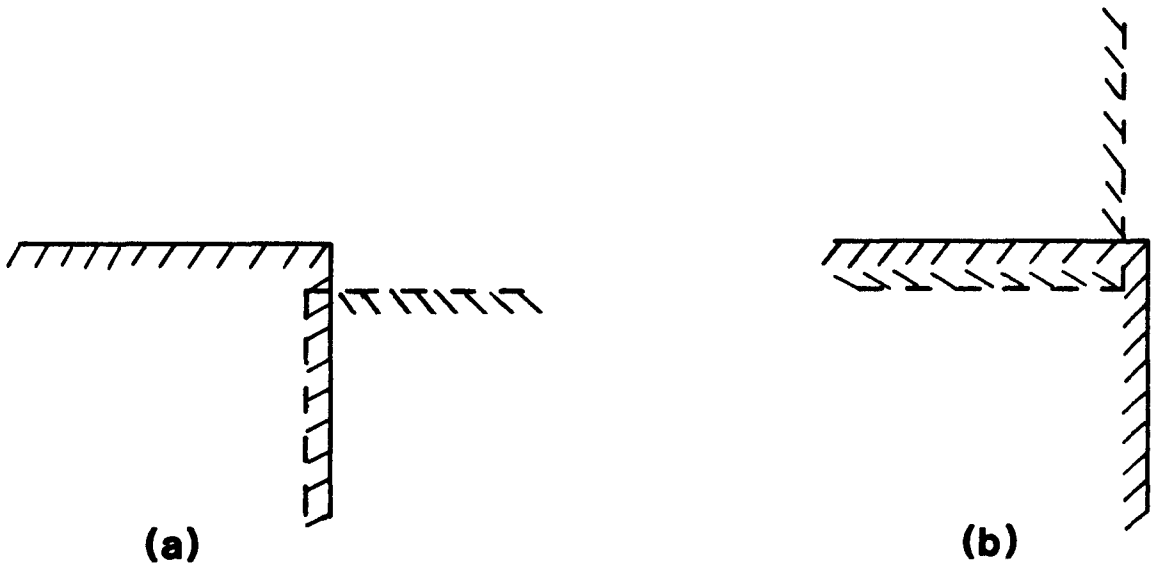


Figure 6.2.7. Illustration of Outside Corner Ambiguities

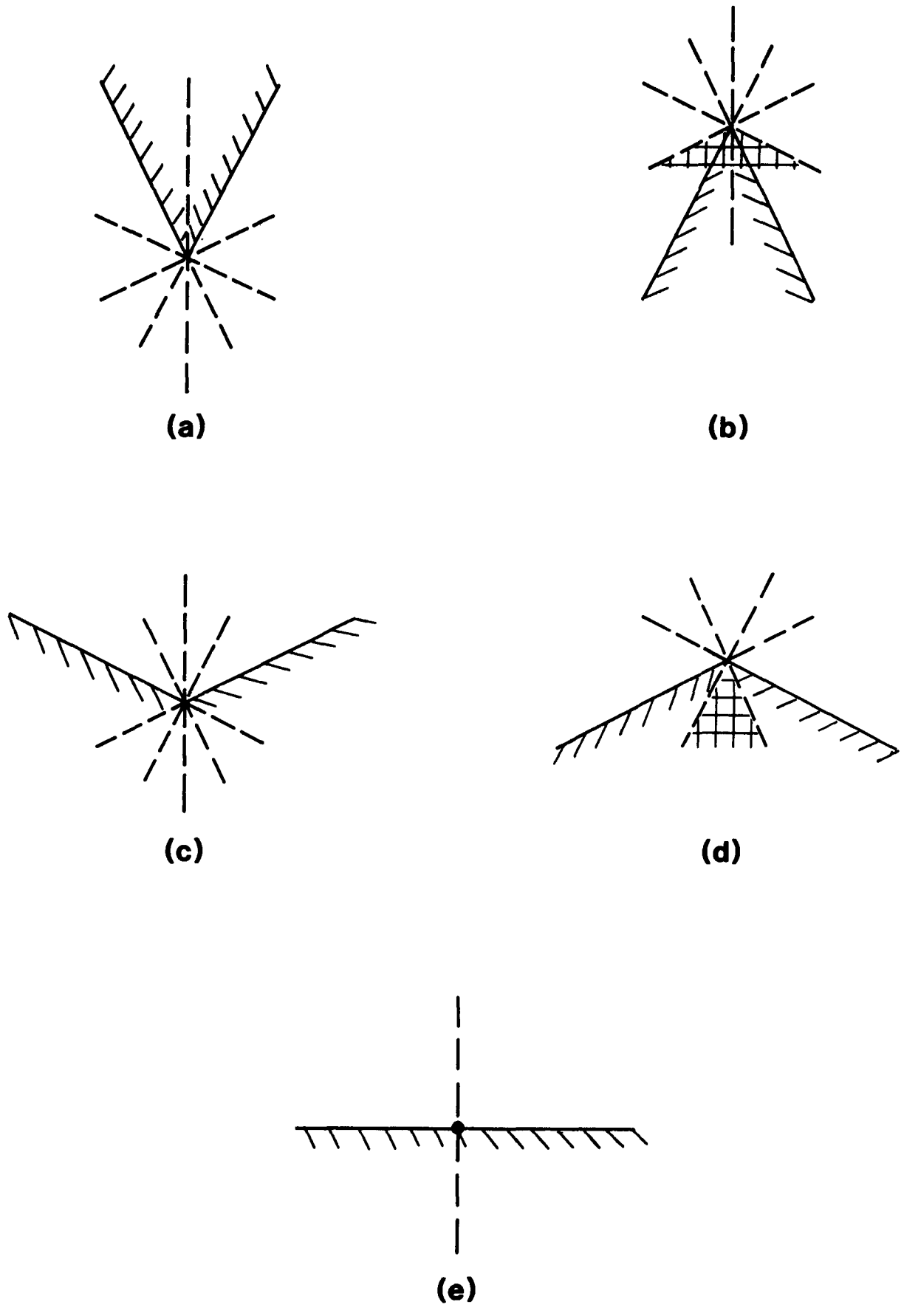


Figure 6.2.8. Examples of the Range of Master Surface Corners

true, we check for penetration of the far segment, rather than the near segment. The exact test we perform is given below for when the precedent segment is the near segment or when the antecedent is the near segment, respectively. These situations are illustrated in Figure 6.2.9. In this figure, we denote the unit normals to the master surface and the slave surface by $\mathbf{m}$ and $\mathbf{n}$, respectively. The subscript p refers to the precedent segment, while the subscript a refers to the antecedent segment.

$$\mathbf{n}_p \cdot \mathbf{m}_a < \mathbf{n}_p \cdot \mathbf{m}_p \tag{6.2.15a}$$

$$\mathbf{n}_a \cdot \mathbf{m}_p < \mathbf{n}_a \cdot \mathbf{m}_a \tag{6.2.15b}$$

The logic that PRONTO follows to determine contact once it has identified the near and far master segments is summarized in Figure 6.2.10. Note that there are only three questions. This logic is so simple that it can be vectorized efficiently.

The final issue for determining contact is to treat edge contacts, i.e., when the nearest master node is at one end of the node list. If the master surface is closed, the nearest master node is the first in the list (1), and the precedent segment (0) is in contact, then we wrap around and contact the last segment (NSIDES) in the list. This reverses the action of Equation (6.2.6a). If the master surface is open, it is not legitimate to contact the precedent segment (0) of the first node (1) nor the antecedent segment (NSIDES+1) of the last node (NNODES). This is because these "segments" where manufactured by Equation (6.2.6b). If this occurs, PRONTO will print an error message and stop.

6.2.5 Contact Forces

We use a partitioned kinematic approach to enforce compliance between two contact surfaces. This means that each surface acts as a master for a fraction of each time step and as a slave for the remainder.

The first task in restoring compliance is to calculate the penetration forces imposed on the master surface by the slave surface. We define these

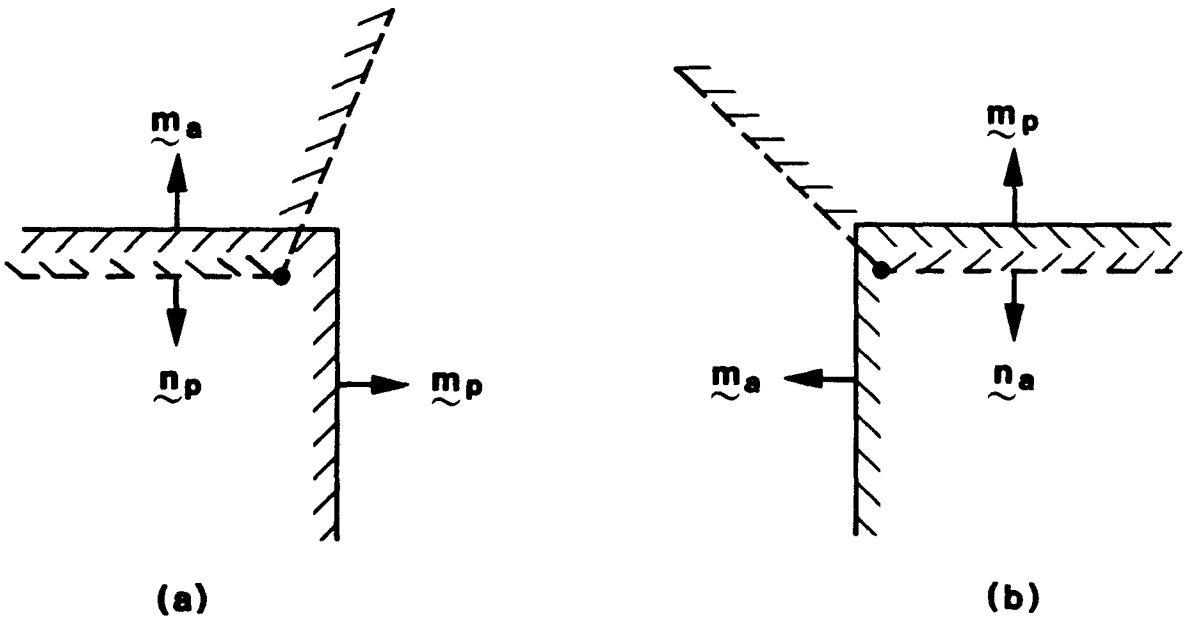


Figure 6.2.9. Outside Corner Contacts

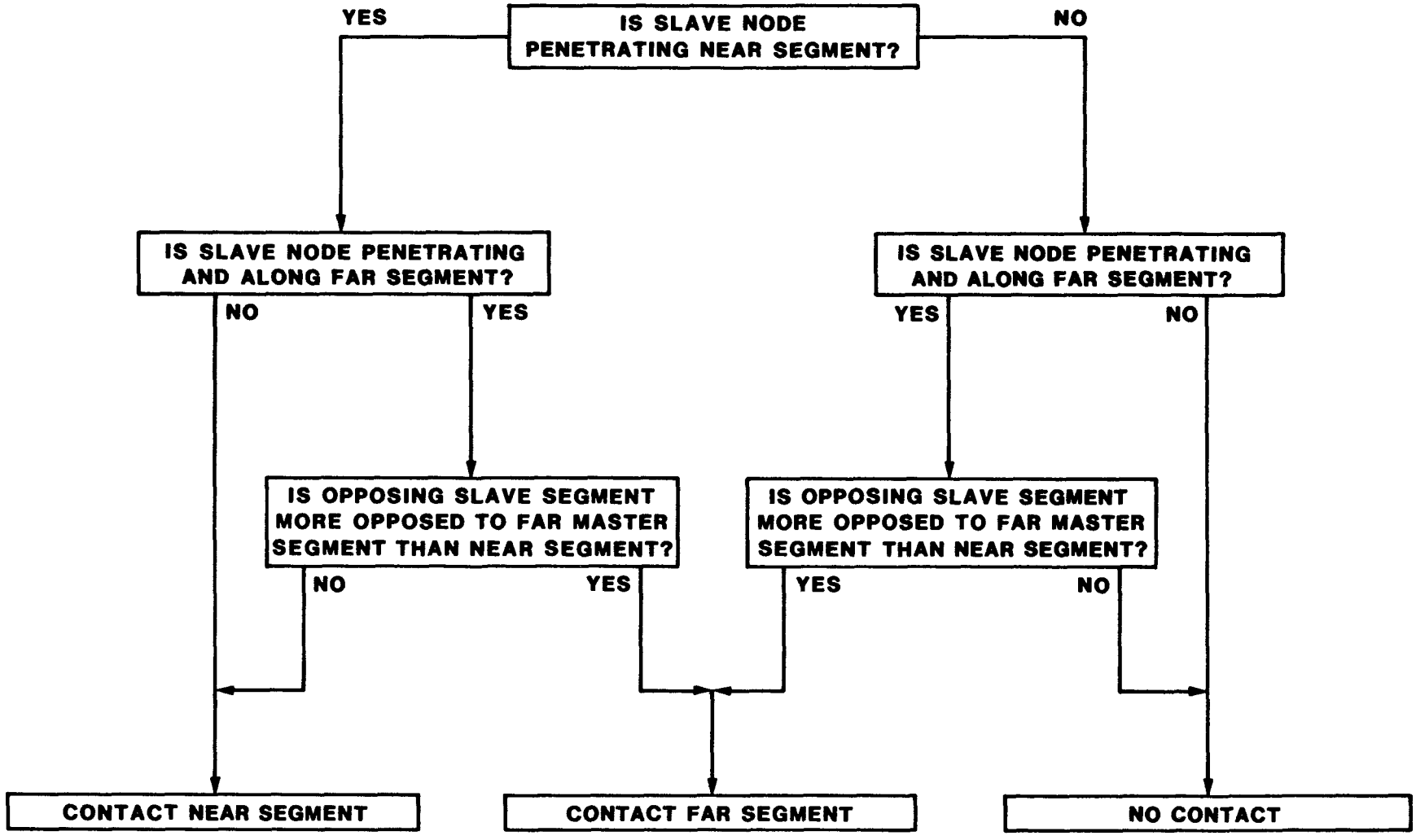


Figure 6.2.10. Flowchart of Logic PRONTO 2D Uses to Determine Contact Once it has Identified the Near and Far Master Segments

forces as a fraction of the forces which would be imposed by the slave nodes if the master surface was rigid. This fraction is the partition factor β, which represents the fraction of each time step for which these surfaces act as master and slave, respectively. Their roles are reversed for the remaining fraction $(1-\beta)$.

The penetration force for a slave node is expressed by

$$\mathbf{f}_p = \beta\, m_s \,/\, \Delta t^2\; \mathbf{n}\cdot(\hat{\mathbf{x}}_s - \hat{\mathbf{x}}_1)\; \mathbf{n} \tag{6.2.16}$$

where m_s is the mass of the slave node, and $\hat{\mathbf{x}}_s$ and $\hat{\mathbf{x}}_1$ are the predicted coordinates of the slave and precedent master node, respectively. This is illustrated in Figure 6.2.11.

Next we want to find the response of the master surface to these penetration forces, such that the response of each contacting slave node is constrained by its master nodes as shown below.

$$\mathbf{a}_{ns} = (1 - \xi)\; \mathbf{a}_{n1} + \xi\; \mathbf{a}_{n2} \tag{6.2.17}$$

where $\mathbf{a}_{ns}$, $\mathbf{a}_{n1}$, and $\mathbf{a}_{n2}$ are the acceleration responses of the slave node, precedent master node, and the antecedent master node, respectively. The interpolation variable ξ is given by

$$\xi = \frac{\mathbf{s}\cdot(\hat{\mathbf{x}}_s - \hat{\mathbf{x}}_1)}{\mathbf{s}\cdot(\hat{\mathbf{x}}_2 - \hat{\mathbf{x}}_1)} \tag{6.2.18}$$

Equation (6.2.17) couples the response of individual master nodes. The principle of virtual work is applied to generate the following equations which define the accelerations of the master nodes in response to the penetration forces.

$$(m_I + \sum_s m_{s_I})\; \mathbf{a}_{nI} = \sum_s \mathbf{f}_{s_I} \tag{6.2.19}$$

where the summation is over all slave nodes in contact.

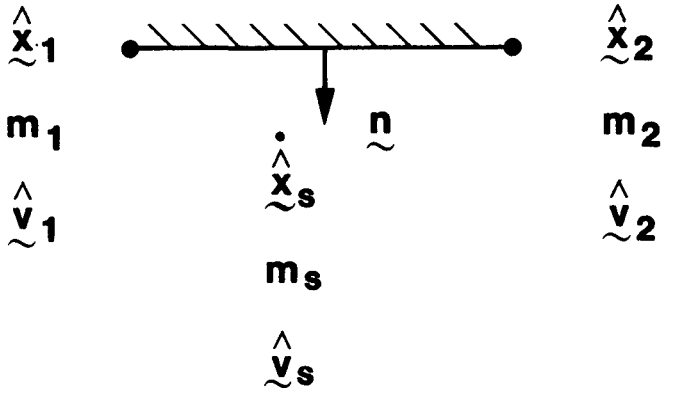


Figure 6.2.11. Definition of Master and Slave Node Quantities for a Contact

The above expression represents a set of uncoupled equations; one for each master node. The mass and force contributions to the above assembly for a given slave node are as follows:

$$m_{s_1} = (1 - \xi)\, m_s \tag{6.2.20}$$

$$m_{s_2} = \xi\, m_s \tag{6.2.21}$$

$$\mathbf{f}_{s_1} = (1 - \xi)\, \mathbf{f}_p \tag{6.2.22}$$

$$\mathbf{f}_{s_2} = \xi\, \mathbf{f}_p \tag{6.2.23}$$

After assembling and solving Equations (6.2.19) for the master accelerations, the slave accelerations are interpolated via Equations (6.2.17) and (6.1.18). Note that this slave response restricts the motion induced by the penetration force given in Equation (6.2.16). In the absence of friction, the corrected nodal accelerations of the master and slave nodes, respectively, then are given by

$$\mathbf{a} = \hat{\mathbf{a}} + \mathbf{a}_n \tag{6.2.24a}$$

$$\mathbf{a} = \hat{\mathbf{a}} + \mathbf{a}_n - \mathbf{f}_p/m_s \tag{6.2.24b}$$

6.2.6 Friction

Friction resists the relative tangential motion of the contacting slave nodes. The tangential component of the relative predicted velocity of the slave node with respect to the master surface is given below in terms of the unit tangent vector (6.2.7) and the its position along the side (6.2.18).

$$v_s = \mathbf{s}\cdot(\hat{\mathbf{v}}_s - (1 - \xi)\, \hat{\mathbf{v}}_1 - \xi\, \hat{\mathbf{v}}_2) \tag{6.2.25}$$

As with the penetration force (6.2.16), we define the tangential contact force as a fraction of the force which must be applied to the slave node to cancel its relative tangential velocity. This force is given by

$$f_s = - \beta \, m_s \, / \, \Delta t \, v_s \tag{6.2.26}$$

where the minus sign reflects that this force would be applied in the direction of **s**, but opposing the motion.

PRONTO currently supports three options for friction: no friction, Coulomb friction with a constant coefficient of friction, or the velocity dependent friction law found in HONDO II [6]. The coefficient of friction can be expressed by

$$\mu = \mu_\infty + (\mu_0 - \mu_\infty) \, e^{-\gamma v_s} \tag{6.2.27}$$

where μ_0 and μ_∞ are the low and high velocity friction coefficients, respectively, and γ is a decay constant. Clearly, if γ equals zero, the coefficient of friction is the constant μ_0. Futhermore, if μ_0 is also equal to zero, the surfaces will be frictionless.

The magnitude of the tangential force exerted by the master surface on a slave node cannot exceed the maximum friction force. This constraint is expressed as

$$f_f = \frac{f_s}{|f_s|} \min(\mu \, f_n \, , \, |f_s|) \tag{6.2.28}$$

where f_n is the magnitude of the normal contact force as given below.

$$f_n = m_s \, \mathbf{n} \cdot (\mathbf{a}_{ns} - \mathbf{f}_p / m_s) \tag{6.2.29}$$

Applying this force to the slave node and balancing forces to the master nodes, then dividing by the appropriate nodal mass yields the following expressions for the tangential accelerations to these respective nodes.

$$\mathbf{a}_{s0} = f_f \, / \, m_s \, \mathbf{s} \tag{6.2.30}$$

$$\mathbf{a}_{s1} = - (1 - \xi) \, f_f \, / \, m_1 \, \mathbf{s} \tag{6.2.31}$$

$$\mathbf{a}_{s2} = - \xi \, f_f \, / \, m_2 \, \mathbf{s} \qquad (6.2.32)$$

Finally, by adding the above tangential accelerations to Equations (6.2.24a) and (6.2.24b), the corrected total acceleration of the contact nodes is expressed in general form for master and slave nodes, respectively, by

$$\mathbf{a} = \hat{\mathbf{a}} + \mathbf{a}_n + \mathbf{a}_s \qquad (6.2.33a)$$

$$\mathbf{a} = \hat{\mathbf{a}} + \mathbf{a}_n - \mathbf{f}_p / m_s + \mathbf{a}_s \qquad (6.2.33b)$$

7.0 BOUNDARY CONDITIONS

PRONTO contains several types of boundary conditions. In this section, we describe how these are implemented in the program. It is important to note that the order in which these boundary conditions is applied is crucial to the accuracy of the program. In Chapter 8, we describe the initialization and time stepping algorithm which should be referred to in order to determine exactly when the different boundary conditions are applied.

7.1 Kinematic Boundary Conditions

The kinematic boundary conditions described below are all accomplished by altering the accelerations of the nodal points. The application of these boundary conditions does not vectorize because they require a function look-up and a scatter of values. All of the kinematic boundary conditions are nodal boundary conditions.

7.1.1 No Displacement Boundary Conditions

The no displacement boundary conditions are accomplished by setting the acceleration of the node to zero.

Note: If velocity or acceleration boundary conditions are specified on a node which has a no displacement boundary condition, they will override the displacement boundary condition.

7.1.2 Prescribed Velocity Boundary Conditions

The prescribed velocity boundary conditions are accomplished by altering the nodal point acceleration such that when the accelerations are integrated once, they provide the proper value of the nodal velocity. The nodal value of acceleration for the time step is calculated by the program as

$$a_t = (v_{t+\Delta t} - v_t)/\Delta t \ . \qquad (7.1.1)$$

The velocity at the end of the time step is computed by

$$v_{t+\Delta t} = s \cdot f(t + \Delta t) \qquad (7.1.2)$$

where s is the scale factor and f(t) is the history function defined by the user. In Equation (7.1.1), the value of velocity at the beginning of the time increment, v_t, is the value computed by Equation (7.1.2) at the previous time increment.

Note: If a prescribed acceleration boundary condition is specified by the user on the same node as a prescribed velocity boundary condition, it will override the prescribed velocity boundary condition.

7.1.3 Prescribed Acceleration Boundary Conditions

Prescribed acceleration boundary conditions are applied by the program by setting the nodal acceleration during the time increment to the value given by

$$a_t = s \cdot f(t) \qquad (7.1.3)$$

where s is the scale factor and f(t) is the history function defined by the user.

Note: A prescribed acceleration boundary condition will override any other kinematic boundary condition on the same node.

7.2 Traction Boundary Conditions

The boundary conditions described below involve applied forces to the boundary of the mesh. The pressure and nonreflecting boundary conditions are side boundary conditions, while the nodal force boundary condition is a nodal boundary condition.

7.2.1 Pressure

The set of consistent nodal point forces arising from pressures distributed over an element side are defined via the principal of virtual work by

$$\delta u_{iI} f_{iI} = \delta u_{iI} \int_s \phi_I (-p n_i) dA \ . \tag{7.2.1}$$

where the range of the lower-case subscripts is 1 to 2 and the upper-case subscripts 1 to 4.

Since the virtual displacements are arbitrary, they may be eliminated to yield:

$$f_{iI} = - \int_s \phi_I \ p n_i \ dA \tag{7.2.2}$$

The most general pressure distribution we allow is mapped from nodal point pressure values via the isoparametric shape functions. The resulting expression for the consistent nodal forces is

$$f_{iI} = - p_J \int_s \phi_I \phi_J \ n_i dA \ . \tag{7.2.3}$$

For the four node constant stress element used in PRONTO, ϕ_I is given by

$$\phi_I = \frac{1}{2} \Sigma_I + \xi \ \Lambda_I \ , \quad - \frac{1}{2} \leq \xi \leq \frac{1}{2} \tag{7.2.4}$$

where

$$\Sigma_i = \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} \quad \Lambda_I = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} \tag{7.2.5}$$

and $n_i n_i = 1$. For the geometry and pressure distribution shown in Figure 7.2.1, it can be shown that

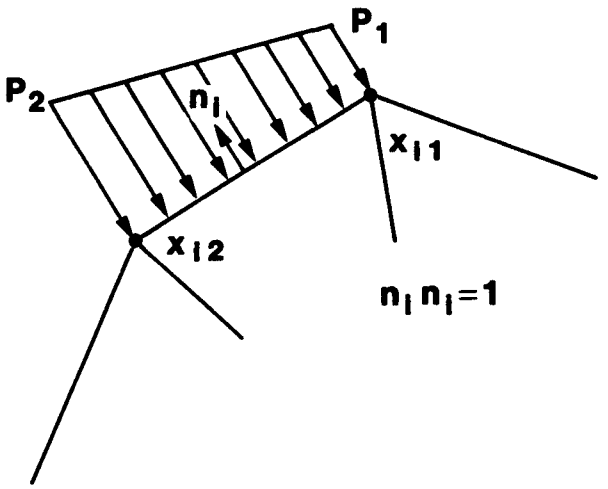


Figure 7.2.1. Definition of a Pressure Boundary Condition Along an Element Side

$$x_i = x_{iI}\ \phi_I \tag{7.2.6}$$

and

$$n_i dA = e_{ij3}\ \frac{\partial x_i}{\partial \xi_j}\ d\xi = e_{ij3}\ x_{jK}\ \Lambda_K\ d\xi\ . \tag{7.2.7}$$

Then the consistent nodal forces can be written as

$$f_{iI} = -p_J e_{ij3}\ x_{jK}\ \Lambda_K \int_{-\frac{1}{2}}^{\frac{1}{2}} \phi_I \phi_J\ d\xi\ . \tag{7.2.8}$$

Combining Equations (7.2.4), (7.2.5), and (7.2.8):

$$f_{iI} = -p_J e_{ij3}\ x_{jK}\ \Lambda_K \left[\frac{1}{4}\ \Sigma_I \Sigma_J + \frac{1}{12}\ \Lambda_I \Lambda_J\right]\ . \tag{7.2.9}$$

The above expression is evaluated as

$$N_i = -e_{ij3}\ x_{jK}\ \Lambda_K = \begin{Bmatrix} y_1 - y_2 \\ x_2 - x_1 \end{Bmatrix} \tag{7.2.10}$$

and

$$f_{iI} = N_i \cdot \frac{1}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \cdot \begin{Bmatrix} p_1 \\ p_2 \end{Bmatrix}\ . \tag{7.2.11}$$

The nodal values for the pressure are calculated using the user supplied scale factor and time history function. The values are calculated for the beginning of the time step.

The application of the pressure boundary conditions is fully vectorized. Blocks of element sides are processed in vector blocks using the scratch element space. After the consistent nodal point forces are calculated for a block of element sides, they are accumulated into the global nodal force array.

7.2.2 Moving Pressures

The moving pressure boundary condition implemented in PRONTO represents a relatively simple way of incorporating both a spatial and temporal distribution of pressure loading on a surface. The implementation described here is intended for blast type loading on a surface where the blast originates from some point defined by the coordinates (x0,y0) and propagates along the surface. We assume that the surface is flat and the distance from any point on the surface to point (x0,y0) is given by d. Then the pressure at any point is written as

$$p(\tau,d) = a\tau e^{-b\tau} \tag{7.2.12}$$

where τ is the time measured from the arrival of the pressure wave at the point and a and b are functions of distance, which are defined below. If w is the propagation speed of the pressure wave along the surface, then τ is given by

$$\tau = t_o - d/w \tag{7.2.13}$$

where t_o is the pressure initiation time at the point (x_o,y_o). The time at which Equation (7.2.12) gives a maximum for the pressure is given by

$$\tau_{max} = 1/b \tag{7.2.14}$$

which we refer to as the rise time. The peak pressure obtained at this time is

$$p_{max} = \frac{a}{b} e^{-1} \tag{7.2.15}$$

We allow the user to define two functions of distance from the point (x0,y0) which describe the behavior of the pressure wave. The first function defines the peak pressure as a function of distance while the second describes the rise time as a function of distance. Using Equations (7.2.14) and (7.2.15), we can write the parameters a and b as functions of distance

$$a(d) = \frac{f_1(d)}{f_2(d)}\, e^{+1} \qquad b(d) = \frac{1}{f_2(d)} \tag{7.2.16}$$

The user can define the functions in any manner he sees fit which allows for a quite general specification of the moving pressure wave. If the user inputs a zero value for the propagation speed, w, the code assumes that the pressure is applied instantaneously along the surface (i.e., this corresponds to an infinite propagation speed). If the assumed pressure description given by Equation (7.2.12) is not suitable, it is a simple task to change this description to some other two parameter functions and alter the code accordingly. Of course the definition of the parameters, a and b, as a function of distance given by Equation (7.2.16) would have to be rederived and changed as well.

7.2.3 Nodal Forces

Nodal point load forces are applied by determining the magnitude of the force determined by the user supplied scale factor and time history function. The time history function is evaluated at the beginning of the time step.

7.3 Nonreflecting Boundaries

In a number of geotechnical applications, it is desirable to model an infinite or semi-infinite space. In these applications, waves are transmitted outward from some disturbance and are absorbed in the far field. PRONTO contains a boundary condition specification which will absorb waves and not reflect them back into the interior mesh. This allows for a much smaller mesh and a significant reduction in the number of degrees of freedom in the problem.

The absorbing or nonreflecting boundary which is implemented in the code was proposed by Lysmer and Kuhlemeyer [37] and discussed in detail by Cohen and Jennings [38]. The exterior infinite region is replaced by an energy absorbing boundary condition. The basic idea is to apply boundary

tractions which will exactly cancel the stresses which are generated at the free surface. On this boundary surface, tractions are applied of the form

$$\sigma_n = \rho V_p \dot{u}_n \tag{7.3.1}$$

and

$$t_s = \rho V_s \dot{u}_t \tag{7.3.2}$$

where:

σ_n = normal stress applied to the boundary

t_s = shear stress applied to the boundary

$\dot{u}_n$ = velocity component normal to the boundary

$\dot{u}_t$ = velocity component tangential to the boundary

ρ = current density of the material at the boundary

V_s = current s-wave velocity in the material at the boundary

V_p = current p-wave velocity in the material at the boundary

The wave speeds required in Equations (7.3.1) and (7.3.2) are calculated using the current shear and dilatational moduli determined in Section 3.4.

The tractions given by Equations (7.3.1) and (7.3.2) are used with the consistent nodal forces which were derived in Section 7.2.1. The nodal normal and tangential velocities are determined for the two nodes on the element side to determine the correct tractions. These are used in Equation (7.2.11) for both the normal and shear components to give the proper consistent nodal point forces for the absorbing boundary.

The application of the nonreflecting boundary condition is vectorized as is the pressure boundary condition. The effective moduli required for the wave speed determinations in Equations (7.3.1) and (7.3.2) are computed and stored during the loop on the elements for elements having this boundary condition specification.

8.0 INITIALIZATION AND TIME-STEPPING ALGORITHM

8.1 Initialization

The user defines a mechanics problem by specifying material properties, body geometry, initial conditions, tractions, and kinematic boundary conditions. PRONTO does an extensive amount of data checking to try to insure that the user has defined a meaningful mechanics problem. These checks range from mundane (e.g., Are nonzero and positive mass densities provided for the materials?) to more subtle (e.g., Are all the contact surfaces simply connected?). We do not guarantee that PRONTO will always detect a bad set of data, but experience with it has shown that it is usually smarter than the authors.

By "initialization" we mean the calculations which must be performed and the data structures which must be set up before entering the time stepping loop. There are two initialization processes in PRONTO. The first has to do with setting up the initial data structures which are specified by the user. This is all done in the INIT routine which is called from the main program. At this time the initial displacements, velocities, and accelerations are all set to zero. Then the initial velocities defined by the user are set. The stresses are initialized to zero and the internal state variables are initialized to the appropriate values for all of the materials. The initial tracking of the contact surfaces is also performed in the INIT routine.

The second and more subtle initialization which must be performed concerns the resolution of the kinematic constraints which the user has defined with the initial velocity field, which the user has also defined. A simple example of why this is required is the case of a bar striking a rigid wall at some nonzero initial velocity. The user defines a rigid contact surface and gives all the material in the bar an initial velocity. But the first row of nodes (initially on the rigid surface) has the initial velocity at time zero; whereas, physically it is in contact and should have zero velocity. It would be a simple matter to just check for this type of condition and set the nodal velocities to zero, but this procedure fails to

correctly transfer momentum, to correctly initialize the artificial bulk viscosity pressure in the first row of elements, to correctly predict the strain rates, and to correctly predict stable time increment. The problem is especially severe if the impact velocity is on the order of the wave speed of the material. For the case of two deformable surfaces (the general contact case), the correct initialization of the kinematic constraints is of even more importance.

In order to correctly perform the initialization, we must first perform what we call a pseudo-time increment. A trial stable time increment is determined in the INIT routine while calculating the element masses. This time increment is based on the state of the body defined by the user which is, in general, stress free. Essentially, it is the time increment which would have been stable had there been a previous time step. Remember that the stable time increment prediction always looks backward and we rely on the conservativeness of Equation (3.5.5) to remain stable. The pseudo-time increment is performed in the SOLVE routine before entering the time step loop. The algorithm is:

1. Set up a pseudo-time step by setting the accelerations to the trial velocities divided by the pseudo-time increment and the velocities to zero. Note that these accelerations are purely an algorithmic invention and have nothing to do with initial accelerations which we correctly calculate at the top of the time step loop. This set of kinematic conditions is the same as the original set specified by the user if no kinematic conditions are present. The velocities must be zero so that we do not integrate the friction in the initialization calculation.

2. Predict a new configuration based on the pseudo-time increment and the user supplied initial velocities.

3. Calculate acceleration corrections to enforce kinematic constraints on initial velocities.

4. Apply kinematic boundary conditions (except prescribed accelerations).

5. Alter initial velocities to enforce kinematic constraints and reset current coordinates to the original configuration.

8.2 Time Step Loop

The order in which calculations are performed during the time step loop is crucial to the question of the accuracy of the algorithm. Our time stepping algorithm proceeds in the following order:

1. Advance the constitutive state to the end of the time increment, calculate internal forces due to stress divergence, artificial viscosity, and hourglass resistance, and determine the stable time increment.

2. Apply external loads: pressures, quiet boundaries, and nodal forces.

3. Calculate accelerations and predict the configuration at the end of time step ignoring kinematic constraints. This predicted configuration will be used in the contact and rigid surface routines to determine the corrections which must be made to the accelerations to bring the surfaces back into compliance.

4. Enforce contact surface and rigid surface constraints by altering the accelerations.

5. Apply kinematic boundary conditions by altering the accelerations so that the kinematic constraints are satisfied: displacement, velocity, and/or acceleration.

6. Write output, if timely.

7. Integrate the velocities and displacements using the altered accelerations and compute the current spatial coordinates which reflect the kinematic constraints.

8. Update the current value of time and go back to step 1 if more time is required. Otherwise, exit the time step loop.

Most of the computation time in PRONTO occurs in step 1. This is where the elements are processed in vector block loops. We calculate the gradient operator, determine the strain rates, advance the constitutive state, determine the critical time increment, and calculate the divergence of the stresses for a block of elements at a time.

9.0 NUMERICAL EXAMPLES

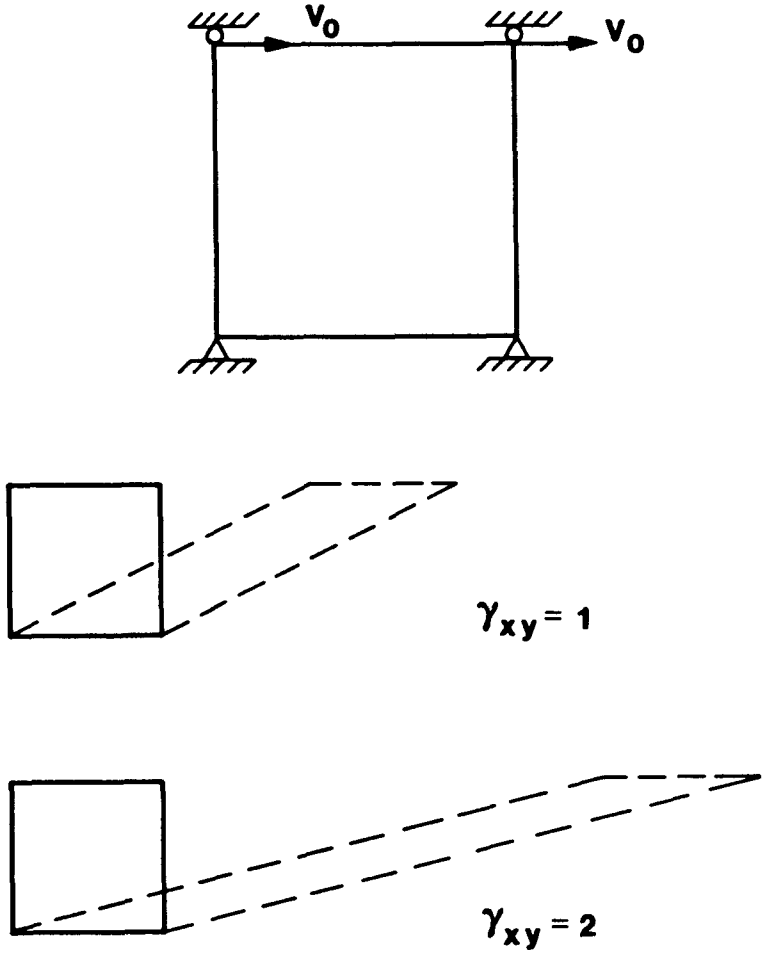
In this chapter, we present several representative example problems which demonstrate some of the features of the numerical algorithms which were described in previous chapters. We will refer to the particular section or chapter so that the reader can gain a better understanding of algorithms we consider vital to the success and accuracy of PRONTO. Only the PRONTO input instructions are shown here; the geometric definitions of the mesh are not given.

9.1 Simple Shear

This is a simple one element example problem which demonstrates the accuracy of the finite rotation algorithm described in Section 3.3. Figure 9.1.1 shows the one element mesh and the boundary conditions applied as well as the PRONTO input required to define the problem. The prescribed constant velocity on the top of the element is maintained until the element has experienced a total shear strain of 400 percent. The deformations in the element are totally prescribed by the kinematic boundary conditions, which means that the time step is arbitrary and the time integration cannot go unstable. Hence, the choice of the mass density simply serves to fix the number of time increments used to integrate the stresses and rotations to the end of the problem. We ran this problem using a mass density which resulted in 400 time steps (1 percent shear strain per time step) and the results were almost indistingishable from the analytical solution given in Figure 2.2.2. The numerical accuracy does not degenerate significantly until the strain increment approaches 5 percent (of course, this strain increment is unreasonable for the integration of any real constitutive model).

9.2 Rotating Cylinder

The rotating cylinder problem illustrated in Figure 9.2.1 was proposed by Longcope and Key [39] as a means of exercising finite rotation algorithms. The cylinder has an initial angular velocity of 4000 rpm with a zero initial stress state. Of course, this is not physically possible



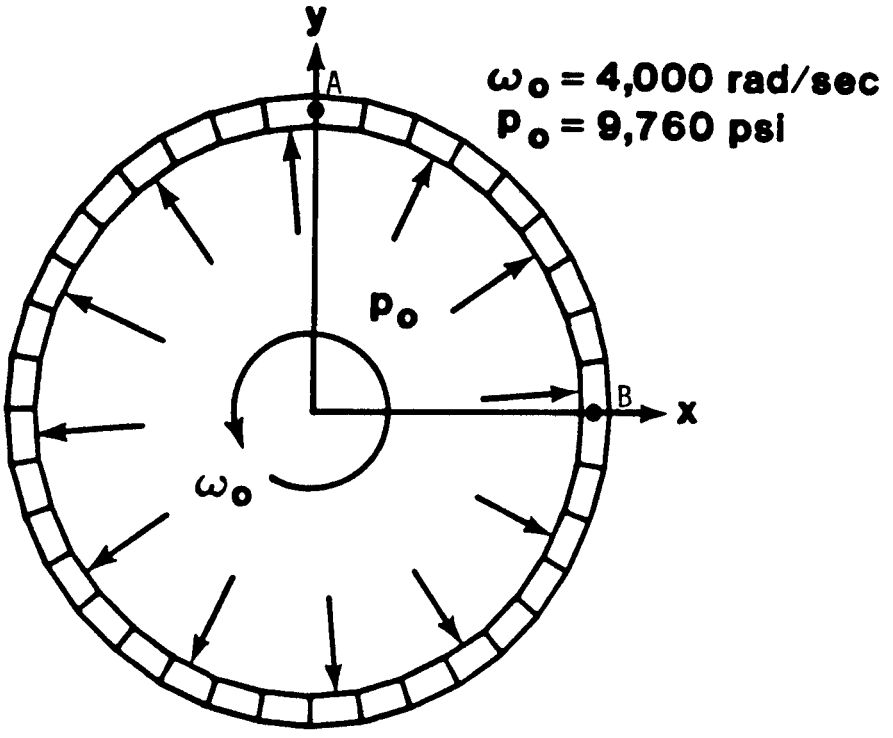
ONE ELEMENT SIMPLE SHEAR PROBLEM

```
TITLE
SIMPLE SHEAR TEST, STRAIN INC = .1 PERCENT
PLANE STRAIN
MATERIAL,1,ELASTIC,1.346E-6
YOUNGS MODULUS,1.,POISSONS RATIO,0,
END
PLOT,NODAL,DISPL
PLOT,ELEMENT,STRESS,ROTATION
TERMINATION TIME,1
OUTPUT TIME,.1
PLOT TIME,.25E-2
FUNCTION,1
0,1
10,1
END
PRESCRIBED VEL,X,4,1,8
PRESCRIBED VEL,X,3,1,8
NO DISPL,X,1
NO DISPL,X,2
NO DISPL,Y,1
NO DISPL,Y,2
NO DISPL,Y,3
NO DISPL,Y,4
EXIT
```

Figure 9.1.1. Geometry and PRONTO 2D Input for the Simple Shear Problem

because the body forces would generate a stress field under this angular velocity, but these initial conditions are acceptable for this numerical experiment. The inside of the cylinder experiences a step load in pressure to a value of 9780 psi, which causes the cylinder to expand. The material model is elastic/plastic with kinematic hardening. Figure 9.2.2 shows the maximum principal stress in the cylinder for the element labeled A in Figure 9.2.1. Since the cylinder is thin, the maximum principal stress corresponds to the hoop stress and is the only stress component of interest in the problem. Examination of Figure 9.2.2 shows that the material first loads up elastically until the yield stress of 41500 psi is reached. It then strain hardens until the cylinder reaches its maximum expansion, where an elastic rebound occurs. The first cycle of rebound actually unloads to the point that the material begins to yield again. Remember that the hardening is kinematic so when the material begins to yield again, the state of stress has actually crossed back over the yield surface in stress space and strain hardening occurs as it begins to yield. The cylinder then settles into a purely elastic rebound mode where the cylinder oscillates in a "breathing" mode. This oscillation occurs entirely within the yield surface in stress space.

This problem illustrates two features of the numerical algorithms in PRONTO. First, the example problem serves as a test of the finite rotation algorithm of Section 3.3. During the time shown in Figure 9.2.2, the element labeled A rotates 90 degrees to the position labeled B in Figure 9.2.1. This is a very large finite rotation in which both the stress and the tensor internal state variable of backstress must be correctly rotated. More than ten thousand time steps were performed during the 90-degree rotation. The obvious question is, "What is the numerical drift in the solution over such a large number of time steps?" To determine this, we also plotted the maximum principal stress for the element which started in position B, and the results were identical to those for element A. These are two elements which have stress states which are oriented 90 degrees from one another. The principal values do not drift apart even though the material is experiencing extremely large rotations and large nonlinear deformations.




```
TITLE
 PRESSURIZED RING TEST PROBLEM, OMEGA = 4000
INITIAL VELOCITY ANGULAR = 1,4000
TERMINATION TIME = .4E-3
OUTPUT TIME,.01E-3
PLOT TIME = 0.
MATERIAL,1,ELASTIC PLASTIC,2.508E-4
YOUNGS MODULUS = 1.03E7 , POISSONS RATIO = .33333
YIELD STRESS = 4.15E4 , HARDENING MODULUS = 5.17E5 , BETA = 0
END
FUNCTION = 1
0,1
1,1
PRESSURE = 100,1,9760
EXIT
```

Figure 9.2.1. Definition of the Rotating Cylinder Problem

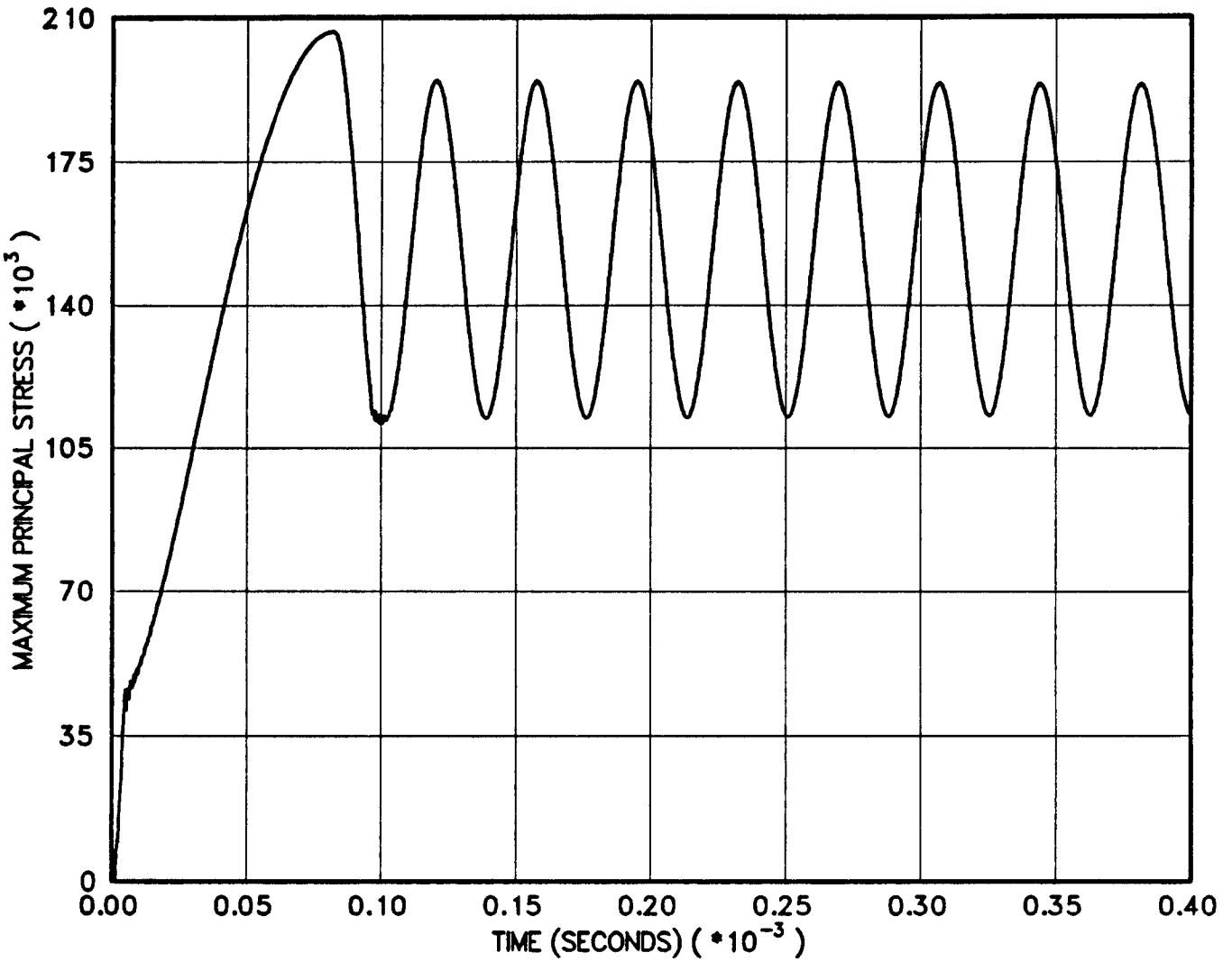


Figure 9.2.2. Principal Stress Versus Time for all Points on the Rotating Cylinder

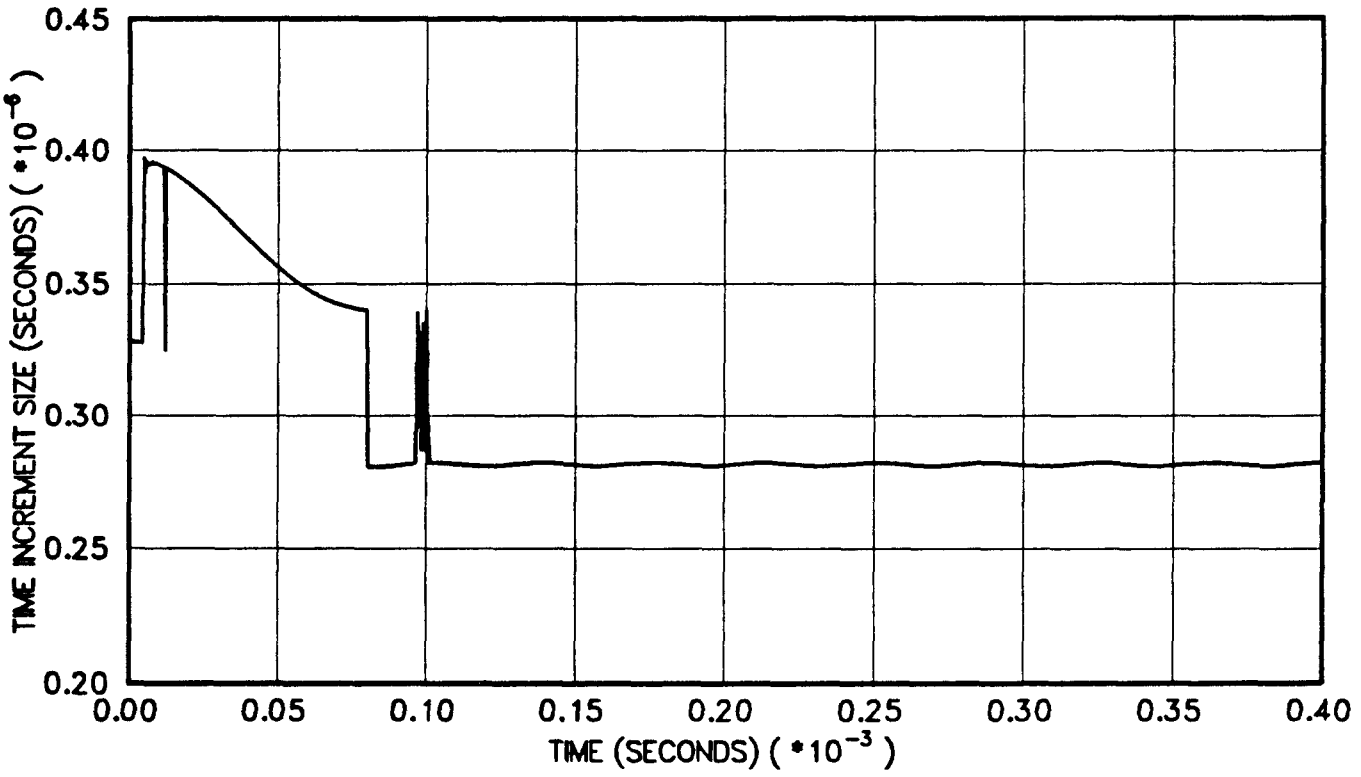


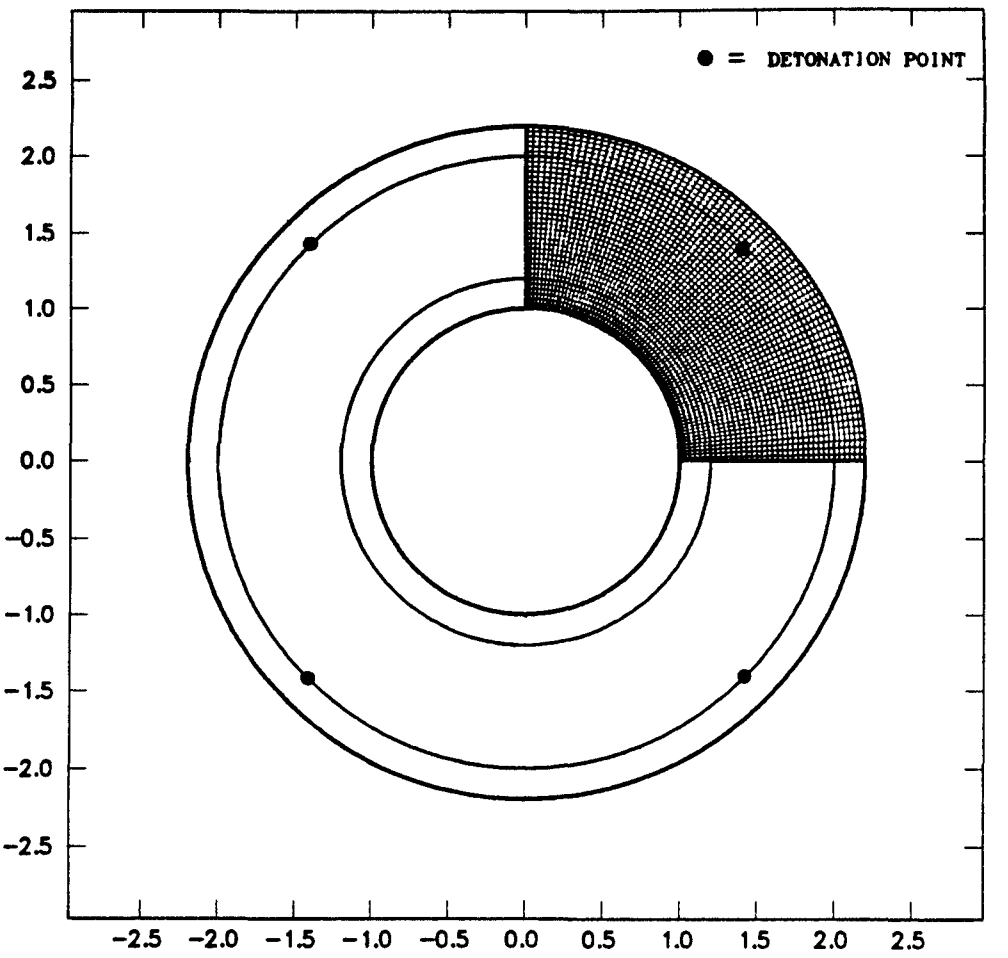
Figure 9.2.3. Internally Computed Time Increment Size Versus Time for the Rotating Cylinder Problem

The second feature which is illustrated is the stable time increment control described in Section 3.5. Figure 9.2.3 shows a plot of the time increment chosen by the code as a function of time. During the initial elastic loading the stable time increment is relatively constant, decreasing slightly due to the change in radius of the cylinder. As the cylinder yields, the time step rises because the material has a softer effective modulus. As elastic rebound begins, the time increment size returns to the small size that was used in the initial elastic loading. When the cylinder begins to yield again on the backside of the first rebound, we see a jump in the stable time increment. Finally, the rebound becomes completely elastic and the time step increment becomes almost constant with the slight oscillation due to the oscillation in the geometry of the cylinder.

9.3 Explosive Pipe Closure

In this problem, two concentric pipes have the annulus between them filled with high explosive (HE). The inside radius of the inner pipe is 1 cm, and the inside radius of the outer pipe is 2 cm. Both pipes are steel with a wall thickness of .2 cm. Each pipe has 6 elements through the wall thickness, and the HE has 24 elements through its thickness. The problem is analyzed as a plane strain problem and only one-quarter of the geometry is modeled due to symmetry. The analysis used 75 elements around the quarter circumference, for a total of 2700 elements and 2812 nodes in the problem. The input data deck and a schematic drawing of the problem are shown in Figure 9.3.1. This particular example is included here to demonstrate the use of an equation of state, in this case the JWL explosive equation of state (see Section 5.2.4.). The HE was detonated at a point on the inside edge of the outer pipe which lies equidistant from the two coordinate axes (along the line x = y). The units in this problem were cm, gm, and usec.

The problem was run to 7.5 microseconds, at which time the inner pipe has been nearly compressed into a solid mass. Figure 9.3.2 shows a sequence of deformed configurations. The numerical solution took 434 time steps and required 14.35 cpu seconds on the CRAY XMP-24 under CTSS.




```
TITLE
 EXPLOSIVE PIPE CLOSURE
TERMINATION TIME = 7.5
OUTPUT TIME = .1
PLOT TIME = .25
PLOT ELEMENT = PRESSURE,DENSITY
NO DISPL X = 200
NO DISPL Y = 100
MATERIAL,1,ELASTIC PLASTIC,7.846
YOUNGS MODULUS = 2.211E-2 , POISSONS RATIO = .279
YIELD STRESS = .0043 , HARDENING MODULUS = 0 , BETA = 0
END
MATERIAL,2,HYDRO,1.9
PRESSURE CUT = 0.
END
EQUATION OF STATE,2,JWL
CD=.7596 A=5.206 B=.053 R1=4.1 R2=1.2 OMEGA=.35 ENERGY=.069
DETONATION POINT = 2, 1.414, 1.414, 0.
END
EXIT
```

Figure 9.3.1. Definition of the Explosive Pipe Closure Problem

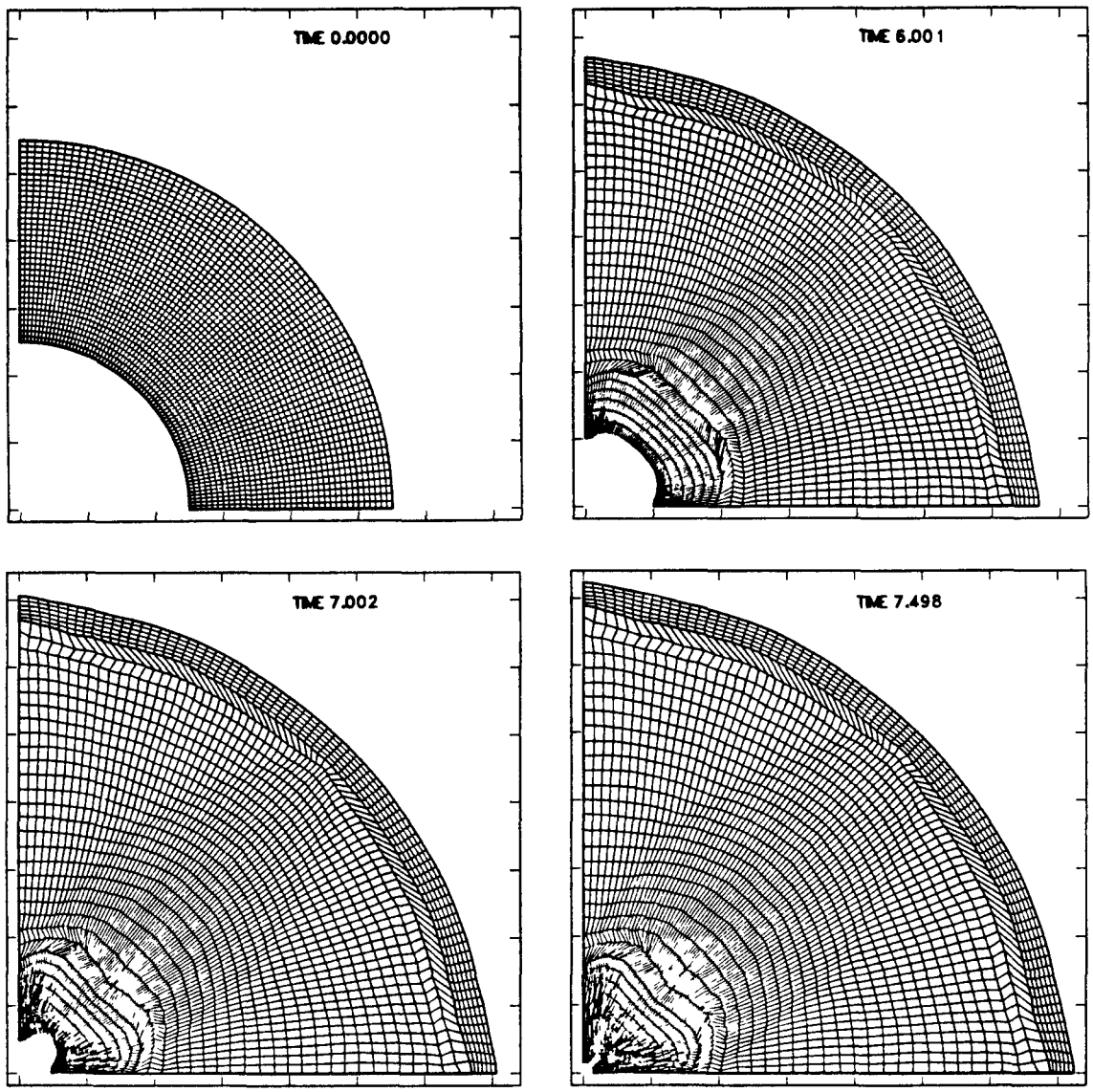


Figure 9.3.2. Sequence of Deformed Configurations for the Explosive Pipe Closure Problem

9.4 Missile Impact

The next example problem involves the structural effects of a 600 foot per second head-on impact of a missile against a hard target. The response of the front section of the missile was computed for 200 microseconds following the initial instant of impact against the assumed target, an unyielding plane perpendicular to the direction of motion of the missile. The finite element mesh for the calculation was comprised of 1157 nodes and 945 elements. The model included the region of the missile from the tip to the point at which the forward case switches from conical to a cylindrical cross section. An extra region was appended to the model at the latter point in order to add the equivalent mass of the remainder of the missile.

Thirteen contact surfaces were used in the model to separate the various components in the missile. A rigid surface was used to model the hard target. Figure 9.4.1 shows the mesh and the location of the various slide lines used in the model. Figure 9.4.2 contains the input file for this problem. We have included this example problem to illustrate how easy it is to use PRONTO with problems containing a large number of contact surfaces which undergo very large deformations.

Figures 9.4.3 and 9.4.4 show the deformed mesh at 100 and 200 microseconds, respectively. This problem required 600 seconds of cpu time on the CRAY/XMP-24 under CTSS.

9.5 Forging Problem

The last example problem represents the drawing of a spherical cup from an initially flat copper disk. Figure 9.5.1 shows the original geometry of the dies and the disk. All of the nodes on the rigid dies were given prescribed displacement boundary conditions using either the no displacement boundary condition or the prescribed velocity boundary condition. Consequently, there is no deformation in the dies and the material properties used for that material are of no importance. In fact, we used the material deletion option to delete that material from the mesh at the beginning of the analysis. The kinematic boundary conditions applied to the

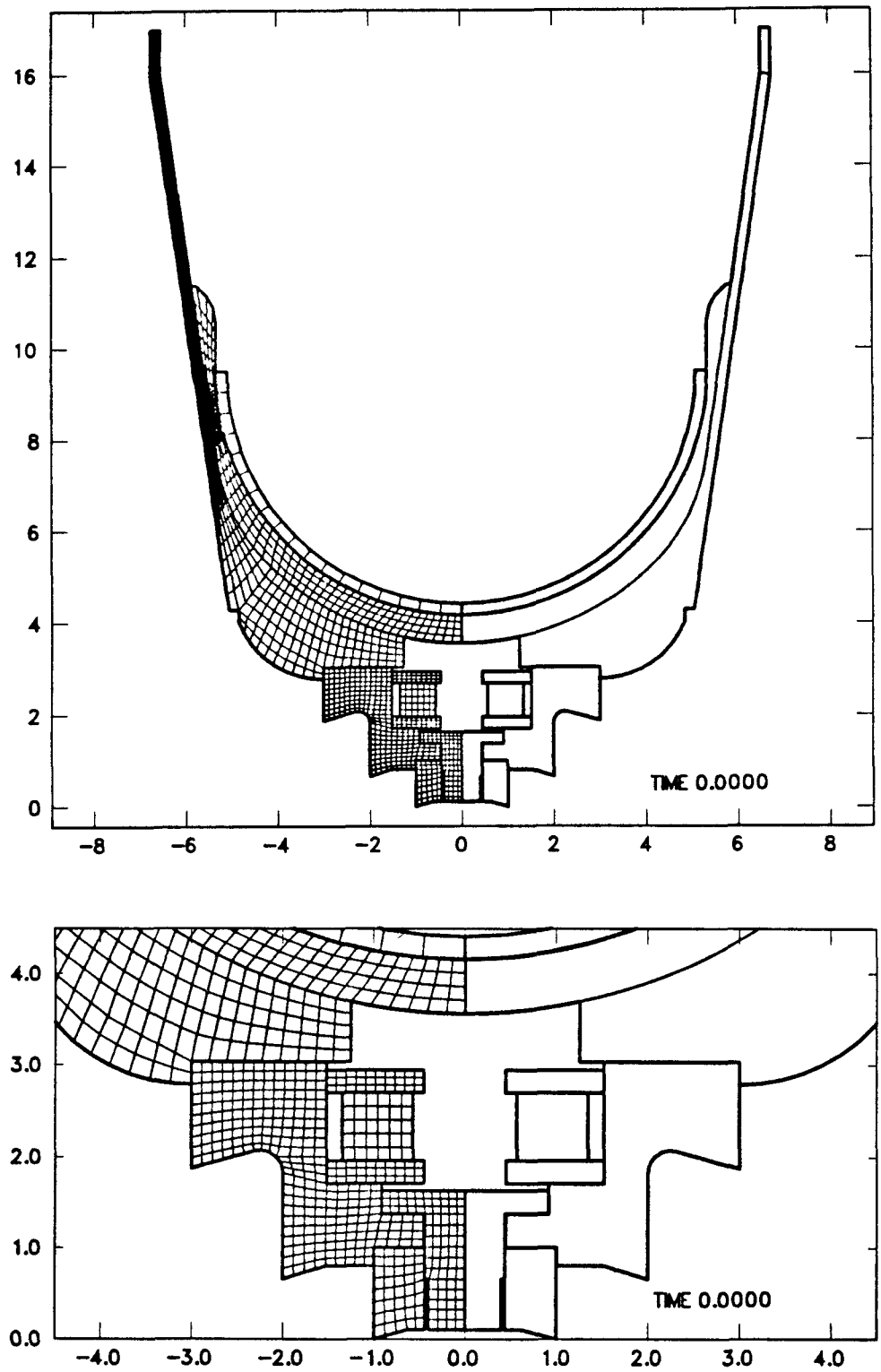


Figure 9.4.1. Undeformed Finite Element Mesh for the Missile Impact Problem

```
TITLE
600FPS/90Deg Hard Impact
AXISYMMETRIC
NO DISPLACEMENT X = 1
TERMINATION TIME = 200 E-6
OUTPUT TIME = 1 0E-6
PLOT TIME = 10 E-6
PLOT NODAL = VEL  REACT
PLOT ELEMENT = PRESSURE, VONMISES
PLOT STATE = EQPS
INIT VEL MAT =  1, 0 , -7200
INIT VEL MAT =  2, 0 , -7200
INIT VEL MAT =  3, 0 , -7200
INIT VEL MAT =  4, 0 , -7200
INIT VEL MAT =  5, 0 , -7200
INIT VEL MAT =  6, 0 , -7200
INIT VEL MAT =  7, 0 , -7200
INIT VEL MAT =  8, 0 , -7200
INIT VEL MAT =  9, 0., -7200
INIT VEL MAT = 10, 0 , -7200
MATERIAL,  1, ELASTIC PLASTIC, 7 32E-4    $ 15-5 PH S S   (Ring Nut, Nose Bolt)
YOUNGS MODULUS = 28 5E+6 , POISSONS RATIO =  27
YIELD STRESS = 170 E+3 , HARDENING MODULUS = 2 85E+6 , BETA = 1
END
MATERIAL,  2, ELASTIC PLASTIC, 7 32E-4    $ 15-5 PH S S   (Nose Cap)
YOUNGS MODULUS = 28 5E+6 , POISSONS RATIO =  27
YIELD STRESS = 170 E+3 , HARDENING MODULUS = 2 85E+6 , BETA = 1
END
MATERIAL   3, ELASTIC PLASTIC, 7 33E-4    $ 4340 Steel  (Forward Case)
YOUNGS MODULUS = 29 0E+6 , POISSONS RATIO =  32
YIELD STRESS = 125 E+3 , HARDENING MODULUS = 2 90E+5 , BETA = 1
END
MATERIAL,  4, ELASTIC PLASTIC, 2 54E-4    $ 6061-T6 Aluminum  (Front Support)
YOUNGS MODULUS = 9 9E+6 , POISSONS RATIO =  33
YIELD STRESS = 35 0E+3 , HARDENING MODULUS = 9 9E+5 , BETA = 1
END
MATERIAL,  5, ELASTIC PLASTIC  7 32E-4    $ 15-5 PH S S   (Impact Sensor Plate)
YOUNGS MODULUS = 28 5E+6 , POISSONS RATIO =  27
YIELD STRESS = 170 E+3 , HARDENING MODULUS = 2 85E+6 , BETA = 1
END
MATERIAL,  6, ELASTIC PLASTIC, 7 21E-3   $ Tungsten Mass  (Impact Sensor)
YOUNGS MODULUS = 58 5E+6 , POISSONS RATIO =  283
YIELD STRESS = 160 E+3 , HARDENING MODULUS = 58 5E+4 , BETA = 1
END
MATERIAL,  7, ELASTIC PLASTIC, 7 32E-4    $ 15-5 PH S S   (1 S  Backing Plate)
YOUNGS MODULUS = 28 5E+6 , POISSONS RATIO =  27
YIELD STRESS = 170 E+3 , HARDENING MODULUS = 2 85E+6 , BETA = 1
END
MATERIAL,  8, ELASTIC PLASTIC, 7 33E-4    $ 4340 Steel  (Explosive Bolt)
YOUNGS MODULUS = 29 0E+6 , POISSONS RATIO =  3
YIELD STRESS = 125 E+3 , HARDENING MODULUS = 29 0E+5 , BETA = 1
END
MATERIAL,  9, ELASTIC, 4 75E-2            $ Equivalent mass elements
YOUNGS MODULUS = 30 0E+6 , POISSONS RATIO =  3
END
MATERIAL, 10, ELASTIC, 2 85E-2            $ Extra material mass
YOUNGS MODULUS = 3 0E+6 , POISSONS RATIO =  3
END
RIGID SURFACE, 11, 0 ,0 ,0.,1 0
RIGID SURFACE, 21, 0 ,0 ,0 ,1 0
RIGID SURFACE  22  0 ,0 ,0 ,1 0
RIGID SURFACE  23  0 ,0 ,0 ,1 0
RIGID SURFACE, 31, 0 ,0 ,0 ,1 0
RIGID SURFACE, 81, 0 ,0 ,0 ,1 0
CONTACT SURFACE   11,  81, 1 0
CONTACT SURFACE   11,  21
CONTACT SURFACE   21,  31, 1 0
CONTACT SURFACE,  21,  51
CONTACT SURFACE,  21,  61
CONTACT SURFACE,  21,  71
CONTACT SURFACE,  21,  81
CONTACT SURFACE,  22,  23
CONTACT SURFACE,  31,  71
CONTACT SURFACE,  41, 101, 0 ,  1 0
CONTACT SURFACE,  51,  81
CONTACT SURFACE,  51,  61
CONTACT SURFACE,  61,  71
EXIT
```

Figure 9.4.2. PRONTO 2D Input for the Missile Impact Problem

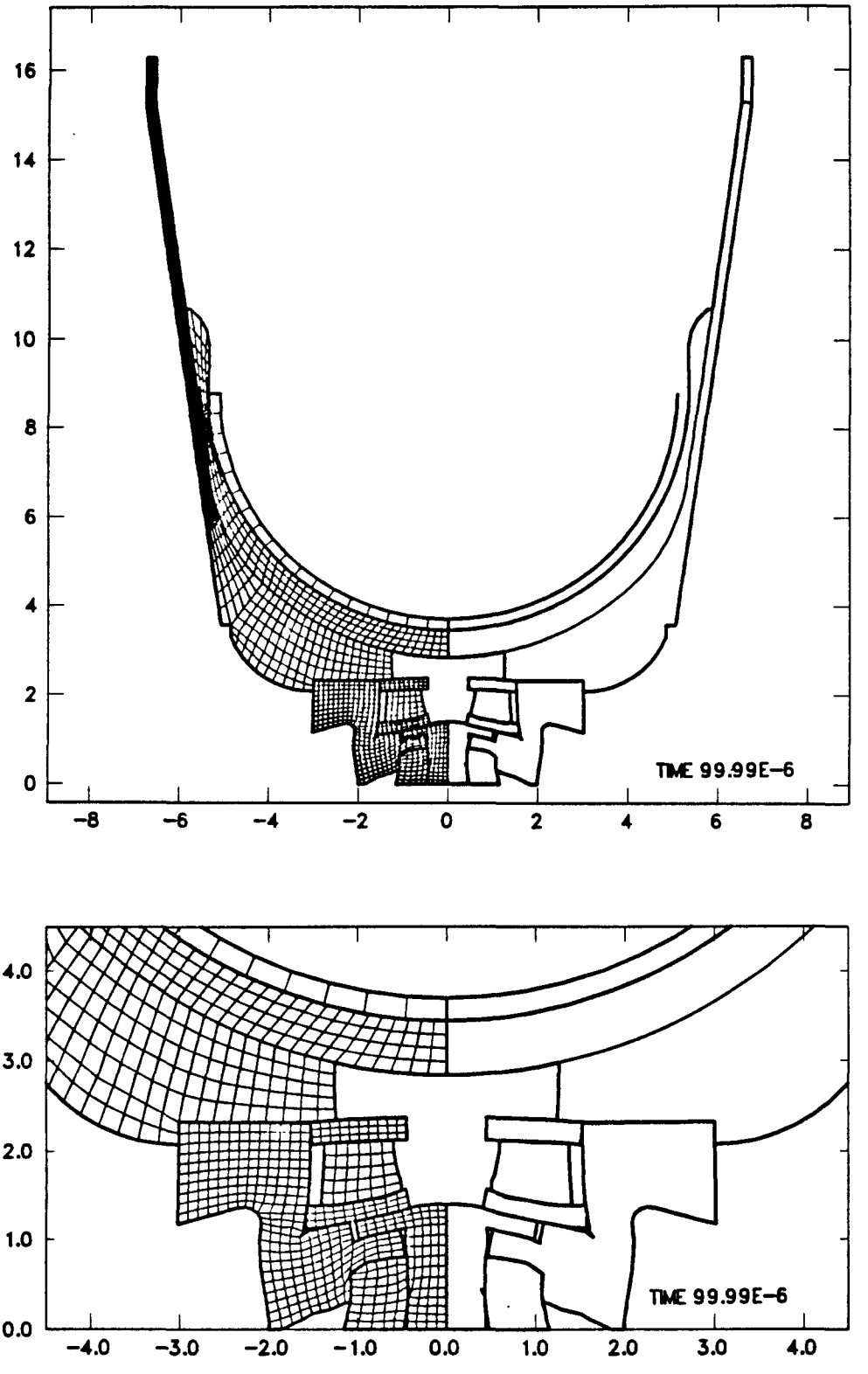


Figure 9.4.3. Deformed Mesh for the Missile Impact at 100 Microseconds

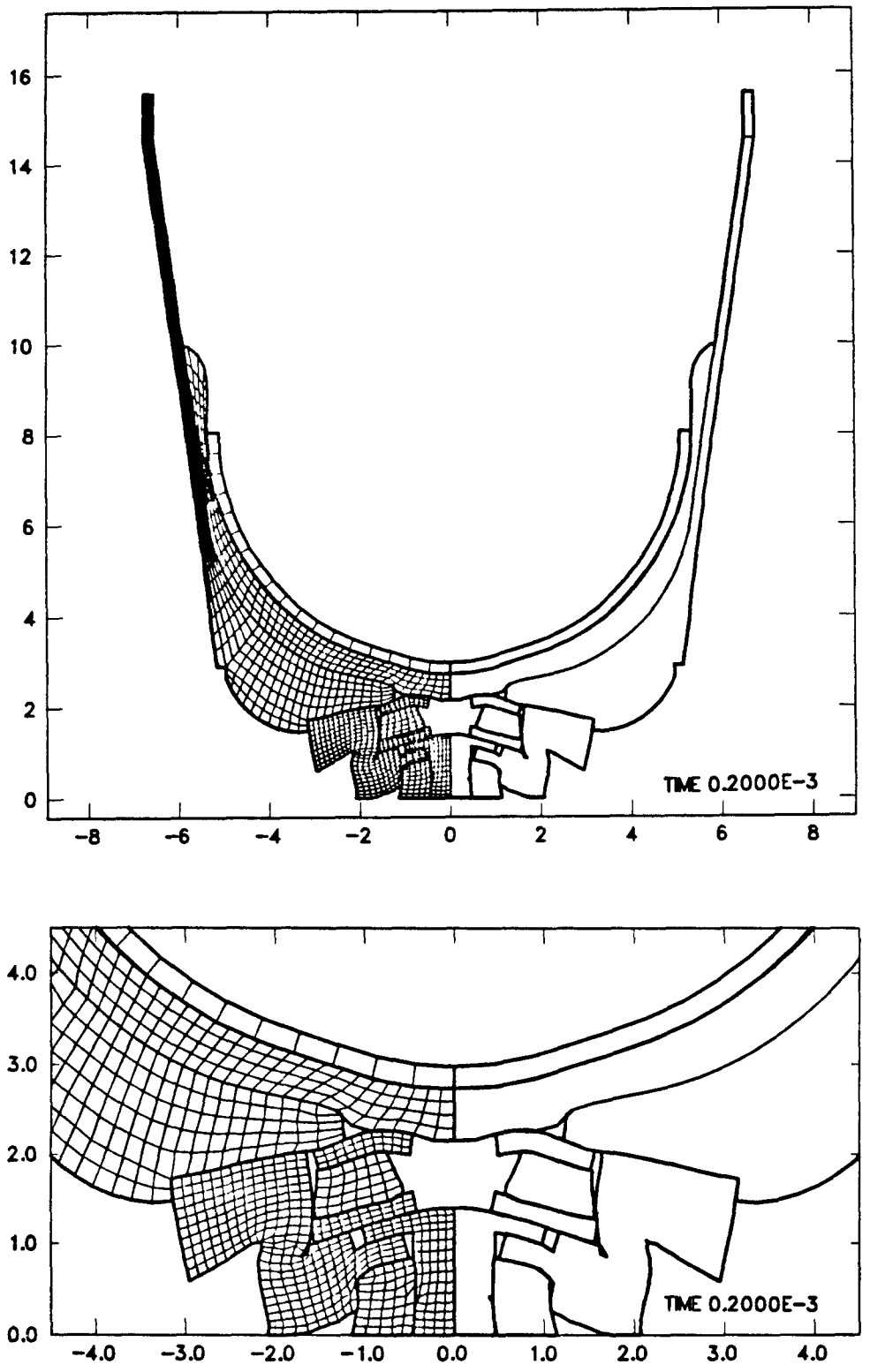


Figure 9.4.4. Deformed Mesh for the Missile Impact at 200 Microseconds

nodes on the dies continue to be applied even though the elements to which they are connected have been deleted from the problem. Likewise, the contact surfaces continue to enforce the contact conditions even though the elements connected to them have been deleted. Since the dies are rigid, we set the contact surfaces to a purely master/slave relationship by setting the bfac parameter (Appendix A.27) to zero on the the contact surface definitions in the input file. Since we were modeling a relatively slow (quasistatic) process, we set the bulk viscosity (Appendix A.14) to a value of .5 to provide a large amount of damping. Figure 9.5.2 shows the input file for this problem. The mesh contains 1663 nodes and 1324 elements. Only the 1200 elements in the copper disk are active (i.e., for these elements constitutive calculations are performed). There are six elements through the thickness in the copper disk.

The analysis was performed in two parts. First, the edge of the copper disk was crimped by moving down the upper right-hand die. This motion occurs in 0.1 seconds of real time, requiring 3968 time steps in PRONTO. The coefficient of friction between the die and the copper was set to 0.2. Figure 9.5.3 shows a closeup of the deformed mesh before and after this was accomplished and after the second step described next was completed. The second step involved drawing the copper disk into a hemispherical cup by moving the punch down 40 mm at a constant velocity of 39.6 mm/sec. The coefficient of friction between the dies and the copper was .08. The total motion occurs in 1.0 seconds of real time requiring an additional 49264 time steps. Two deformed shapes during the motion of the punch are shown in Figure 9.5.4. The total cpu time required for the 53232 total time steps was 1497 seconds on the CRAY/XMP-24 under CTSS.

The load deflection curve for the process is shown in Figure 9.5.5. The initial peak and rebound of the force at a displacement of 2 mm is due to the 39.6 mm/sec impact of the die onto the copper disk. The oscillations which occur in the force after a displacement of 25 mm are caused by the slip of the copper from between the dies used to crimp the edge of the disk (see Figure 9.5.3c). Figure 9.5.6 shows the radial displacement of the node on the far left edge of the copper disk on the centerline. The initial positive radial displacement is due to the initial crimping. Up to a punch

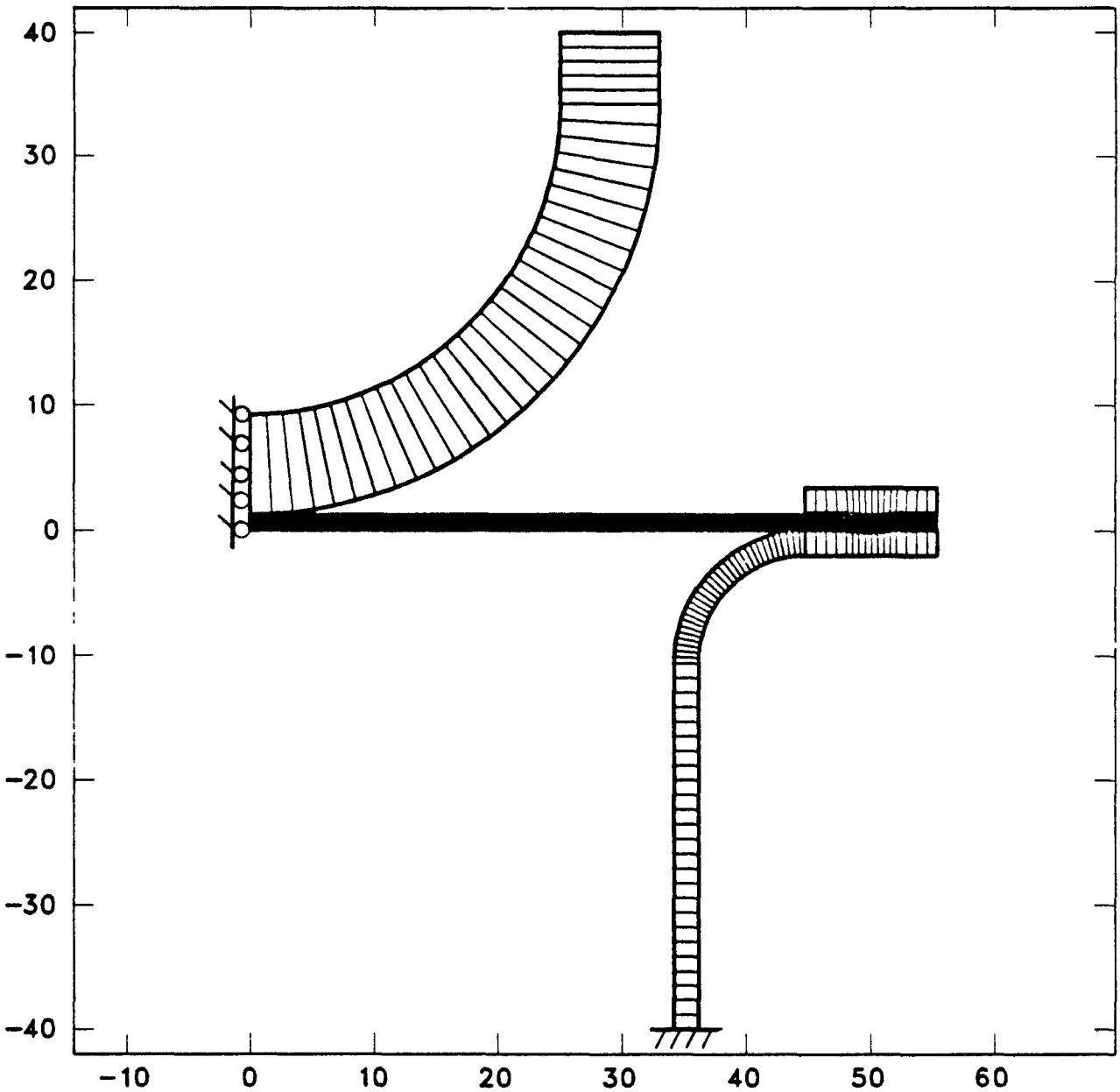


Figure 9.5.1. Undeformed Finite Element Mesh for the Hemispherical Punch Problem

```
TITLE
 DEEP DRAWING  CRIMP FRICT =  2   PUNCH FRICT =  08
AXISYMMETRIC
BULK VISCOSITY= 5.1 2
TERMINATION TIME = 1 1
OUTPUT TIME =  005
PLOT TIME =  050
WRITE RESTART =  1
PLOT NODAL = DISPLACEMENT,REACTION
PLOT STATE = EQPS
PLOT ELEMENT = STRESS,VONMISES,PRESSURE
FUNCTION 1
0  1
 1 0
1 1 0
END
FUNCTION 2
0  0
 1 0
 1 1
1 1 1
END
NO DISPLACEMENT X = 1
NO DISPLACEMENT X = 2
NO DISPLACEMENT X = 3
NO DISPLACEMENT X = 4
NO DISPLACEMENT Y = 4
PRESCRIBED VELOCITY Y = 2 , 1 , -4 002
PRESCRIBED VELOCITY Y = 3 , 2 , -39 6
CONTACT SURFACE = 200 , 100 ,  1 , 0
CONTACT SURFACE = 300 , 100 , 0  , 0
CONTACT SURFACE = 500 , 400 , 0  , 0
CONTACT SURFACE = 600 , 400 ,  1 , 0
MATERIAL 1,ELASTIC,1 0              $ RIGID DIES
YOUNGS MODULUS = 1 0 , POISSONS RATIO = 0
END
DELETE MATERIAL 1 = 0
MATERIAL 2 ELASTIC PLASTIC,9        $ COPPER
YOUNGS MODULUS    = 103 4E6 , POISSONS RATIO =  3
HARDENING MODULUS = 7 E4 , YIELD STRESS = 55 16E3   BETA = 1
END
EXIT
```

Figure 9.5.2. PRONTO 2D Input for the Hemispherical Punch Problem

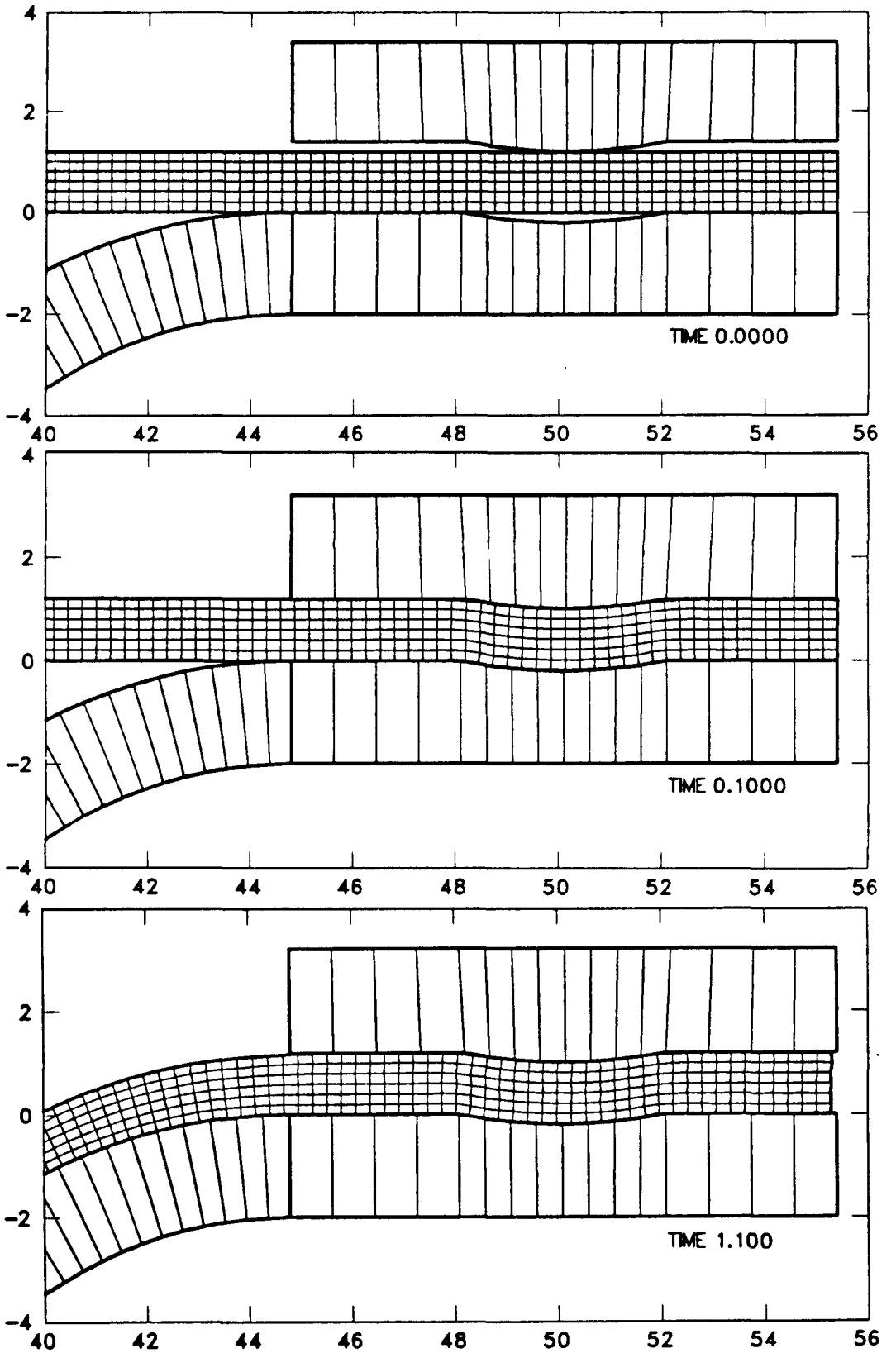


Figure 9.5.3. Close-ups of the Deformed Shapes of the Edge of the Plate at Times 0.0, 0.1, and 1.1 Seconds

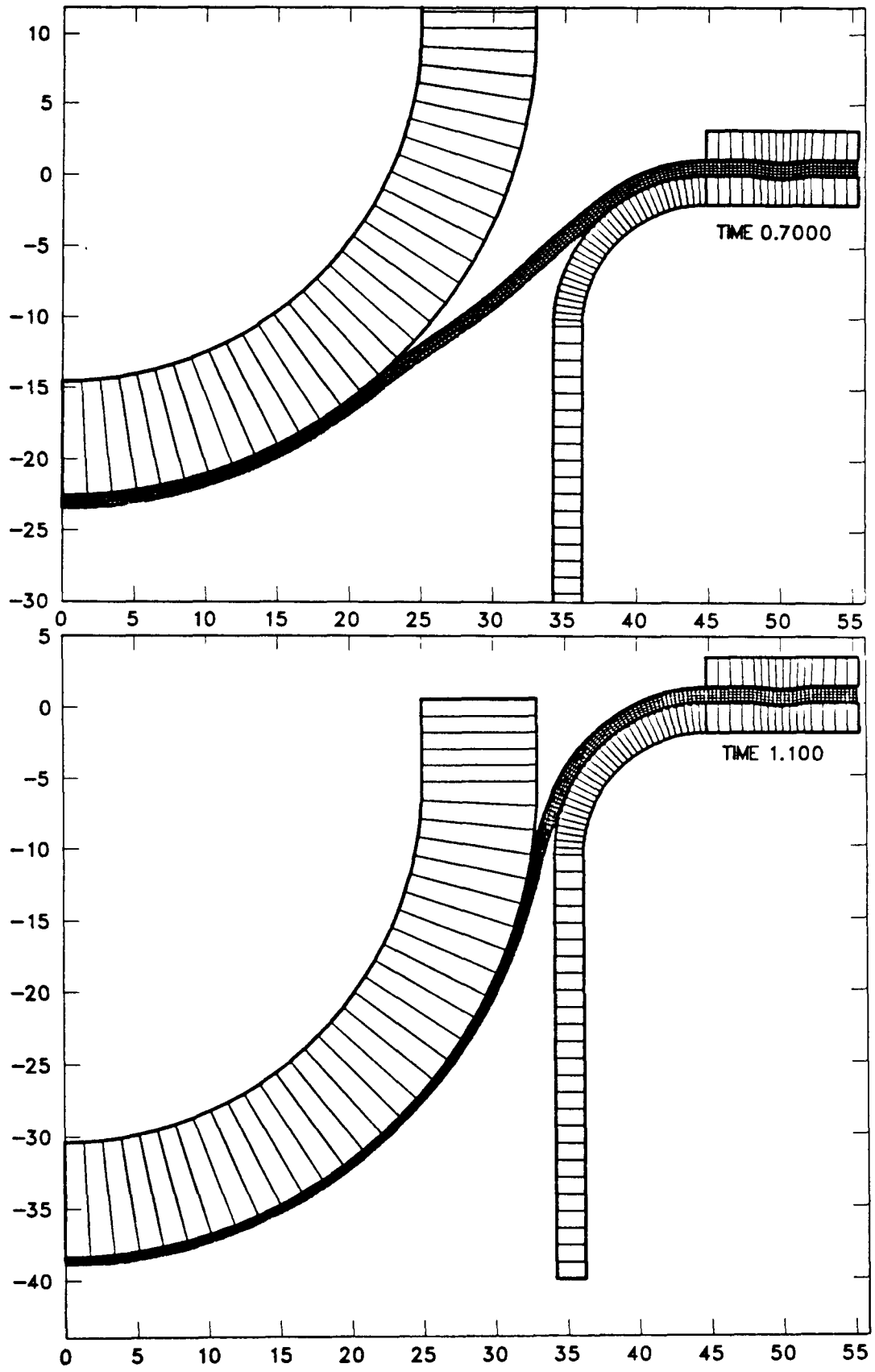


Figure 9.5.4. Deformed Shapes of the Hemispherical Punch at Times 0.7 and 1.1 Seconds; Punch Velocity = 39.6 mm/sec

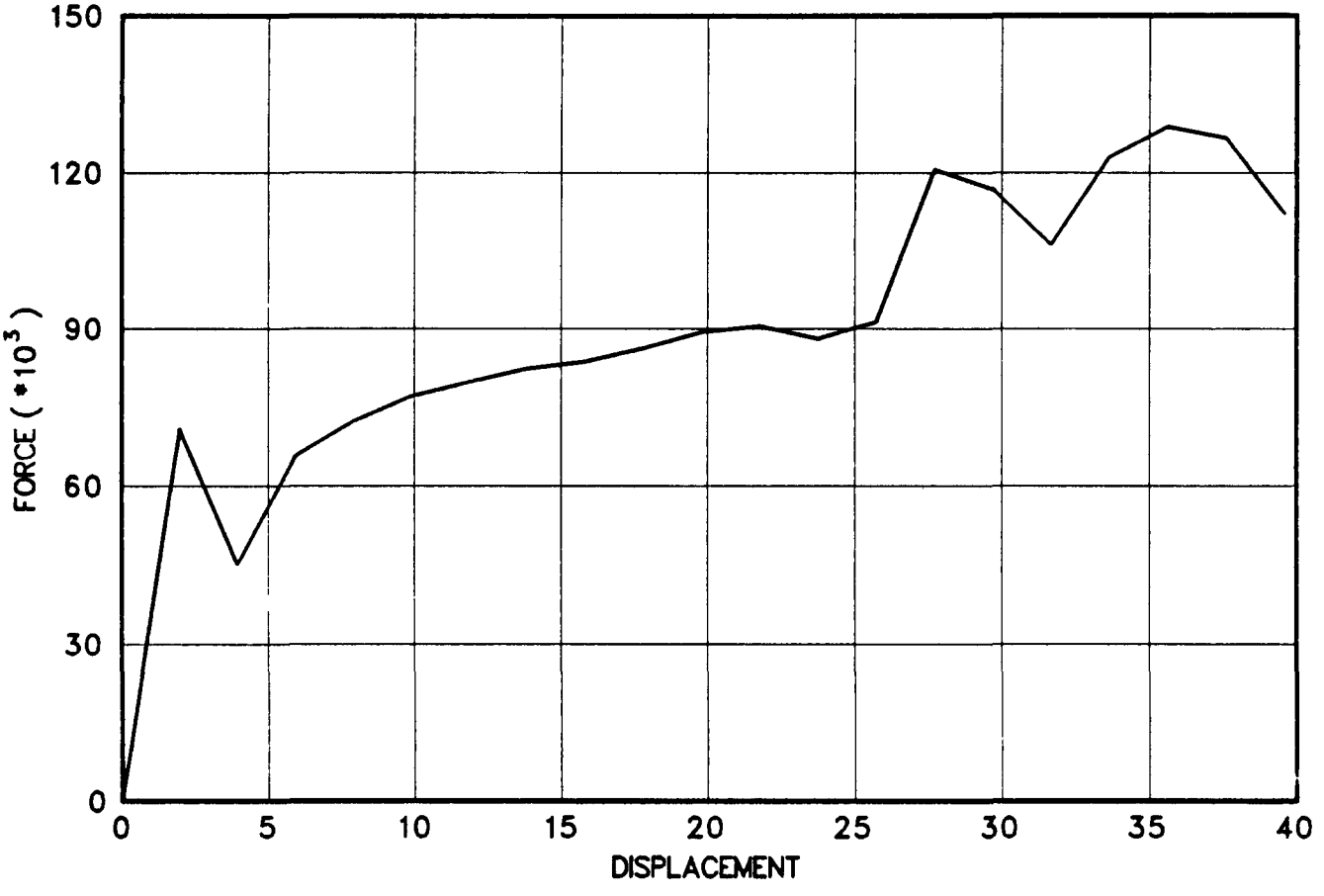


Figure 9.5.5. Force-Displacement Curve for the Hemispherical Punch Problem; Punch Velocity = 39.6 mm/sec

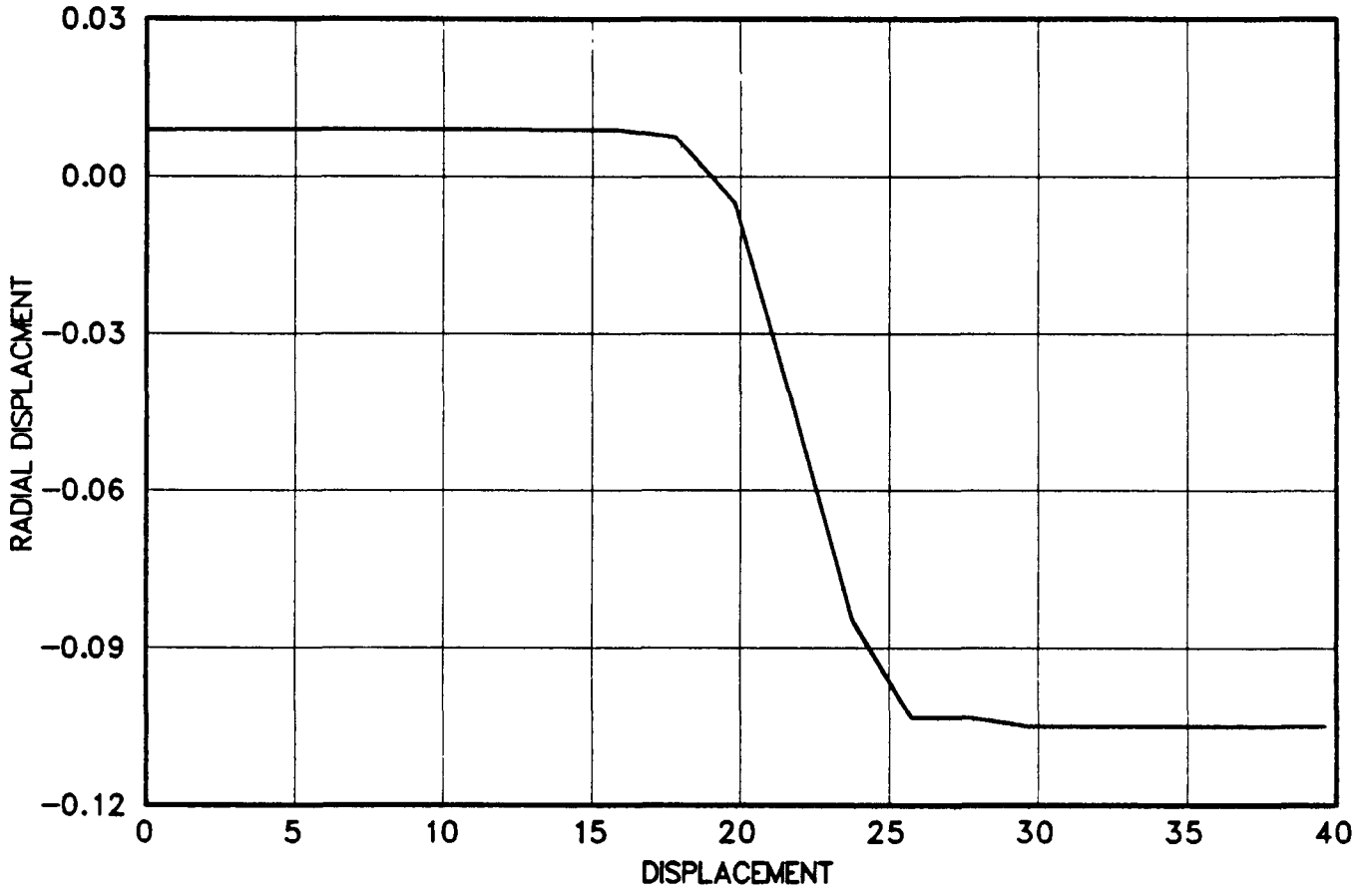


Figure 9.5.6. Punch Displacement Versus Radial Displacement of a Point on the Edge of the Copper Plate; Punch Velocity = 39.6 mm/sec

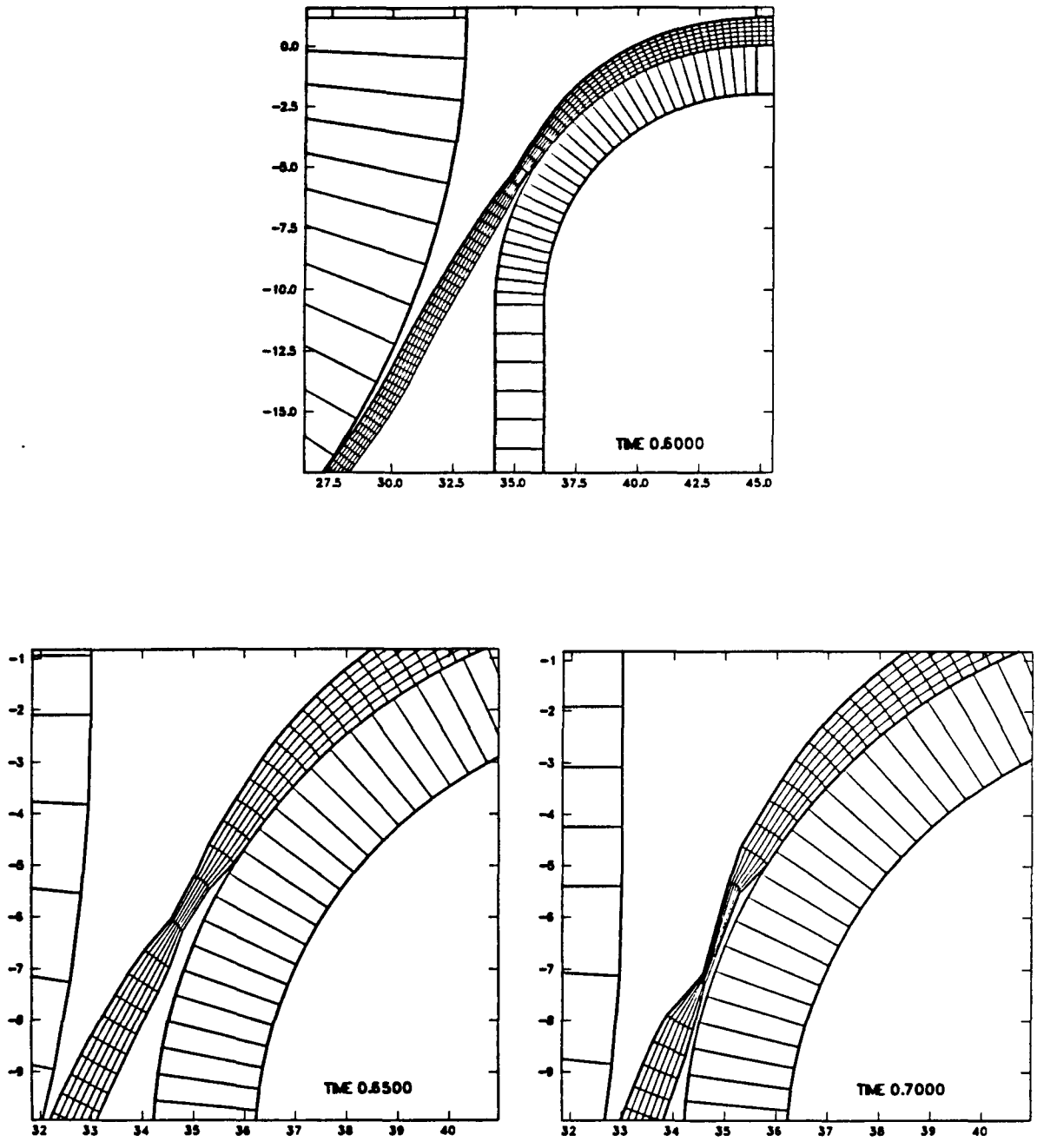


Figure 9.5.7. Formation of a Neck in the Copper Plate at a Punch Velocity of 66 mm/sec

displacement of 20 mm, there is no slippage. At a punch displacement of 20 mm, there is slippage which stops at a punch displacement of 25 mm.

Modeling the problem with different punch velocities can have a dramatic effect on the ability to draw the copper disk into a hemispherical cup. We ran the problem using a punch velocity of 66 mm/sec so that the same punch motion of 40 mm occurs in .6 seconds. With this punch velocity, the copper does not slip as early at the edges, and a neck forms in the copper under the punch. Figure 9.5.7 shows the formation of the neck. The location of the neck is strongly dependent on the rate of drawing.

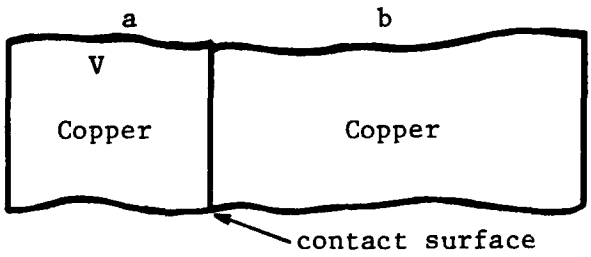
9.6 Impact on Copper Target

This example problem is taken directly from the WONDY [3] manual (example problem 3, page 171). It represents the impact of a copper plate onto a copper target. A plate of thickness .001 m impacts a target of thickness .002 m at a velocity of 210 m/sec. The lateral displacements are constrained resulting in a one-dimensional response. There are 20 elements through the thickness of the plate and 40 through the thickness of the target. There is a contact surface defined between the plate and the target. The geometry defining the problem and the PRONTO instructions for this problem are shown in Figure 9.6.1.

In the WONDY calculation of this problem, the fracture stress was set to a value of 3.5 GPa. We used the adaptive element death option (see Section 3.8, Appendix A.34) to delete elements when the tensile pressure exceeded this value.

The WONDY code uses a higher amount of artificial bulk viscosity than the default values in PRONTO (see Section 3.7). For this problem, we have changed the linear and quadratic bulk viscosity coefficients from their default values to 0.1 and 2.0, respectively (Appendix A.14).

Figure 9.6.2 shows a sequence of shock profiles up to 1.4E-6 seconds. The results very well agree with the WONDY calculations. There is some discrepancy between the results from the two codes late in time (after



a = .001 m (20 elements)

b = .002 m (40 elements)

V = 210 m/sec

```
TITLE
 COPPER PLATE IMPACTING COPPER TARGET (210 M/S)
PLANE STRAIN
BULK VISCOSITY = .1, 2.
TERMINATION TIME = 1.5E-6
OUTPUT TIME = .15E-6
PLOT TIME = .05E-6
PLOT NODAL = VELOCITY
PLOT ELEMENT = PRESSURE,BULKQ
NO DISPLACEMENT Y = 2
INITIAL VELOCITY MATERIAL = 1 , 210 , 0.
CONTACT SURFACE,100,200
DEATH = 1 , PRESSURE, MIN ,-3.5E9
DEATH = 2 , PRESSURE, MIN ,-3.5E9
MATERIAL 1 = HYDRO , 8930.
PRESSURE CUTOFF = -10E9
END
EQUATION OF STATE 1 = MG  US-UP
CO=3940, S=1.489, GAMMA=1.99
END
MATERIAL 2 = HYDRO , 8930.
PRESSURE CUTOFF = -10E9
END
EQUATION OF STATE 2 = MG US-UP
CO=3940, S=1.489, GAMMA=1.99
END
EXIT
```

Figure 9.6.1. Definition of the Copper Target Impact Problem

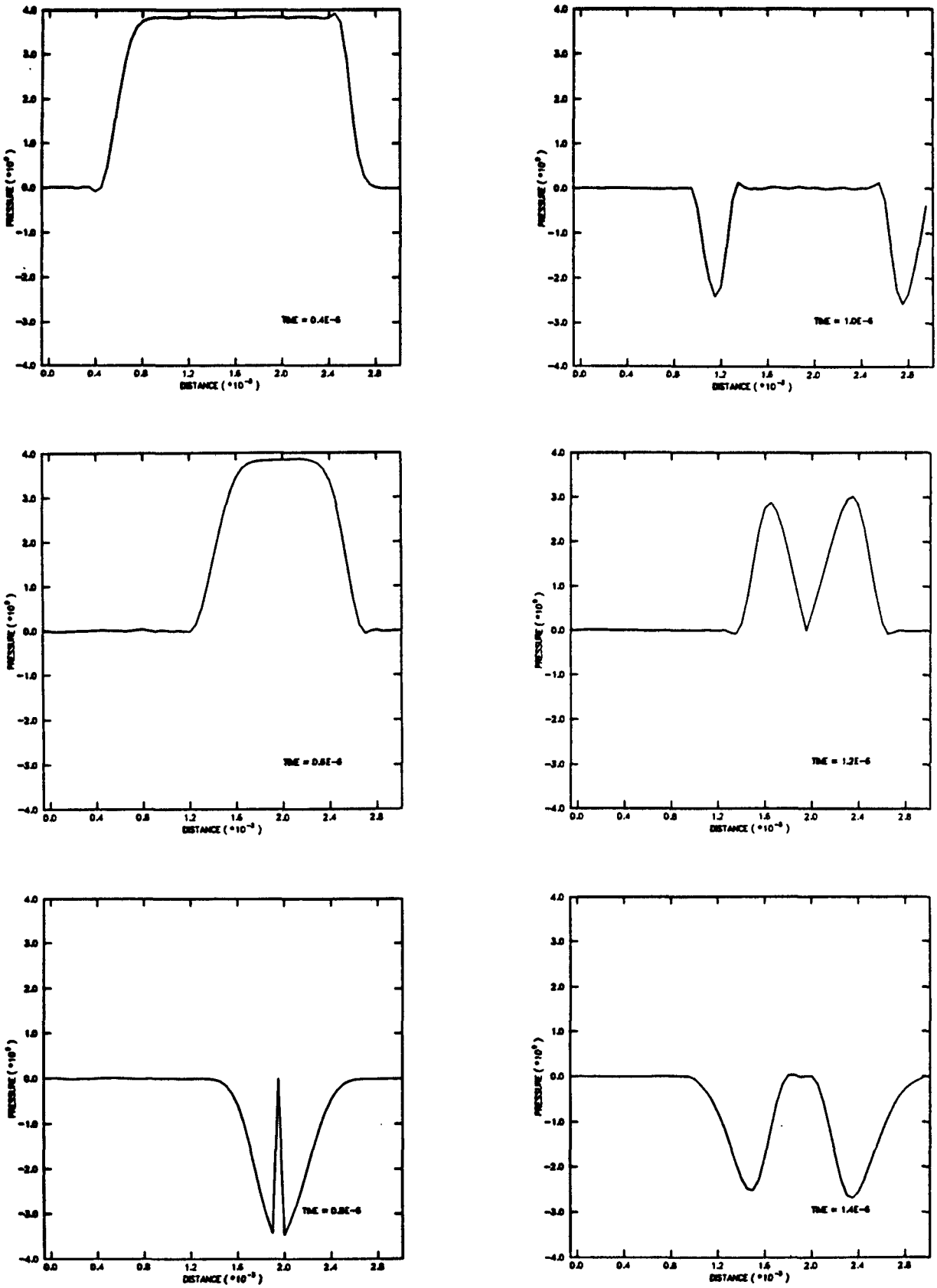


Figure 9.6.2. Calculated Shock Profiles for the Copper Target Impact Problem

1.0e-6 seconds). This is due to the different manner in which the two codes treat the tensile failure. In WONDY, the material is allowed to carry load again in compression after it has failed in tension, while in PRONTO the material is deleted from the mesh and cannot carry load either in tension or compression.

REFERENCES

1. J. von Neumann and R. D. Richtmyer, "A Method for the Numerical Calculation of Hydrodynamic Shocks," J. Appl. Phys., Vol. 21, No. 3, pp 232-237, 1950.

2. W. Herrmann and E. Mack, "WAVE I, A FORTRAN Program for the Calculation of One-Dimensional Wave Propagation," Massachusetts Institute of Technology, ASRL Report No. 1004, March 1962.

3. M. E. Kipp and R. J. Lawrence, "WONDY V - A One-Dimensional Finite-Difference Wave Propagation Code," SAND81-0930, Sandia National Laboratories, Albuquerque, New Mexico, June 1982.

4. J. W. Swegle, "TOODY-IV - A Computer Program for Two-Dimensional Wave Propagation," SAND78-0552, Sandia National Laboratories, Albuquerque, New Mexico, September 1978.

5. M. L. Wilkins, "Calculation of Elastic-Plastic Flow," Methods in Computational Physics, Vol. 3, edited by B. Alder, S. Fernback, and M. Rotenberg, Academic Press, pp 211-263, 1964.

6. S. W. Key, Z. E. Beisinger, and R. D. Krieg, "HONDO II - A Finite Element Computer Program for the Large Deformation Dynamic Response of Axisymmetric Solids," SAND78-0422, Sandia National Laboratories, Albuquerque, New Mexico, October 1978.

7. American National Standard Programming Language FORTRAN, American National Standards Institute, ANXI X3.9-1978, New York, 1978.

8. J. O. Hallquist, "User's Manual for DYNA2D - An Explicit Two-Dimensional Hydrodynamic Finite Element Code with Interactive Rezoning," UCID-18756, Rev. 1, Lawrence Livermore National Laboratory, Livermore, California, February 1982.

9. G. R. Johnson, "EPIC-2, A Computer Program for Elastic-Plastic Impact Computations in 2 Dimensions Plus Spin," ARBRL-CR-00373, Honeywell Inc., Hopkins, Mn., June 1978.

10. L. M. Taylor, D. P. Flanagan, and W. C. Mills-Curran, "The GENESIS Finite Element Mesh File Format," SAND86-0910, Sandia National Laboratories, Albuquerque, New Mexico, May 1986.

11. Z. E. Beisinger, "The SEACO Post Processing Data Base".

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

13. J. K. Dienes, "On the Analysis of Rotation and Stress Rate in Deforming Bodies," Acta Mechanica, Vol. 32, pp 217-232, 1979.

14. G. C. Johnson and D. J. Bammann, "A Discussion of Stress Rates in Finite Deformation Problems," Int. J. Solids Structures, Vol. 20, No. 8, pp 725-737, 1984.

15. D. P. Flanagan and L. M. Taylor, "A Numerical Algorithm for the Accurate Integration of Stress Under Finite Rotations," to appear in Comp. Meth. Appl. Mech. and Eng.

16. C. Truesdell, The Elements of Continuum Mechanics, Springer Verlag, New York, 1966.

17. D. P. Flanagan and T. Belytschko, "A Uniform Strain Hexahedron and Quadrilateral with Orthogonal Hourglass Control," Int. J. Numer. Meth. Eng., Vol. 17, pp 679-706, 1981.

18. G. L. Goudreau and J. O. Hallquist, "Recent Developments in Large Scale Finite Element Hydrocode Technology," J. Comp. Meths. Appl. Mechs. Eng., Vol. 30, 1982.

19. T. J. R. Hughes and J. Winget, "Finite Rotation Effects in Numerical Integration of Rate Constitutive Equations Arising in Large-Deformation Analysis," Int. Jour. for Numerical Methods in Engineering, Vol. 15, No. 12, pp 1862-1867, 1980.

20. D. P. Flanagan and T. Belystchko, "Eigenvalues and Stable Time Steps for the Uniform Strain Hexahedron and Quadrilateral," Jour. Appl. Mech., 84-APM-5, Transactions of the ASME, 1984.

21. R. D. Krieg and D. B. Krieg, "Accuracies of Numerical Solution Methods for the Elastic-Perfectly Plastic Model, ASME Journal Pressure Vessel Technology, Vol. 99, pp 510-515, 1977.

22. H. L. Schreyer, R. F Kulak and J. M. Kramer, "Accurate Numerical Solutions for Elastic-Plastic Models," ASME Journal Pressure Vessel Technology, Vol. 101, pp 226-234, 1979.

23. L. M. Taylor and E. B. Becker, "Some Computational Aspects of Large Deformation, Rate-Dependent Plasticity Problems," Computer Methods in Applied Mechanics and Engineering, Vol. 41, No. 3, pp 251-277, 1983.

24. P. Perzyna, "Fundamental Problems in Viscoplasity," in: Recent Advances in Applied Mechanics, Vol. 9, pp 243-377, Academic Press, New York, 1966.

25. L. M. Taylor, E. P. Chen, and J. S. Kuszmaul, "Microcrack-Induced Damage Accumulation in Brittle Rock Under Dynamic Loading," Computer Methods in Applied Mechanics and Engineering, Vol. 55, No. 3, pp 301-320, May 1986.

26. B. Budiansky and R. J. O'Connell, "Elastic Moduli of a Cracked Solid," International Journal of Solids and Structures, Vol. 12, pp 81-97, 1976.

27. D. Grady, "The Mechanics of Fracture Under High-rate Stress Loading," preprints of the William Prager Symposium on Mechanics of Geomaterials:

Rocks, Concretes and Soils, edited by Z. P. Bazant, Northwestern University, 1983.

28. M. E. Kipp, D. E. Grady, and E. P. Chen, "Strain-rate Dependent Fracture Initiation," International Journal of Fracture, Vol. 16, No. 5, 1980.

29. R. D. Krieg, "A Simple Constitutive Description for Soils and Crushable Foams," SC-DR-72-0883, Sandia National Laboratories, Albuquerque, New Mexico, 1978.

30. D. V. Swenson and L. M. Taylor, "A Finite Element Model for the Analysis of Tailored Pulse Stimulation of Boreholes," International Journal for Numerical and Analytical Methods in Geomechanics, Vol. 7, pp 469-484, 1983.

31. M. K. Neilsen, H. S. Morgan, and R. D. Krieg, "Mechanical Behavior of Low Density Polyurethane Foams," SAND86-2927, Sandia National Laboratories, Albuquerque, New Mexico, in press.

32. D. J. Bammann, "An Internal Variable Model of Viscoplasticity," Int. J. Engng. Sci, Vol 22, pp 1041-1053, 1984.

33. D. J. Bammann and G. J. Johnson, "A Strain Rate Dependent Flow Surface Model of Plasticity," in press.

34. D. J. Bammann and R. D. Krieg, "A Discussion of Unified Creep Plasticity Models," in Unified Constitutive Equations for Plastic Deformation and Creep of Engineering Alloys, ed. A. K. Miller, Pergammon Press.

35. E. Lee, M. Finger, and W. Collins, "JWL Equation of State Coefficients for High Explosives," UCID-16189, Lawrence Livermore National Laboratory, Livermore, California, January 1973.

36. B. M. Dobratz, "LLNL Explosives Handbook, Properties of Chemical Explosives and Explosive Simulants," UCRL-52997, Lawrence Livermore National Laboratory, Livermore, California, March 1981.

37. J. Lysmer and R. L. Kuhlemeyer, "Finite Dynamic Model for Infinite Media," J. Eng. Mechanics Div. ASCE, pp 859-877, August 1979.

38. M. Cohen and P. C. Jennings, "Silent Boundary Methods," in Computational Methods for Transient Analysis, Volume 1, edited by T. Belytschko and T. J. R. Hughes, North-Holland, 1983.

39. D. B. Longcope and S. W. Key, "On the Verification of Large Deformation Inelastic Dynamic Calculations Through Experimental Comparisons and Analytic Solutions," PVP-PB-023, American Society of Mechanical Engineers, 1977.

40. Y. C. Fung, Foundations of Solid Mechanics, Prentice-Hall, NJ, 1965.

41. K. J. Bathe and E. L. Wilson, Numerical Methods in Finite Element Analysis, Prentice-Hall, NJ, 1976.

APPENDIX A
PRONTO 2D USERS MANUAL

Listed below are all the keywords used in the PRONTO 2D input. They are listed in the order they appear in the text below.

1. TITle
2. PLAne STRain
3. AXIsymmetric
4. TERMination TIme
5. OUTput TIme
6. PLOT TIme
7. REAd RESTart
8. WRIte RESTart
9. TIme STep SCale
10. EXIT
11. PLOT NODal
12. PLOT ELement
13. PLOT STate
14. BULK VISCosity
15. HOURglass STIFfening
16. FUNCtion
17. NO DISplacement
18. PRESCribed VELocity
19. PRESCribed ACCeleration
20. PRESCribed FORce
21. INITial VELocity NODeset
22. INITial VELocity MATerial
23. INITial VELocity ANGular
24. PRESsure
25. MOVing PREssure
26. SILent BC
27. CONtact SURface
28. RIGid SURface
29. MATerial
30. EQuation OF STate
31. DETonation POint
32. BURN CONstant
33. MATerial POint
34. DEATh
35. DELete MATerial

The input data to PRONTO is a free field form using keywords. The keywords are intended to define a user friendly program language input. The input is order independent and can be entered in any order the user finds convenient. Words typed below in UPPER CASE represent keywords in the list above. Most of the words can be abbreviated to the first few characters. In the list above, the upper case characters indicate the shortest abbreviation allowed. The words typed in lower case below indicate variables for which the user should enter a value.

The free field input allows the user to delineate entries by either a blank, a comma, or an equals sign. We find it useful to use blanks with commands (keywords), equal signs to separate keywords and/or lists, and commas for

lists of values. The material data requires material cues and their associated values and equal signs are useful there. See the example input below.

A dollar sign indicates that whatever follows on the line of input is a comment and is ignored. An asterisk indicates that the current input line is to be continued on the next line.

1. ***** TITLE
   (enter a suitable title on the next line)

2. ***** PLANE STRAIN (default) (Sec 3.1.1)

3. ***** AXISYMMETRIC (Sec 3.1.2)

4. ***** TERMINATION TIME, tend
   - tend — time to terminate the analysis

5. ***** OUTPUT TIME, tout
   - tout — time interval at which to print output (default = tend/200, where tend is from the TERMINATION TIME, command 4 above)

6. ***** PLOT TIME, tplot, tstart, tpend
   - tplot — time interval at which to write plotting data base (default = (tpend-tstart)/10)
   - tstart — time to start writing data on plotting data base (default = 0.)
   - tpend — time to stop writing data on plotting data base (default = tend from TERMINATION TIME line)

7. ***** READ RESTART, restm (Append F)
   - restm — time at which restart is to begin

8. ***** WRITE RESTART, trsdmp (Append F)
   - trsdmp — time interval at which to write restart dump files (default is to write no restart files)

9. ***** TIME STEP SCALE, scft, ssft (Sec 3.5)
   - scft — scale factor to be applied to the internally calculated time step (default = 1.0)
   - ssft — scale factor to be applied to the internally calculated time step if

strain softening occurs (default = 1.0)
See Section 3.5.

10. ***** EXIT (required to terminate the input data)

11. ***** PLOT NODAL, nodal name 1, nodal name 2, ...
allowable nodal variable names:

DISPLACEMENT	- nodal displacements (DISPLX,DISPLY)
VELOCITY	- nodal velocities (VELX,VELY)
ACCELERATION	- nodal accelerations (ACCLX,ACCLY)
MASS	- nodal lumped masses (MASS)
REACTION	- nodal reactions (RX,RY)

The default nodal variables written on the plotting data base are the displacements, velocities and accelerations. The MASS specification results in having the lumped nodal masses written on the data base. The user always gets the displacements whether he asks for them or not.

The names in parenthesis indicate the alphanumeric name of the variables which are written on the plotting data base. The default element variables are the stresses and the energies.

12. ***** PLOT ELEMENT, element variable 1, element variable 2, ...
allowable element variable names:

STRESS	- stresses (SIGXX,SIGYY,SIGZZ,TAUXY)
ENERGY	- internal energy per unit volume (ENERGY)
STRAIN	- total strains (EPSXX,EPSYY,EPSZZ,EPSXY)
RATEDFM	- deformation rates (DXX,DYY,DZZ,DXY)
STRETCH	- material stretches: V of F = V R (STRECHXX,STRECHYY,STRECHZZ,STRECHXY)
ROTATION	- material rotations: R of F = V R (COSTHETA,SINTHETA)
DENSITY	- current mass per unit volume (DENSITY)
PRESSURE	- pressures (PRESSURE)
VONMISES	- Von Mises equivalent stress (VONMISES)
HG	- hourglass resistance forces (HGX,HGY)
BULKQ	- fraction of pressure due to bulk viscosity (BULKQ)

The names in parenthesis indicate the alphanumeric name of the variables which are written on the plotting data base. The default element variables are the stresses and the energies.

13. ***** PLOT STATE, state variable 1, state variable 2, ...

The user can ask for any of the internal state variables to be written on the plotting data base. Since all materials do not have the same internal state variables (some have none), a zero will be written on the data base for an element using a material model that does not have a state variable which is specified by the user. Hence, if the user

asks for EQPS (equivalent plastic strain) and ALPHA11,ALHA22,ALPHA33, and ALPHA12 (back stress components for kinematic hardening) and he has a model where half the mesh uses the ELASTIC material and half the mesh uses the ELASTIC PLASTIC material, much of the data written on the plotting data base will contain zeros. The table below gives the internal state variables names for all the current material models. See the theory section for definitions of the variables if they are not obvious.

The default state variables are none.

WARNING: Indiscriminate use of this option can create extremely large plotting data bases.

MATERIAL	ALLOWABLE NAMES						
ELASTIC	=	(no internal state variables)					
ELASTIC PLASTIC	= EQPS	ALPHA11	ALPHA22	ALPHA33	ALPHA12	RADIUS	
VISCOPLASTIC	= EQPS	SIGYLD					
DAMAGE	= EQPS	DAMAGE	EVMAX	FRAGSIZE	CRKDENS		
HYDRO	=	(no internal state variables)					
LOW DENSITY FOAM	=	(no internal state variables)					
SOIL N FOAMS	= EVMAX	EVFRAC	EV	NUM			
EP TEMP DEPEND	= EQPS	ALPHA11	ALPHA22	ALPHA33	ALPHA12	RADIUS	TEMP
EP HYDRODYNAMIC	= EQPS	ALPHA11	ALPHA22	ALPHA33	ALPHA12	RADIUS	

14. ***** BULK VISCOSITY, b1, b2 (Sec 3.7)

b1	linear bulk viscosity coefficient (default = .06)
b2	quadratic bulk viscosity coefficient (default = 1.2)

15. ***** HOURGLASS STIFFENING, hgstiff, hgvis (Sec 3.6)

hgstiff	hourglass stiffening factor (default = .05 for plane strain and .01 for axisymmetric)
hgvis	hourglass viscosity factor (default = .0 for plane strain and .03 for axisymmetric)

16. ***** FUNCTION, function id

function id	any nonzero number you wish to identify with this function; after a FUNCTION statement you must enter a list of points defining your function:

```
x1,f(x1)
x2,f(x2)
 ., .
 ., .
xn,f(xn)
END
```

(The list is terminated by a line containing the word
END as shown. Any other valid input cue will also work)

If the function represents a time history function or a function of distance and the value of time or distance is not within the limits defined by x1 and xn, no boundary condition will be applied until the current value falls within the limits. This means that you can have a boundary condition turn on at a specific time or distance and turn off at a specific time or distance.

17. ***** NO DISPLACEMENT, direction, node set id (Sec 7.1.1)

direction	either X or Y
node set id	identifying number from the input data base which identifies the nodes you want to have no displacement (note this is a nodal bc!)

18. ***** PRESCRIBED VELOCITY, dir, node set id, function id, scale factor, a0, b0 (Sec 7.1.2)

dir	either X, Y, RADIAL, TANGENT, or NORMAL
node set id	identifying number from the input data base which identifies the nodes you want to have this velocity (note this is a nodal bc!)
function id	identifying number of the function you want to use to specify the time dependence of the velocity
scale factor	scale factor to be applied to the function (default = 1.0)
a0,b0	not used if direction = X or Y center of cylinder of sphere if direction = RADIAL or TANGENT components of normal (if direction = NORMAL)

19. ***** PRESCRIBED ACCELERATION, direction, node set id, function id, scale factor (Sec 7.1.3)

direction	either X or Y
node set id	identifying number from the input data base which identifies the nodes you want to have this acceleration (note this is a nodal bc!)
function id	identifying number of the function you want to use to specify the time dependence of the acceleration
scale factor	scale factor to be applied to the function (default = 1.0)

20. ***** PRESCRIBED FORCE, direction, node set id, function id, scale factor, a0, b0 (Sec 7.2.3)

direction	either X, Y, RADIAL, TANGENT, or NORMAL
node set id	identifying number from the input data base which identifies the nodes you want to have this force (note this is a nodal bc!)
function id	identifying number of the function you want

	to use to prescribe the time dependence of the force
scale factor	scale factor which will be applied to the function (default = 1.0)
a0,b0	not used if direction = X or Y center of cylinder of sphere if direction = RADIAL or TANGENT components of normal if direction = NORMAL

21. ***** INITIAL VELOCITY NODESET, node set id, x-velocity, y-velocity

node set id	identifying number from the input data base which identifies the nodes you want to have this initial velocity (note this is a nodal bc!)
x-velocity	initial velocity in the x direction
y-velocity	initial velocity in the y direction

22. ***** INITIAL VELOCITY MATERIAL, material no, x-velocity, y-velocity

material no	material number of the material to receive this initial velocity
x-velocity	initial velocity in the x direction
y-velocity	initial velocity in the y direction

23. ***** INITIAL VELOCITY ANGULAR , material no, omega, x0, y0

material no	material number of the material to receive this initial angular velocity
omega	initial angular velocity in radians per second
x0,y0	coordinates of point which the body is spinning about

24. ***** PRESSURE, side set id, function id, scale factor (Sec 7.2.1)

side set id	identifying number from the input data base which identifies the sides you want to have this pressure (note this is a side bc!)
function id	identifying number of the function you want to use to prescribe the time dependence of the pressure
scale factor	scale factor which will be applied to the function (default = 1.0)

25. ***** MOVING PRESSURE, side set id, x0, yo, function #1 id, (Sec 7.2.2)
function #2 id, wave speed, t0,
scale factor

side set id	identifying number from the input data base which identifies the sides you want to have this pressure (note this is a side bc!)
x0, y0	position of point from which pressure propagates
function #1 id	identifying number of the function you want to use to describe the peak pressure as a function of distance from the position (x0,y0)

function #2 id	identifying number of the function you want to use to describe the pressure rise time as a function of distance from the position (x0,y0)
wave speed	propagation speed of the wave along the surface away from the point (x0,y0)
t0	time at which the wave begins to propagate (default = 0.0)
scale factor	scale factor which will be applied to function number 1 to scale the peak pressures (default = 1.0)

26. ***** SILENT BC, side set id (Sec 7.3)

side set id	identifying number from the input data base which identifies element sides which have the nonreflecting boundary condition (note this is a side bc!)

27. ***** CONTACT SURFACE, side set id 1, side set id 2, mu0, pfac, mu1, gamma (Sec 6.2)

side set id 1	identifying number from the input data base which identifies sides on one of the surfaces to be in contact (note this is a side bc!)
side set id 2	identifying number from the input data base which identifies sides on the other surface to be in contact (note this is a side bc!)
mu0	static coefficient of friction (default = 0.)
pfac	partition factor (default = .5)
mu1	high velocity coeffient of friction (default = 0.)
gamma	velocity decay factor

The partition factor is a relative weighting of the master slave relationship of the two surfaces. A value of zero implies that the first surface (defined by side set id 1) acts only as a master and the second surface acts only as a slave. A value of one reverses these roles. The default value (0.5) gives a totally symmetric treatment of the contact. If one surface is much more massive than the other, this variable should be adjusted so that it is treated as a master. By more massive, we mean that the surface either has a higher material density and/or a coarser mesh refinement.

28. ***** RIGID SURFACE, slave id, x0, y0, nx, ny, mu0, mu1, gamma (Sec 6.1)

slave id	identifying number from the input data base which identifies sides that are slaved to the rigid surface (note this is a side bc!)
x0,y0	coordinates of a point on the rigid surface
nx,ny	outward unit normal to the rigid surface
mu0	static coefficient of friction (default = 0.)
mu1	high velocity coefficient of friction (default = 0.)
gamma	velocity decay factor (default = 0.)

29. ***** MATERIAL, material id, material name, density (Ch 4)

material id — material identification number from the input data base

material name — valid material type name, the current material types allowed in PRONTO are:

- ELASTIC
- ELASTIC PLASTIC
- VISCOPLASTIC
- DAMAGE
- SOIL N FOAMS
- LOW DEN FOAM
- HYDRO
- EP TEMP DEPEND
- EP HYDRODYNAMIC

density — material density

Appropriate material data for the given material name are entered here. An END statement is required to terminate the data for each material entered. Each material type requires different material cues. The material data can be entered in any order separated by commas.

Currently, the allowable material names and their required material constants are:

1. ELASTIC (number of cues=2) (Sec 4.1)
   material cues: YOUNGS MODULUS
   POISSONS RATIO

2. ELASTIC PLASTIC (number of cues=5) (Sec 4.2)
   material cues: YOUNGS MODULUS
   POISSONS RATIO
   YIELD STRESS
   HARDENING MODULUS
   BETA

3. VISCOPLASTIC (number of cues=6) (Sec 4.3)
   material cues: YOUNGS MODULUS
   POISSONS RATIO
   YIELD STRESS
   HARDENING MODULUS
   GAMMA
   P

4. DAMAGE (number of cues=6) (Sec 4.4)
   material cues: YOUNGS MODULUS
   POISSONS RATIO
   YIELD STRESS
   M
   K
   FRACTURE TOUGHNESS

5. SOIL N FOAMS (number of cues=7) (Sec 4.5)
   material cues: TWO MU

BULK MODULUS
A0
A1
A2
PRESSURE CUTOFF (negative for tension)
FUNCTION ID (if the function id is zero, the original bulk modulus is used; otherwise this is the function which gives yield stress as a form of pressure)

6. LOW DEN FOAM (number of cues=7) (Sec 4.6)
material cues: YOUNGS MODULUS
A
B
C
NAIR
P0
PHI

7. HYDRO (number of cues=1) (Sec 4.7)
material cues: PRESSURE CUTOFF (Note: negative for tension) (a valid equation of state must be defined which corresponds to this material number)

8. EP TEMP DEPEND (number of cues=17) (Sec 4.8)
material cues: YOUNGS MODULUS
POISSONS RATIO
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
C11
C12
BETA
RHOCV = ρC_v
TEMP = initial temperature

9. EP HYDRODYNAMIC (number of cues=16) (Sec 4.9)
material cues: YOUNGS MODULUS
POISSONS RATIO
YIELD STRESS
HARDENING MODULUS
BETA
PRESSURE CUTOFF (Note: negative for tension) (A valid equation of state must be defined for this material)

Examples for the ELASTIC PLASTIC material are given below to illustrate how the user might input the data in different forms. All three examples are identical as far as PRONTO is concerned.

```
Example 1.
  MATERIAL,1,ELASTIC PLASTIC,2.7E-3
  HARDENING MODULUS = 30.E4
  YOUNGS MODULUS = 30.E6
  BETA = .5
  POISSONS RATIO = .3
  YIELD STRESS = 30.E3
  END
Example 2.
  MATERIAL,1,ELASTIC PLASTIC,2.7E-3
  YOUNGS MODULUS = 30.E6  POISSONS RATIO = .3  BETA =.5
  YIELD STRESS = 30.E3  HARDENING MODULUS = 30.E4
  END
Example 3.
  MATERIAL,1,ELASTIC PLASTIC,2.7E-3
  YOUNGS MODULUS = 30.E6  POISSONS RATIO    = .3     BETA = .5
  YIELD STRESS   = 30.E3  HARDENING MODULUS = 30.E4  END
```

30. ***** EQUATION OF STATE, material id, equation of state name (Ch 5)

material id	material identification number from the input data base
equation of state name	valid equation of state name, the current equations of state defined in PRONTO are: MG US-UP MG POWER SERIES JWL IDEAL GAS

Appropriate material data for the given equation of state name is entered here. An END statement is required to terminate the material data. Each equation of state type requires different material cues. The material data can be entered in any order separated by commas.

Currently, the allowable equation of state names and their required material constants are:

1. MG US-UP (number of cues=3) (Sec 5.2.1)
   material cues: C0
   S
   GAMMA

2. MG POWER SERIES (number of cues=4) (Sec 5.2.2)
   material cues: K0
   K1
   K2
   GAMMA

3. JWL (NUMBER OF CUES=7) (Sec 5.2.4)

material cues: CD
A
B
OMEGA
R1
R2
ENERGY

4. IDEAL GAS (NUMBER OF CUES=2) (Sec 5.2.3)

material cues: GAMMA
SOUND SPEED

An example for the MG US-UP equation of state is given below. Note that the MATERIAL card using the HYDRO material name is shown also and that the material number on both the HYDRO and the EQUATION OF STATE card matches.

Example:

```
MATERIAL,8,HYDRO,2.7E-3
PRESSURE CUTOFF=-1.E9        (note that the pressure is negative
END                                in tension!)
EQUATION OF STATE,8,MG US-UP
C0=5380 S=1.337 GAMMA=2
END
```

31. ***** DETONATION POINT, material no, x0, y0, t0 (Sec 5.2.4)

material no	material number of high explosive to be detonated
x0,y0	coordinates of the detonation point
t0	detonation time (default = 0.)

32. ***** BURN CONSTANT, bs (Sec 5.2.4)

bs	high explosive burn constant (default = 2.5)

33. ***** MATERIAL POINT, x, y

x,y	coordinates of a material point which will be monitored and printed at the tout intervals.

34. ***** DEATH, material id, variable name, mode, level (Sec 3.8)

material id	material identification number
variable name	either ENERGY, VONMISES, PRESSURE, SIGMAX or one of the state variables given in the table defined above for defining the plotting data base (see 13.)
mode	either MIN, MAX, or ABSolute
level	value at which death occurs

Care must be taken to avoid deleting elements which have side boundary conditions applied to them.

The element adaptive death capability requires a very mature user who understands how his material behaves. The capability built into the code is quite general and allows the element to be deleted depending upon the level of energy, vonmises stress, pressure, maximum principal stress or any of the internal state variables for the material type. The mode of comparison given on the line of data (MIN, MAX, or ABS) determines how the comparison is made. For example, the line

```
DEATH = 3 , DAMAGE , MAX , .8
```

would delete elements in the material with id number 3 in which the damage exceeds a value of 0.8. Note that the code will check to see if that material is a damage material and print a fatal error message if it is not. The MIN or ABSolute specification will check whether the value of the variable is less than the level specified or whether the absolute value of the variable exceeds the level specified, respectively. The user should be aware that it is possible to define nonsensical data by using a mode specification which is inappropriate to the variable name. An example of this would be using the MIN specification with the VONMISES variable and a negative value of the level.

If this option is being used, the element array STATUS is automatically written on the plotting data base. This array contains a zero if the element is ALIVE and a one if it is DEAD.

35. ***** DELETE MATERIAL, material id, deletion time

material id	material identification number
deletion time	time at which all elements made up of this material should be deleted from the mesh.

If this option is being used, the element array STATUS is automatically written on the plotting data base. This array contains a zero if the element is ALIVE and a one if it is DEAD.

APPENDIX B
STORAGE ALLOCATION FOR PRONTO 2D

1.0 DIMENSIONING PARAMETERS AND VARIABLES

PARAMETERS

Name	Description
MFIELD	maximum number of fields per line of free field input = 22
NASYMM	number of components in an antisymmetric tensor = 1
NELNS	number of nodes in an element = 4
NHGM	number of hourglass modes per coordinate = 1
NSPC	number of spatial coordinate components = 2
NSYMM	number of components in a symmetric tensor = 4

VARIABLES

Name	Description
MCONES	maximum number of equation of state constants
MCONS	maximum number of material constants
NACCBC	number of prescribed acceleration boundary conditions
NANGV	number of specifications of initial angular velocity
NCONT	number of contact surfaces
NDEATH	number of specifications of adaptive element death
NDETPT	number of detonation points
NEMBLK	number of materials
NFORCE	number of prescribed nodal point force boundary conditions
NFUNC	number of functions
NIVFLG	number of specifications of initial velocity by node sets
NIVMAT	number of specifications of initial velocity by materials
NMATPT	number of material points
NMPBC	number of moving pressure boundary conditions
NNOD	number of nodes
NODISP	number of no displacement boundary conditions
NPRBC	number of pressure boundary conditions
NQUIET	number of nonreflecting or silent boundary conditions
NRIGID	number of rigid surfaces
NRTOT	total number of nodes on all rigid surfaces
NTOTSN	total number of nodes on all contact surfaces
NTOTSV	total number of internal state variables for all elements
NUMEL	number of nodes
NVELBC	number of prescribed velocity boundary conditions

2.0 NODAL POINT VARIABLES

Array	Dimension	Description
COORD	(NNOD,NSPC)	Original nodal point coordinates
CUR	(NNOD,NSPC)	Current nodal coordinates

DISPL	(NNOD,NSPC)	Nodal displacements
VEL	(NNOD,NSPC)	Nodal velocities
ACCL	(NNOD,NSPC)	Nodal accelerations
FORCE	(NNOD,NSPC)	Nodal forces
XMASS	(NNOD)	Nodal point lumped mass

3.0 ELEMENT VARIABLES

Array	Dimension	Description
SIG	(NSYMM,NUMEL)	Element stresses (Note: the number of entries in a symmetric tensor, NSYMM, is four in the 2D case) (1,N) = sigma XX (2,N) = sigma YY (3,N) = sigma ZZ (4,N) = tau XY
HGR	(NSPC,NHGM,NUMEL)	Element hourglass control (Note: the number of hourglass modes, NHGM, is one in the 2D case) (1,1,N) = X hourglass resistance (2,1,N) = Y hourglass resistance
ELMASS	(NUMEL)	Element masses
LINK	(NELNS,NUMEL)	Element connectivity (Note: the number of element nodes, NELNS, is four in the 2D case) (1,N) = node 1 of element (2,N) = node 2 of element (3,N) = node 3 of element (4,N) = node 4 of element
STRECH	(NSYMM,NUMEL)	Element material stretches (1,N) = stretch in X dir (2,N) = stretch in Y dir (3,N) = stretch in Z dir (4,N) = stretch in X-Y dir
ROTATE	(2,NASYMM,NUMEL)	Element material rotations (Note: the number of entries in an asymmetric tensor, NASYMM, is one in the 2D case) (1,1,N) = cosine theta (2,1,N) = sine theta
RHO	(NUMEL)	Element current densities
ENERGY	(NUMEL)	Element internal energies per unit volume
VISPR	(NUMEL)	Element bulk viscosity pressures

SV	(NTOTSV)	Element internal state variables (Note: NTOTSV is the length of the total storage for all of the internal state variables for all of the material models including the equations of state)

Each material block is allocated a specific portion of the SV array whose structure depends upon the material model. The pointer IPSV locates this portion which is processed as SV(NINSV,NELB), where NINSV is the number of internal state variables per element for this material model and NELB is the number of elements in this material block. IPSV, NINSV, and NELB are defined for each material block within the KONMAT data structure.

Each material block with a material model which references an equation of state is also allocated a specific portion of the SV array for the storage of internal state variables for the equation of state. The structure of this array depends upon the equation of state. The pointer IPESV locates this portion which is processed as SVEOS(NESV,NELB), where NESV is the number of internal state variables per element for this equation of state and NELB is the number of elements in this material block. Note that if NESV is zero the pointer is not used. IPESV, NESV, and NELB are defined for each material block within the KONMAT data structure.

4.0 OPTIONAL ELEMENT VARIABLES

There are some element arrays which are only allocated if the user specifies certain options which require them or the user asks for them on the plotting data base.

Array	Dimension	Description
STRAIN	(NSYMM,NUMEL)	Element strains; the strains are only allocated if the strain flag KSFLG = 1. (1,N) = strain XX (2,N) = strain YY (3,N) = strain ZZ (4,N) = strain XY
DOPT	(NSYMM,NUMEL)	Element strain rates (the D tensor). The strain rates are only allocated if the strain rate flag KSRFLG = 1. Normally, the strain rates are only temporaries stored in the SCREL array. (1,N) = strain rate XX (2,N) = strain rate YY (3,N) = strain rate ZZ (4,N) = strain rate XY
STATUS	(NUMEL)	Element status. The element status is allocated if the status flag KSTAT = 1. The status array contains either a zero or a one; one indicates an element is active and a zero indicates it is inactive. This array will be used for deleting elements.

5.0 MATERIAL DATA

Array	Dimension	Description
KONMAT	(10,NEMBLK)	Element block data. The elements are ordered internally such that all of the elements for each material definition are grouped together. We call these material blocks. The following data structure defines this blocking: (1,N) = material id number (2,N) = material kind (3,N) = starting element number (4,N) = ending element number (5,N) = number elements in block (6,N) = number inter. state var (7,N) = IPSV, pointer into SV array (8,N) = EOS type (if any) (9,N) = number of internal state variables for the EOS (10,N) = IPESV, pointer into SV array
DATMAT	((MCONS+MCONES), (NEMBLK+1))	Material properties data array. Each material block also has an array of material properties. Different material models require different amounts of data to be defined. The material interface subroutine returns the variables MCONS and MCONES which are the maximum number of material constants and equation of state constants that will be required. A column in the DATMAT array is allocated to save the material properties for each material block. We allocate one extra column so that if the user inputs an illegal material or makes some other error on material input, we have someplace to put the data as we make the first and second input pass. (1,N) = PROP(1) (2,N) = PROP(2) (3,N) = . . = . . = . (MCONS-2,N) = PROP(MCONS) (MCONS-1,N) = LAMBDA + TWO MU (MCONS,N) = DENSITY (MCONS+1,N) = EOSDAT(1) (MCONS+2,N) = EOSDAT(2) . . . . (MCONS+MCONES-1,N) = EOSDAT(MCONES)

Note: We increment MCONS by 2 over the values set in the data statements in MATINT to make space for the density and dilatational wave speed for the material.

6.0 OPTIONS ARRAYS

6.1 CONTACT SURFACES

Array	Dimension	Description
KLSURF	(12,NCONT)	Integer array containing data for the contact surfaces (Note: NCONT = number of contact surfaces defined). (1,N) = surface 1 side set flag (2,N) = surface 2 side set flag (3,N) = pointer to surface 1 element list (4,N) = pointer to surface 2 element list (5,N) = pointer to surface 1 side set node list (6,N) = pointer to surface 2 side set node list (7,N) = pointer to surface 1 node list (8,N) = pointer to surface 2 node list (9,N) = number sides in surface 1 list (10,N) = number sides in surface 2 list (11,N) = number nodes in surface 1 (12,N) = number nodes in surface 2
CLSURF	(2,NCONT)	Real array containing data for the contact surface. (1,N) = partition balance factor (2,N) = coefficient of friction (3,N) = high velocity coefficient of friction (4,N) = decay constant
KSLIST	(NTOTSN*2)	This array contains the list of surface nodes and tracted element sides for all contact surfaces. The pointers in positions 7 and 8 in KLSURF above point into the KSLIST array. (IP) = surface node number (IP+NSIDES+1) = tracted element side number
CONDAT	(MCTEMP)	This array provides temporary storage to the CONTAC routine for unit normals and assembly of master surface quantities.

6.2 RIGID SURFACES

Array	Dimension	Description
KRIGID	(3,NRIGID)	Integer array containing data for the rigid surface definitions. (Note: NRIGID = the number of rigid surfaces defined) (1,N) = slave side set id flag (2,N) = pointer to slave list (3,N) = number of nodes in list

RIGID	(7,NRIGID)	Real array containing data for the rigid surface definitions. (1,N) = static coefficient of friction (2,N) = high velocity coefficient of friction (3,N) = velocity decay constant (4,N) = XO (5,N) = YO (6,N) = nx, X normal component (7,N) = ny, Y normal component

6.3 NO DISPLACEMENT BOUNDARY CONDITIONS

Array	Dimension	Description
KDISPL	(4,NODISP)	Integer array containing information defining the no displacement boundary conditions (Note: NODISP = the number of no displacement boundary conditions supplied by the user). (1,N) = node set flag (2,N) = pointer into the IBC array (3,N) = direction specification (x-dir = 1., y-dir = 2.) (4,N) = number of nodes with this bc

6.4 PRESCRIBED VELOCITY BOUNDARY CONDITIONS

Array	Dimension	Description
KPVELL	(5,NVELBC)	Integer array containing information defining the prescribed velocity boundary conditions. (Note: NVELBC = the number of prescribed velocity boundary conditions supplied by the user) (1,N) = node set flag (2,N) = pointer into IBC array (3,N) = function id, changed to function number in TELALL (4,N) = direction specification (x-dir = 1., y-dir = 2.) (5,N) = number of nodes with this bc
PVBC	(4,NVELBC)	Real data array containing floating point information for the prescribed velocity boundary conditions. (1,N) = scale factor (2,N) = a0 (3,N) = b0 (4,N) = velocity at last time step

6.5 PRESCRIBED ACCELERATION BOUNDARY CONDITIONS

Array	Dimension	Description
KPACCL	(5,NACCBC)	Integer data array containing information for the prescribed acceleration boundary conditions. (Note: NACCBC = the number of prescribed acceleration boundary conditions set by the user) (1,N) = node set flag (2,N) = pointer into IBC array (3,N) = function id, changed to function number in TELALL (4,N) = direction specification (x-dir = 1., y-dir = 2.) (5,N) = number of nodes with this bc
PABC	(NACCBC)	Scale factors to apply to the time history function for each acceleration boundary condition.

6.6 PRESCRIBED NODAL FORCE BOUNDARY CONDITIONS

Array	Dimension	Description
KFORCE	(5,NFORCE)	Integer data array containing information pertaining to the prescribed nodal point force boundary conditions defined. (Note: NFORCE = the number of prescribed nodal point force boundary conditions set by the user. (1,N) = node set flag (2,N) = pointer into IBC array (3,N) = function id, changed to function number in TELALL (4,N) = direction specification (x-dir = 1., y-dir = 2.) (5,N) = number of nodes
PFORCE	(3,NFORCE)	Real data array containing information for the prescribed nodal point force boundary conditions (1,N) = scale factor (2,N) = a0 (3,N) = b0

6.7 FUNCTIONS

Array	Dimension	Description
KFDAT	(3,NFUNC)	Integer array containing data describing the function definitions. (Note: NFUNC = the total number of functions defined by the user) Each function has a pointer into the FUNCS array to the data that defines the function (the third

		entry). The number of points in the function is stored the second entry. Note that the FUNCS array has all of the functions stored in it and some may be time history functions, some may be spatial functions, and some may be material constitutive data functions. (1,N) = function id number (2,N) = number of points in function (3,N) = pointer into the FUNCS array
FUNCS	(2,NTOTFV)	Function definitions. Note that the FUNCS array has all of the functions stored in it and some may be time history functions, some may be spatial functions, and some may be material constitutive data functions. (1,N) = abscissa (2,N) = ordinate

6.8 INITIAL VELOCITY NODESET

Array	Dimension	Description
KVELFL	(3,NIVFLG)	Integer array containing data defining the initial velocity by nodeset specifications. (Note: NIVFLG = the number of initial velocity by nodeset specifications defined by the user) (1,N) = node set flag (2,N) = pointer into IBC array (3,N) = number of nodes with this initial condition
VELFL	(2,NIVFLG)	Real data array containing the initial velocities specified by the user. (1,N) = X initial velocity (2,N) = Y initial velocity

6.9 INITIAL VELOCITY MATERIAL

Array	Dimension	Description
KVELM	(NIVMAT)	Material identification numbers of those materials which are to receive initial velocity definitions. (Note: NIVMAT = the number of initial velocity by materials defined by the user)
VELM	(2,NIVFLG)	Real data array containing the initial velocities specified by the user. (1,N) = X initial velocity (2,N) = Y initial velocity

6.10 INITIAL VELOCITY ANGULAR

Array	Dimension	Description
KANGV	(NANGV)	Material identification numbers of those materials which are to receive initial angular velocity definitions. (Note: NANGV = the number of initial angular velocity by materials defined by the user)
ANGVEL	(3,NANGV)	Real data array containing the initial angular velocity and the point the body is spinning about as specified by the user. (1,N) = angular velocity (2,N) = XO (3,N) = YO

6.11 DETONATION POINTS

Array	Dimension	Description
KDETPT	(NDETPT)	Material identification numbers in which each of the detonation point are defined. (Note: NDETPT = total number of detonation points defined by the user)
DETPT	(3,NDETPT)	Real data array containing information defining the detonation points. (1,N) = x coordinate of the detonation point (2,N) = y coordinate of the detonation point (3,N) = detonation time

6.12 PRESSURE BOUNDARY CONDITIONS

Array	Dimension	Description
KPBC	(4,NPRBC)	Integer data array defining the pressure boundary conditions. (Note: NPRBC = the number of pressure boundary conditions specified by the user) (1,N) = sideset id (2,N) = pointer to side list (3,N) = function id (4,N) = number of sides with this bc
PBCDAT	(NPRBC)	Scale factors to be applied to the time history function used with each pressure boundary condition.

6.13 MOVING PRESSURE BOUNDARY CONDITIONS

Array	Dimension	Description
KMPBC	(5,NMPBC)	Integer data array defining each of the moving pressure boundary conditions. (1,N) = side set id (2,N) = pointer to side set list (3,N) = number of sides with this BC (4,N) = function no. 1 id (5,N) = function no. 2 id
PMPBC	(5,NMPBC)	Floating point data array defining each of the moving pressure boundary conditions. (1,N) = x0 (2,N) = y0 (3,N) = scale factor (4,N) = t0 (5,N) = wave speed
XMPBC	(3,NSLIST)	This array is dimensioned so that it is allocated for all side boundary condition side sets. We realize that this is wasteful of storage, but it simplifies the coding considerably. For each side in the side set containing the moving pressure boundary condition, there are three pieces of data: (1,N) = a (2,N) = b (3,N) = td, delay time

6.14 SILENT BOUNDARY CONDITIONS

Array	Dimension	Description
KQUIET	(4,NQUIET)	Integer data array defining each of the silent or nonreflecting boundary conditions. (1,N) = side set id (2,N) = pointer to element list (3,N) = pointer to side list (4,N) = number of sides with this bc

6.15 MATERIAL POINT DEFINITIONS

Array	Dimension	Description
KMPTS	(3,NMATPT)	Integer data array defining each of the material points defined. (Note: NMATPT = the number of material points defined by the user) (1,N) = nearest nodal point number (2,N) = element number containing point (3,N) = material block number

PTSDAT	(2,NMATPT)	Coordinates of the material points (1,N) = X coordinate of point (2,N) = Y coordinate of point

6.16 ELEMENT BLOCK DELETION

Array	Dimension	Description
DELETE	(NEMBLK)	This array contains the time at which each material block of elements is to be deleted from the analysis. The default times are twice the termination time.

6.17 ADAPTIVE ELEMENT DELETION

Array	Dimension	Description
KDEATH	(4,NDEATH)	Integer data array defining the adaptive element deletions. (1,N) = material id (2,N) = material type (3,N) = variable to base death upon (4,N) = mode of death (MIN,MAX, or ABS)
DEATH	(NDEATH)	Value upon which adaptive deletion depends.

7.0 VECTOR BLOCKING ARRAYS

A number of arrays are needed to vectorize the code. These arrays all have one dimension given by the vector blocking factor, NEBLK. It should be noted that the particular order of the dimensions of many of the arrays in PRONTO is such that the GATHER and SCATTER routines can be used. An array called SCREL for SCRatch ELement is dimensioned in the main routine 50 by NEBLK. It is passed into the SOLVE array and used as needed for vector scratch arrays. Maps are provided in the comments in SOLVE which indicate how this space is being managed.

8.0 BOUNDARY CONDITIONS

Arrays of boundary condition data are read from the GENESIS data base. For more information on the GENESIS data base, see Appendix D.

8.1 NODAL BOUNDARY CONDITIONS

Array	Dimension	Description
KFLAGS	(NBCNOD)	This array contains a list of all the node set id's found on the GENESIS file. (Note: NBCNOD = the number of id's found) PRONTO does not necessarily use all the flags found.
NPFLAG	(NBCNOD)	This array contains the number of nodes in each of the nodal sets which have node set id's in the KFLAGS array above. The IBC array below contains the actual list of nodes.
NFLOC	(NBCNOD)	This array contains the pointer into the IBC array to the list of nodes for each flag.
IBC	(NNLIST)	This array contains the list of nodes having nodal boundary conditions. (Note: NNLIST = the total number of all nodes having nodal boundary conditions specified) Some nodes may be repeated in this list because more than one flag was specified on that particular node.
VALNOD	(NNLIST)	This array contains the list of multiplication values to be applied to the boundary condition specification (currently this option is not used but it is anticipated that it will be used to define spatial distributions of nodal boundary conditions).

8.2 SIDE BOUNDARY CONDITIONS

Array	Dimension	Description
NSFLG	(NBCSID)	This array contains a list of all the element side boundary condition id's found on the GENESIS file. (Note: NBCSID = the total number of side boundary condition id's found on the file)
NSLEN	(NBCSID)	This array contains the number of element sides in each side set.
NVLEN	(NBCSID)	This array contains the number of nodes which define the sides in each side set.
NSPTR	(NBCSID)	This array contains the pointer into the NELEMS array where the elements numbers for the side set are located.

NVPTR	(NBCSID)	This array contains a pointer for each side set which locates the nodes associated with this side set relative to a concatenated list in the array NSNODE.
NELEMS	(NSLIST)	This array contains a concatenated list of elements numbers which encompasses all side sets. The element number for each side is provided in order that the analysis code can determine material properties which may be relevant to a particular surface condition.
NSNODE	(2,NSLIST)	This array contains a concatenated list of side nodes which encompasses all side sets. This list is ordered such that the local node index cycles faster than the element side index (all connected nodes for side 1, then all connected nodes for side 2, etc.). The list usually contains repeated node numbers since associated element sides tend to be connected. In PRONTO, we sometimes remove the repeated nodes and repack this array (e.g., when processing slidelines). (1,N) = first node number (2,N) = second node number
SVALUE	(2,NSLIST)	This array contains a concatenated list of nodal distribution factors. The list has a one-to-one correspondence to the NSNODE array above. These distribution factors can be used to prescribe a spatial distribution of a boundary condition. In PRONTO, we do not currently support this capability, but we do use this array in the slideline calculations. (1,N) = factor at first node (2,N) = factor at second node

9.0 SEACO DATA BASE

We allow the user to construct his own plotting output database, but give him a default data base. We allocate some arrays in PRONTO (the main routine) to set up the data base.

9.1 NODAL PLOTTING ARRAYS

Array	Dimension	Description
NODWR	(5)	This array contains zeros or ones and indicates whether a particular nodal quantity is to be written to the SEACO data base. A zero indicates

		that the quantity is not written, a one indicates the quantity is written. (1) = displacement flag (default =1) (2) = velocity flag (default=1) (3) = acceleration flag (default=1) (4) = nodal mass flag (default=0) (5) = reaction flag (default=0)
LISTND	(9)	This array contains the list of variable names to be written on the plotting data base. The default names are set in the first six entries and get reset in subroutine SNLIST if the user changes them. Below, we show the defaults and indicate the last three are defaulted to null. (1) = default = DISPLX (2) = default = DISPLY (3) = default = VELX (4) = default = VELY (5) = default = ACCLX (6) = default = ACCLY (7) = null (8) = null (9) = null

9.2 ELEMENT PLOTTING ARRAYS

Array	Dimension	Description
NELWR	(11)	This array contains zeros or ones and indicates whether a particular element quantity is to be written to the SEACO data base. A zero indicates that the quantity is not written, a one indicates the quantity is written. (1) = stress flag (default =1) (2) = energy flag (default=1) (3) = hourglass flag (default=0) (4) = strain flag (default=0) (5) = strech flag (default=0) (6) = rotation flag (default=0) (7) = ratedfm flag (default=0) (8) = density flag (default=0) (9) = pressure flag (default=0) (10) = vonmises flag (default=0) (11) = bulkq flag (default=0)
LISTEL	(25)	This array contains the list of variable names to be written on the plotting data base. The default names are set in the first six entries and get reset in subroutine SELIST if the user changes them. Below we show the defaults and indicate the last three are defaulted to null. (1) = default = SIGXX (2) = default = SIGYY (3) = default = SIGZZ

(4) = default = TAUXY
(5) = default = ENERGY
(6) = null
(7) = null
. .
. .
(25) = null

9.3 STATE VARIABLE PLOTTING ARRAYS

Array	Dimension	Description
LISTSV	(MFIELD)	This array contains the list of state variable names to be written on the plotting data base. The default names are all null and get reset in subroutine SVLIST if the user asks for any internal state variables. (Note: MFIELD is a parameter set in the parameter statement set and is the maximum number of fields which can be read on one line of input by the free field reader; currently this value is set to 22, which should be sufficient.)
MAPIE	(NEMBLK,NSVLST)	Mapping array for writing internal state variables to the plotting data base. If the user specifies that he wants to write internal state variables on the plotting data base, this mapping is constructed which indicates where the internal state variable resides for each material. If a particular material does not have that internal state variable, a zero is entered into the mapping. This mapping is required because totally different material models may have the same internal state variable (e.g., equivalent plastic strain) but it may not be the first state variable for one material and the last for another. (Note: NSVLST = the number of internal state variables specified by the user.) (1,N) = location of state variable in this material (2,N) = location of state variable in this material (3,N) . . . (NEMBLK,N) = location of state variable in this material

APPENDIX C
ADDING A NEW CONSTITUTIVE MODEL TO PRONTO

PRONTO was designed from the beginning to serve as a testbed for new constitutive models and algorithms. We have incorporated a material interface subroutine which allows you (the constitutive model developer) to add a new material model with very little effort. We have purposely designed this interface so that you do not have to understand the inner workings of the finite element code, especially with respect to the allocation and management of computer memory. If the instructions in subroutine MATINT are followed correctly, the computer program will handle all memory allocation, material data reading, and material data printing. There are three steps that you should follow to add a new model.

STEP 1.

Subroutine MATINT contains instructions in the FORTRAN COMMENT cards which outline the steps that you should follow to add a new material model. Most of the changes required involve adding or changing numbers in DATA and PARAMETER statements. Since we have no foreknowledge of what the material constants represent for a particular material, we require in one of the steps that a few lines of FORTRAN be added which tells the code what the initial dilatational modulus (lambda+two mu) is for the material. This value must be stored in the variable DATMOD in step 12. At the same place in the code, it is possible to calculate any combinations of the input material constants that may be required in the constitutive subroutine (e.g., bulk modulus from Young's modulus and Poisson's ratio).

There is a restriction to twenty characters in the material name, material cues and internal state variable names which are defined in subroutine MATINT. Also, since the names may have blanks (i.e., you may use multiple word cues as in YOUNGS MODULUS) the names must be defined such that each word in the name is unique to the first three characters. This means that material cues C1, C2, C3, etc., are legal; but CON1, CON2, CON3, etc., are not. Finally, please do not use unusual characters in your words as we cannot guarantee the results.

Again, we reiterate, all of the steps that you must take when adding a new material model are outlined in detail with comments in the FORTRAN in subroutine MATINT.

STEP 2.

This step is optional and is only required if the new material model contains internal state variables which must be initialized to some value other than zero (we initialize all internal state variables to zero by default). If state variables must be initialized, you must add an ELSE IF statement to subroutine SVINIT for this material. This statement should read:

```
ELSE IF( MKIND .EQ. (new material no.) ) THEN
                    .
          initialize internal
          state variables here
                    .
                    .
                    .
```

The new material number corresponds to the position where the material resides in the list of materials defined in subroutine MATINT. Generally, when adding a new material, the new material is the last one defined it will be the same as the number of materials defined which is in step 1 in MATINT. This step should be obvious from looking at how other material models are coded in SVINIT. Please use comments so that years from now we have some chance of figuring out what was added to the code.

STEP 3.

In subroutine UPDSTR, the call to the new material subroutine must be added. The material subroutine may have any appropriate name, but we have been naming them MAT1, MAT2, etc., where the number corresponds to the MKIND (see STEP 2) above. The call is included by adding an ELSE IF block to subroutine UPDSTR which should read:

```
ELSE IF( MKIND .EQ. (new material no.) ) THEN
   CALL new subroutine( .... argument list .... )
```

The new material number corresponds to where the material resides in the list of materials defined in subroutine MATINT. Generally, when adding a new material, the new material is the last one defined and the new material number will be the same as the number of materials defined, which is set in step 1 in MATINT. This step should be obvious from looking at how other material models are coded in UPDSTR. Please use comments.

APPENDIX D
GENESIS FILE FORMAT

The GENESIS file format defines the geometry of the problem and is generated by an external preprocessor. PRONTO reads the file in two places. The first two records on the file are read in the MSHDAT routine. The first record contains a title which is not used, but the second record contains the sizing data for the problem. The FORTRAN to read these two records appears as:

```
C
C---- Comment Record (Ignored)
C
      READ(9)
C
C---- Problem Size Data
C
      READ(9) NNOD,NDIM,NUMEL,NEMBLK,NBCNOD,NNLIST,NBCSID,NSLIST,NVLIST
```

After reading the sizing information and allocating space for the appropriate arrays, PRONTO reads the remainder of the GENESIS file in the MSHDAT routine. The element connectivity data is read in blocks of elements according to the material identification numbers of the various materials in the mesh. Note that we depend on the ANSI FORTRAN standard to execute zero trip DO LOOPS correctly. The FORTRAN to read the rest of the file appears as:

```
C
      PARAMETER (NSPC=2,NELNS=4,NESNS=2,NSYMM=4,NASYM=1,NONSYM=5,NHGM=1,
     *           NEBLK=64,MFIELD=22)
      DIMENSION COORD(NNOD,NSPC),LINK(NELNS,NUMEL),KONMAT(10,NEMBLK),
     *  KFLAGS(*),NPFLAG(*),NFLOC(*),IBC(*),VALNOD(*),NSFLG(*),NSLEN(*),
     *  NVLEN(*),NSPTR(*),NVPTR(*),NELEMS(*),NSNODE(*),SVALUE(*)
C
C---- Nodal Coordinates
      READ(9) COORD
C
C---- Element Order Map (We ignore this record in PRONTO)
      READ(9)
C
C---- Element Block Data
C
      IEND = 0
      DO 10 N = 1,NEMBLK
C
C     Element Block Parameters -
         READ(9) MATID,NUMELB,NELNOD,NATRIB
C           MATID  - material id number
C           NUMELB - number of elements in this material block
C           NELNOD - number of nodes in the element (must be equal
C                    to four for the four node quadrilateral)
C           NATRIB - number of element attributes (must be equal to
C                    zero because we have no element attributes)
C
```

```
         ISTRT = IEND + 1
         IEND  = IEND + NUMELB
C
C     Element Connectivity -
         READ(9) ((LINK(J,I),J=1,NELNOD),I=ISTRT,IEND)
C
C     Element Attributes (Ignored in PRONTO) -
         READ(9)
   10 CONTINUE
C
C---- Nodal Boundary Condition Data
C
      READ(9) (KFLAGS(I),I=1,NBCNOD)
      READ(9) (NPFLAG(I),I=1,NBCNOD)
      READ(9) (NFLOC(I),I=1,NBCNOD)
      READ(9) (IBC(I),I=1,NNLIST)
      READ(9) (VALNOD(I),I=1,NNLIST)
C
C---- Side Boundary Condition Data
C
      READ(9) (NSFLG(I),I=1,NBCSID)
      READ(9) (NSLEN(I),I=1,NBCSID)
      READ(9) (NVLEN(I),I=1,NBCSID)
      READ(9) (NSPTR(I),I=1,NBCSID)
      READ(9) (NVPTR(I),I=1,NBCSID)
      READ(9) (NELEMS(I),I=1,NSLIST)
      READ(9) (NSNODE(I),I=1,NVLIST)
      READ(9) (SVALUE(I),I=1,NVLIST)
C
```

APPENDIX E
SEACO FILE FORMAT

The SEACO file format is a standard post processing format adopted by the Engineering Analysis Department at Sandia. PRONTO writes a post processing file which follows this format. The beginning of the file contains information defining the mesh and the names of the nodal, element, and global quantities which are written at regular time intervals. Each time record then contains the actual values of the nodal, element, and global quantities at that specific time.

In PRONTO we construct the lists of nodal and element variables to be written on the file as instructed by the user. These lists of names are stored in the arrays LISTND and LISTEL, respectively. The default nodal variable list gives displacements, velocities, and accelerations. The default element variable list gives stresses and internal energies. The complete default lists are:

COORDINATES	NODAL	ELEMENT	GLOBAL
X	DISPLX	SIGXX	TMSTEP
Y	DISPLY	SIGYY	KE
	VELX	SIGZZ	XMOM
	VELY	TAUXY	YMOM
	ACCLX	ENERGY	
	ACCLY		

Note that the FORTRAN shown here is not exactly as it is written in PRONTO but the result is the same. We have simplified it somewhat to make it easier to read.

The first part of the file is written as:

```
      PARAMETER(NSPC=2,NELNS=4,IDUM=0)
      DIMENSION COORD(NNOD,2),LINK(4,NUMEL),MATID(NUMEL)
      CHARACTER*8 LISTND(NDLIST),LISTEL(NELIST)
      CHARACTER*8 MODIFY,NAMEGB(4),NAMEX(2)
C
      DATA NAMEX/'X','Y'/
      DATA NAMEGB/'TM STEP','KE','XMOM','YMOM'/
      DATA MODIFY/' '/
C
C  Write title and QA information -
      WRITE(11) HEAD,VERSON,RDATE,RTIME,MODIFY,MODIFY,MODIFY
C  Write parameters -
      WRITE(11) NSPC,NNOD,NUMEL,NELNS,NEMBLK,NDLIST,NVAREL,NGLOBL,
     *          IDUM,IDUM,IDUM
C  Write alphanumeric names of the coordinates -
      WRITE(11) NAMEX
C  Write alphanumeric names of the nodal variables -
      WRITE(11) LISTND
```

```
C  Write alphanumeric names of the element variables -
      WRITE(11) LISTEL
C  Write alphanumeric names of the global variables -
      WRITE(11) NAMEGB
C  Write the initial coordinates of the mesh -
      WRITE(11) COORD
C  Write the element connectivity array -
      DO 20 J=1,NUMEL
         WRITE(11) (LINK(I,J),I=1,4)
   20 CONTINUE
C  Write the material identification array (WARNING: if only ONE
C  material id is present in the mesh this record is not written;
C  this has caused a considerable amount of grief in the past)
      IF(NEMBLK.GT.1) WRITE(11) (MATID(I),I=1,NUMEL)
C
```

Each time interval contains records defining the values of the nodal, element, and global variables defined above. Since the user can construct his own list of nodal and element variables, the variables written at each time interval may not be the same as those shown below for the default case. For the default case, the nodal variables are displacements, velocities, and accelerations, and the element variables are stresses and internal energy. For the default case, the SEACO file is written for each time interval as:

```
C
C  Write the current time
      WRITE(11) TIME
C
C  NODAL variables -
C
C  Write the displacements
         WRITE(11) (DISPL(I,1),I=1,NNOD)
         WRITE(11) (DISPL(I,2),I=1,NNOD)
C  Write the velocities
         WRITE(11) (VEL(I,1),I=1,NNOD)
         WRITE(11) (VEL(I,2),I=1,NNOD)
C  Write the accelerations
         WRITE(11) (ACCL(I,1),I=1,NNOD)
         WRITE(11) (ACCL(I,2),I=1,NNOD)
C
C  ELEMENT variables -
C
C  Write the stresses
      DO 10 J = 1,4
         WRITE(11) (SIG(J,I),I=1,NUMEL)
   10 CONTINUE
C  Write the energy per unit volume
      WRITE(11) (ENERGY(I),I=1,NUMEL)
C
C  GLOBAL Variables -
C
      WRITE(11) DT,XKE,XMOM,YMOM
```

APPENDIX F
RESTART FILE FORMAT

The PRONTO restart file contains the data defining the state of the problem at regular intervals in time. The philosophy behind the manner in which we do restarts is to allow the user as much leeway as possible in changing the problem upon restart. He can add or delete slidelines, change algorithmic parameters such as bulk viscosity or hourglass control, or make any changes which are consistent with the mechanics principles inherent in the problem definition which resides on the restart file. Typically, the restart input will be identical to the original PRONTO input deck except for the data line defining the time of restart (e.g., READ RESTART = 2.E-6). Also, the restart run will usually use the same GENESIS data file.

The code will carefully check the restart file to see if it is compatible with the mechanics problem defined by the input data. Most of these checks are relatively simple (e.g., checking to see if there are the same number of elements in the mesh). PRONTO checks very carefully to make sure that the user has not changed the material definitions or properties upon restarting.

The only real complication involved in reading and writing restart files correctly lies with the contact and rigid surfaces. The PRONTO contact algorithms are written such that each node in the contact list can only be in contact with one surface segment at one time. This means that there is at most one tangential nodal force due to friction. Since the friction forces are history dependent, these forces must be written on the restart file. We simply construct a list of all the nodal friction forces, most of which most are zero, and write that list on the restart file. It is then a simple effort to reconstruct the contact data since we can detect all the normal contact conditions from the current configuration and then search the nodal list of tangential forces for the current friction force for that contact. This procedure makes it possible to add and/or delete slidelines upon restarting.

There are always seven records in each restart state:

1. header record
2. material properties record
3. nodal and element state record
4. internal state variables record
5. element status record (null if death option is not used)
6. element densities record (null if element densities are not required)
7. element strains record (null if element strains are not required)

Each record in the restart file is written in the following manner:

```
C
C 1.  Write the header information
      MATX = ( MCONS + MCONES ) * NEMBLK
      ISTATE = 1
      WRITE(30) TIME,DT,NNOD,NUMEL,NEMBLK,NSPC,MATX,KDFLG,KSFLG,KSTAT,
     *          ISTATE,NTOTSV,NSTEPS
C       where:
C         TIME   = current time for the state
C         DT     = current time increment
C         NNOD   = number of nodes in the mesh
C         NUMEL  = number of elements in the mesh
C         NEMBLK = number of materials in the mesh
C         NSPC   = number of spatial coordinate components
C         MATX   = total number of material constants for all materials
C         KDFLG  = flag indicating whether material densities are required
C                  in this problem. KDFLG = 0 indicates no densities are
C                  required and an empty record is written. KDFLG = 1
C                  indicates densities are required and the densities are
C                  written on the file.
C         KSFLG  = flag indicating whether element strains are required
C                  in this problem. KSFLG = 0 indicates no strains are
C                  required and an empty record is written. KSFLG = 1
C                  indicates strains are required and the strains are
C                  written on the file.
C         KSTAT  = flag indicating whether the elements status are required
C                  in this problem. KSTAT = 0 indicates no status is
C                  required and an empty record is written. KSTAT = 1
C                  indicates status is required and the status is written
C                  on the file.
C         ISTATE = 1, indicates the internal state variables are written
C                  on the file.
C         NTOTSV = total number of internal state variables for all the
C                  elements and all the materials
C         NSTEPS = current time step number
C
C 2.  Write the material property arrays
      WRITE(30) ((KONMAT(J,I),J=1,10),I=1,NEMBLK),
     *          (DATMAT(I),I=1,(MCONS+MCONES)*NEMBLK)
C
C 3.  Write the current state
      WRITE(30) ((DISPL(I,J),I=1,NNOD),J=1,NSPC),
     *          ((VEL(I,J),I=1,NNOD),J=1,NSPC),
     *          (ENERGY(I),I=1,NUMEL),
     *          ((STRECH(J,I),I=1,NUMEL),J=1,NSYMM),
     *          ((ROTATE(J,I),I=1,NUMEL),J=1,2),
     *          ((HGR(J,I),I=1,NUMEL),J=1,2),
     *          ((SIG(J,I),I=1,NUMEL),J=1,NSYMM),
     *          (VISPR(I),I=1,NUMEL)
      IDUM = 0
C
```

```
C 4. Write internal state variables
      IF( NTOTSV .NE. 0 ) THEN
         WRITE(30) (SV(I),I=1,NTOTSV)
      ELSE
         WRITE(30) IDUM
      END IF
C
C 5. Write material status array if needed
      IF( KSTAT .NE. 0 ) THEN
         WRITE(30) (STATUS(I),I=1,NUMEL)
      ELSE
         WRITE(30) IDUM
      END IF
C
C 6. Write material density per unit volume if needed
      IF( KDFLG .NE. 0 ) THEN
         WRITE(30) (RHO(I),I=1,NUMEL)
      ELSE
         WRITE(30) IDUM
      END IF
C
C 7. Write element strains if needed
      IF( KSFLG .NE. 0 ) THEN
         WRITE(30) ((STRAIN(J,I),I=1,NUMEL),J=1,NSYMM)
      ELSE
         WRITE(30) IDUM
      END IF
```

Distribution:

Dr. R. T. Allen
Pacifica Technology
P.O. Box 148
Del Mar, CA 92014

Prof. S. Atluri
Center for the Advancement of
Computational Mechanics
School of Civil Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Dr. William E. Bachrach
Areojet Research Propulsion Inst.
P. O. Box 13502
Sacramento, CA 95853-4502

Prof. E. B. Becker
Department of Aerospace Engineering
and Engineering Mechanics
University of Texas
Austin, TX 78712

Prof. T. Belytschko
Department of Civil Engineering
Northwestern University
Evanston, IL 60201

Mr. Chuck Charman
GA Technologies
P.O. Box 81608
San Diego, CA 92138

Mr. Dwight Clark
Morton Thiokol Corp.
P. O. Box 524
Mail Stop 281
Brigham City, UT 84302

Mr. Gerald Collingwood
Morton Thiokol
Huntsville, AL 35807

Mr. Bill Cook
Los Alamos National Laboratory
Los Alamos, NM 87545

Dr. R. S. Dunham
Anatech International Corp.
3344 N. Torrey Pines Ct.
Suite 320
La Jolla, CA 92307

Dr. Arlo F. Fossum
RE/SPEC, Inc.
P.O. Box 725
Rapid City, SD 57709

Dr. Francisco Guerra
Mail Stop C931
WX11 Division
Los Alamos National Laboratory
Los Alamos, NM 87545

Dr. Gerry Goudreau
Methods Development Group
Mechanical Engineering Department
Lawrence Livermore National Lab
Livermore, CA 94550

Dr. John Hallquist
Methods Development Group
Mechanical Engineering Department
Lawrence Livermore National Lab
Livermore, CA 94550

Dr. David Hibbitt
Hibbitt, Karlsson & Sorrensen, Inc.
100 Medway St.
Providence, RI 02906

Mr. Jeffery P. Hill
Mail Stop F644
Group CTR-4
Los Alamos National Laboratory
Los Alamos, NM 87545

Mr. Roger Hill
Mail Stop D449
P15 Division
Los Alamos National Laboratory
Los Alamos, NM 87545

Mr. John Hopson
Group T3
Mail Stop B216
Los Alamos National Laboratory
Los Alamos, NM 87545

Dr. William Hufferd
United Technologies
Chemical Systems Division
P.O. Box 50015
San Jose, CA 95150-0015

Prof. T. J. R. Hughes
Department of Mechanical Engineering
Stanford University
Palo Alto, CA 94306

Dr. Rembert Jones and (2)
Dr. Alan S. Kushner
Office of Naval Research
Structural Mechanics Div. (Code 434)
800 N. Quincy Street
Arlington, VA 22217

Prof. George C. Johnson
Mechanical Engineering Dept.
University of California
6127 Etcheverry Hall
Berkeley, CA 94720

Dr. Gordon R. Johnson
Honeywell Inc.
5901 S. County Rd. 18
Edina, MN 55436

Mr. James Johnson
Rm L120, CPC Engineering Center
30003 Van Dyke
Warren, MI 48090

Prof. J. K. Lee
Department of Engineering Mechanics
Ohio State University
Columbus, OH 43210

Mr. Richard Lung
TRW Corporation
P. O. Box 1310
Bldg 527, Rm 709
San Bernadino, CA 91763

Mr. J. J. Murphy (5)
Vehicle Technology 59-22,
Bldg 580
Lockheed Missiles and Space Co.
P. O. Box 3504
Sunnyvale, CA 94088

Prof. V. D. Murty
5000 N. Willamette Blvd.
School of Engineering
University of Portland
Portland, OR 97203

Dr. Joop Nagtegaal
Marc Analysis Research Corp.
260 Sheridan Ave
Suite 200
Palo Alto, CA 94306

Prof. S. Nemat-Nasser
Department of Applied Mechanics
and Engineering Sciences
University of California
San Diego
La Jolla, CA 92093

Dr. R. E. Nickell
c/o Anatech International Corp.
3344 N. Torrey Pines Court
Suite 320
La Jolla, CA 92037

Mr. Dean Norman
Waterways Experiment Station
P.O. Box 631
Vicksburg, MS 39180

Prof. J. T. Oden
Department of Aerospace Engineering
and Engineering Mechanics
University of Texas
Austin, TX 78712

Dr. Robert Pardue
Martin Marietta
Y-12 Plant, Bldg. 9998
Mail Stop 2
Oak Ridge, TN 37831

Mr. Mitchell R. Phillabaum
Monsanto Research Corp.
MRC-MOUND
Miamisburg, OH 45342

Dr. Joe Rashid
Anatech International Corp.
3344 N. Torrey Pines Ct.
Suite 320
La Jolla, CA 92307

Dr. Timothy J. Ross
Air Force Weapons Laboratory
Civil Engineering Research Division
Kirtland AFB, NM 87117

Mr. Donald W. Sandidge
U.S. Army Missile Command
AMSMI-RLA
Redstone Arsenal, AZ 35898-5247

Prof. H. L. Schreyer
New Mexico Engineering Research Inst.
Campus P. O. Box 25
University of New Mexico
Albuquerque, NM 87131

Prof. M. Stern
Department of Aerospace Engineering
and Engineering Mechanics
University of Texas
Austin, TX 78712

Mr. Ray Stoudt
Lawrence Livermore National Lab
P.O. Box 808, L200
Livermore, CA 94550

Prof. D. V. Swenson
Mechanical Engineering Dept.
Durland Hall
Kansas State University
Manhattan, KS 66506

Mr. Sing C. Tang
P. O. Box 2053
Rm 3039 Scientific Lab
Dearborn, MI 48121-2053

Mr. David Wade, 36E
Bettis Atomic Power Laboratory
P.O. Box 79
West Mifflan, PA 15122

Dr. Paul T. Wang
Fabricating Technology Division
Aluminum Company of America
Alcoa Technical Center
Alcoa Center, PA 15069

Prof. Kaspar Willam
Civil, Environmental, and
Architectural Engineering
Campus Box 428
University of Colorado
Boulder, CO 80309-0428

Ms. Emily Young
K-Tech Corporation
901 Pennsylvania, N.E.
Albuquerque, NM 87110

1510	J. W. Nunziato
1511	D. K. Gartling
1520	W. Herrmann, Actg.
1521	S. N. Burchett
1521	R. D. Krieg
1521	J. D. Miller
1521	G. D. Sjaardema
1521	C. M. Stone
1521	G. W. Wellman
1522	R. C. Reuter
1522	B. J. Kipp
1522	K. W. Schuler
1522	P. P. Stirbis
1523	J. H. Biffle
1523	E. P. Chen
1523	D. P. Flanagan (25)
1523	L. M. Taylor (25)
1524	K. W. Gwinn
1524	A. K. Miller
1530	L. W. Davison
1531	S. L. Thompson
1531	J. W. Swegle
1533	M. E. Kipp
1533	S. T. Montgomery
1533	P. Yarrington
1534	J. R. Asay
1540	W. C. Luth
1542	W. R. Wawersik
1550	R. C. Maydew
3141-1	S. A. Landenberger (5)
3151	W. L. Garner (3)
3154-1	C. H. Dalin (28) for DOE/OSTI
6253	N. R. Warpinski
6258	P. J. Hommert
6258	J. S. Kuszmaul
8024	P. W. Dean
9113	J. T. Schamaun
9122	M. J. Forrestal
9122	M. H. Gubbels