Modeling of elastic-viscoplastic behavior and its finite element implementation

Ted Diehl
I, Ted Diehl, do hereby grant Wallace Memorial Library permission to reproduce my thesis in whole or in part. Any reproduction by Wallace Memorial Library will not be used for commercial use or profit.
Abstract

A state variable approach is developed to simulate the isothermal quasi-static mechanical behavior of elastic-viscoplastic materials subject to small deformations. Modeling of monotonic/cyclic loading, strain-rate effect, work hardening, creep, and stress relaxation are investigated. Development of the constitutive equations is based upon Hooke's law, the separation of the total strain into elastic and plastic quantities, and the separation of work hardening into isotropic and kinematic quantities. The formulation consists of three coupled differential equations; a power law measuring viscoplastic strain-rate and two first order equations for isotropic and kinematic hardening. Derivation of, behavior of, and use of the model are discussed. Actual material data from uniaxial monotonic and cyclic tests is simulated numerically. The formulation, excluding kinematic hardening, is also expanded into multiple dimensions and the compression of a cylinder with constrained ends is solved using the finite element method.
Acknowledgments

Immeasurable gratitude is owed to my advisor Dr. Hany Ghoneim for his insight, guidance, boundless encouragement, and warm friendship.

I would also like to thank Dr. Surendra Gupta for his technical and scientific advice, as well as introducing me to the typesetting software \LaTeX{}.

I wish to thank Dr. Mark Kempski for his valuable suggestions and encouragement.

I would like to recognize Dr. Charles Haines for generous support in the use of ACSL and valued advice in development of various equations used in this work.

The assistance of Fritz Howard, Joe Runde, and Dr. Victor Genberg is acknowledged.

Special thanks is owed to Monica DeMare for preparing most the figures in this work.

Most of all, I would like to express my deepest appreciation and gratitude to my parents, Joan and Carl Diehl. It was their encouragement and support that made my college years bearable ..., and almost always fun.
# Contents

1 INTRODUCTION 1
   1.1 Basic Types of Deformation 1
   1.2 Relevant History of Constitutive Modeling 2
   1.3 Objectives and Scope 6

2 ONE-DIMENSIONAL ANALYSIS 7
   2.1 Development of the Model 7
      2.1.1 The Basic Constitutive Equation 7
      2.1.2 Stress-Strain Curves 11
      2.1.3 Strain-Rate Sensitivity, \( n \) 12
      2.1.4 Yielding and Monotonic Saturation 14
      2.1.5 Work Hardening 17
      2.1.6 Creep and Stress Relaxation 32
   2.2 Numerical Simulations of Real Materials 37
      2.2.1 AISI 1040 Steel 38
      2.2.2 Commercially Pure Titanium 40
      2.2.3 Annealed Type 304 Stainless Steel 42
   2.3 Summary of Uniaxial Model 43

3 MULTIDIMENSIONAL ANALYSIS 46
   3.1 Development of Elastic-Viscoplastic Constitutive Equations in Multiple Dimensions 46
   3.2 Finite Element Implementation 49
   3.3 A Numerical Example 52
      3.3.1 Elastic Solution 54
      3.3.2 Strain-Rate Effects For Elastic-Viscoplastic Model 54
      3.3.3 Isotropic Hardening For Elastic-Viscoplastic Model 58

4 CONCLUSIONS 64

References 67

A PROGRAM FOR ONE DIMENSIONAL ANALYSIS, VISCO 70
B DERIVATIONS OF FINITE ELEMENT EQUATIONS 78
C AXISYMMETRIC FINITE ELEMENT PROGRAM, FEPROG 84
D PLOTTING PROGRAMS 131
List of Figures

1.1 Types of Material Behavior ................................................. 3
1.2 Stress-Strain Curves For Metals and Polymers ......................... 4

2.1 Basic Stress-Strain Curve For Uniaxial Loading .......................... 9
2.2 Strain-Rate Sensitivity .................................................. 13
2.3 Simulation of Strain-Rate Jump Tests ................................... 14
2.4 Changes of Yielding With Variations of Drag Values .................... 15
2.5 The Effects of the Epsilon Ratio on Monotonic Saturation ............... 16
2.6 Schematic For Ideally Plastic and Work Hardening Behavior .......... 18
2.7 Isotropic and Kinematic Hardening ...................................... 19
2.8 Drag Stress Changing as a Function of Time ............................ 21
2.9 Variation of Isotropic Hardening Parameters For Time-Dependent Hardening .................................................. 22
2.10 Variation of Isotropic Hardening Parameters For Viscoplastic-Dependent Hardening ................................................. 24
2.11 Kinematic Hardening Using the Formulation of James ................ 26
2.12 Kinematic Hardening Using a Modified Formulation .................. 27
2.13 Schematic of Cyclic Hardening and Cyclic Softening .................. 28
2.14 Cyclic Hardening With Combined Kinematic and Isotropic Hardening 29
2.15 Cyclic Softening With Only Isotropic Hardening ...................... 30
2.16 Cyclic Softening With Combined Kinematic and Isotropic Hardening 31
2.17 Summary of Equations For Modeling Work Hardening .................. 32
2.18 The Three Stages of Creep .............................................. 33
2.19 Simulation of Creep Test Showing Stage I and Stage II ............... 34
2.20 Schematic of Stress Relaxation ......................................... 35
2.21 Simulation of Stress Relaxation ........................................ 36
2.22 AISI 1040 Steel Data of Meyers ...................................... 39
2.23 Simulation of AISI 1040 Steel .......................................... 39
2.24 Two Simulations of Titanium at Different Strain Rates ............... 41
2.25 Actual Cyclic Test Data and Numerical Simulation of Titanium For 10 Cycles ......................................................... 41
2.26 Simulations of 304 Stainless Steel at Different Strain-Rates .......... 43
2.27 Actual Cyclic Test Data and Numerical Simulation of 304 Stainless Steel at 0.4%, 0.6%, and 1.0% Total Strain Range ....................... 44
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Actual Constrained Cylinder and Its Axisymmetric Representation</td>
<td>53</td>
</tr>
<tr>
<td>3.2</td>
<td>Contours of The Four Stress Components For an Elastic Compression of 0.01% Strain</td>
<td>55</td>
</tr>
<tr>
<td>3.3</td>
<td>Resulting Effective Stress For an Elastic Compression of 0.01% Strain</td>
<td>56</td>
</tr>
<tr>
<td>3.4</td>
<td>Strain Rate Effects For 2-D Model</td>
<td>57</td>
</tr>
<tr>
<td>3.5</td>
<td>Effective Stress Contours at 0.15% Engineering Strain For Engineering Strain Rates of 10.0 sec(^{-1}), 1.0 sec(^{-1}), and 0.1 sec(^{-1})</td>
<td>59</td>
</tr>
<tr>
<td>3.6</td>
<td>Variation of Drag Parameters For 2-D Model</td>
<td>60</td>
</tr>
<tr>
<td>3.7</td>
<td>Effective Stress Contours Without Hardening at Engineering Strain Levels of 0.04%, 0.05%, 0.10%, and 0.15%</td>
<td>61</td>
</tr>
<tr>
<td>3.8</td>
<td>Effective Stress Contours With Isotropic Hardening at Engineering Strain Levels of 0.04%, 0.05%, 0.10%, and 0.15%</td>
<td>62</td>
</tr>
<tr>
<td>3.9</td>
<td>Effective Stress Contours at 0.15% Engineering Strain For Various Sets of Drag Parameters</td>
<td>63</td>
</tr>
<tr>
<td>B.1</td>
<td>Isoparametric Mapping and Shape Function For Bilinear Quadrilateral Element</td>
<td>81</td>
</tr>
<tr>
<td>C.1</td>
<td>Flow Chart For Finite Element Program</td>
<td>85</td>
</tr>
</tbody>
</table>
Notation

$A, A_0$ current area, original area
$a_1, b_1$ isotropic hardening parameters
$a_2, b_2$ kinematic hardening parameters
$B, B_i$ back stress, initial back stress
$[C]$ strain-displacement matrix
$D, D_i$ drag stress, initial drag stress
$E, [E]$ modulus of elasticity, elastic stiffness matrix
$f$ general function
$[F^\text{ext}]$ applied external forces-rates
$[F^\text{vp}]$ viscoplastic force-rates
$G$ modulus of rigidity
$j$ Jacobian determinant
$[K]$ element stiffness matrix
$L, L_0$ current length, original length
$[L]$ linear differential operator
$M$ number of Gauss points in an element
$n$ strain-rate sensitivity factor
$N$ number of nodes in an element
$P$ applied load
$r, \theta, z$ Cylindrical coordinates; radial, theta, and axial
$S_{ij}, [S]$ deviatoric stress tensor (matrix)
$dS$ differential surface of element
$t$ time
$[U], u, v$ general displacements in an element
$[\delta U]$ variation of displacement
$W_i$ weighting factor for Gauss point $l$
$x, y, z$ Cartesian coordinates, also element physical coordinates
$\gamma_{xz}$ shear strain
$\delta_{ij}$ Kronecker delta
$\epsilon, \epsilon_i$ total strain, initial total strain
$\epsilon_c$ strain from creep
$\epsilon^e$ elastic strain
$\epsilon_{\text{eng}}$ engineering strain
$\epsilon_{\text{true}}$ true strain
$\epsilon^{\text{vp}}$ viscoplastic strain
$\dot{\epsilon}^{\text{vp}}$ effective viscoplastic strain-rate
$\dot{\epsilon}_o$ saturation constant
$[\delta \epsilon]$ variation of total strain
\( \lambda \)  
Lame constant in elastic stiffness matrix

\( \bar{\lambda} \)  
parameter in flow rule

\( \nu \)  
Poisson’s ratio

\( \xi, \eta \)  
natural coordinates of parametric space

\( \sigma, \sigma_{ij} \)  
stress (1-D), stress tensor

\( \sigma_{\text{eff}} \)  
effective stress

\( \sigma_{\text{eng}} \)  
enGINEERING stress

\( \sigma_{m} \)  
mean stress

\( \sigma_{s} \)  
stress at saturation

\( \sigma_{\text{true}} \)  
true stress

\( \sigma_{c}^{y} \)  
yield stress in compression

\( \sigma_{t}^{y} \)  
yield stress in tension

\( \tau \)  
shear stress

\( \psi \)  
shape functions

\( d\Omega \)  
differential volume of element

**Symbols**

\( \Delta \)  
small increment or change

\( || \)  
absolute value

\( \cdot \)  
time derivative \( \frac{d}{dt} \)

\( \frac{\partial}{\partial x} \)  
partial derivative with respect to \( x \) (or any other variable)

\( \text{sgn}(\cdot) \)  
returns sign of argument

\( \sum \)  
summation

\( [\cdot] \)  
matrix (or vector) quantity

\( [\cdot]^T \)  
matrix transpose

\( - \)  
nodal quantity
Chapter 1

INTRODUCTION

Modeling of material behavior is an integral part of structural mechanics and analysis. For simple elastic problems with common geometries and boundary conditions, Hooke's law and some simple handbook equations are sufficient. However, with the onset of the advanced technology age, demands for efficiency have forced engineers to design complicated structures capable of withstanding plastic deformations [1]. This need coupled with the improvement in numerical techniques such as finite elements, have resulted in the increasing use of predictive models. For these models to be accurate, a complete description of the material’s elastic and plastic behavior as a function of strain, strain-rate, and temperature is necessary [2]. The steam generator of the Clinch River Breeder Reactor [3] is an example where stress analysis required evaluation of the building materials’ strain-rate dependency.

This study focuses on the modeling of elastic-viscoplastic materials at constant temperature and their finite element implementations. A qualitative analysis of the developed constitutive equations is performed along with some quantitative simulations of real test data. The importance of deriving concise formulations, as noted by Eisenberg [4], has been a driving force in developing a relatively simple (but powerful) set of constitutive equations. The basic construction of the model is based on macroscopic physical behavior but does have roots in microscopic physical mechanisms. This work is a continuation of the investigations of Ghoneim [5].

1.1 Basic Types of Deformation

A discussion of the history of this subject is preceded by a brief overview of some of the types of deformation that a material can endure. A material under a particular load can experience elastic deformation, plastic deformation, or both. Elastically deformed bodies experience no permanent deformation and are path independent. This means that a stretched material will return to its original shape upon release of the applied load. Furthermore, exactly the same stress state will be reached if torsion and then tension is applied or visa-versa. On the other hand, plastically
deformed bodies do experience permanent deformation and are path dependent. Here, a stretched material will not return to its original shape and the stress state is dependent on the order (path) of the applied loads; torsion followed by tension is different from tension followed by torsion. Furthermore, if either elastic or plastic deformation depends on strain-rate, it is called viscoelastic or viscoplastic deformation, respectively. As noted in [1,6], strength properties in general tend to increase as strain-rates increase. These four types of deformation are summarized in figure 1.1.

Three of the four deformation processes in figure 1.1 dissipate energy. For these three processes, energy is assumed to transform from a mechanical form to primarily a thermal form. As discussed in [6], high strain-rates do not allow sufficient time for heat dissipation. Thus, the material temperature can increase during deformation. Furthermore, if there is time for some heat to dissipate, a temperature gradient can develop within a test specimen. For materials which have a high temperature dependence, this can cause localization of stresses. For simplicity, it is assumed in this study that these effects are negligible and that an isothermal condition exists.

Due to the increasing interest in strain-rate sensitivity testing after WWII [6], it is now well established that stress-strain curves for most metals are strain-rate sensitive [7]. In particular, they are elastic-viscoplastic; rate-independent for elastic deformation while being rate-dependent for plastic deformation [8]. There are no materials which are viscoelastic-plastic. Examples of a viscoelastic-viscoplastic material are some polymers such as polyethylene and polycarbonate. Typical uniaxial\(^1\) stress-strain curves for these cases are displayed in figure 1.2.

### 1.2 Relevant History of Constitutive Modeling

Several investigators have developed models governing material behavior. In the specific area of the theory of plasticity, models are generally either physical or mathematical [1]. Physical theories attempt to explain why metals flow (deform) plastically by looking at materials from a microscopic viewpoint; looking at grain boundaries, slip, and dislocations. This is the province of the material scientist and is too detailed for most engineering applications. Many engineers are not trying to describe why the deformation took place, but rather what happens to a material undergoing deformation in terms of stresses and strains. The mathematical theories are phenomenological, based on macroscopic observations. The most extreme of which are the empirical relations. Soroushian [9] derived empirical expressions for ratios of static to dynamic values of the yield and ultimate stresses for several types of steels. Unfortunately, multidimensional forms of these equations are not possible because of their purely empirical nature. As a result, the most useful theories for engineers are those that combine both approaches into one unified theory.

---

\(^1\)Uniaxial loading implies an one-dimensional state of stress. A common example is a tension test.
Figure 1.1: Types of Material Behavior
of plasticity.

Since Tresca published his paper in 1864, and Saint-Venant and Levy proposed some the the basic foundations for the theory of plasticity, numerous developments have occurred in this field [1]. From these original investigations, three main theories have emerged: Inviscid theory, Internal State theory, and Endochronic theory.

- Inviscid theory states that plastic deformation is path-dependent and time-independent. The elastic region is enclosed with a yield surface that translates (kinematic hardening) and expands (isotropic hardening) due to flow\(^2\) normal to the yield surface. As a result, plastic deformation occurs only when a stress parameter, usually the effective stress\(^3\), equals the value of the yield surface. Drucker [10] investigated time-independent cyclic loading using a yield surface. The formulation included kinematic hardening and exhibited the Bauschinger effect.\(^4\) Others such as Perzyna [8], Naghdi [11], and Rubin [12] have modified the inviscid theory to include a rate-dependent yield surface.

- Internal State theory requires that plastic deformation be both path-dependent and time-dependent. The history-dependence is incorporated through integration of the differential constitutive equations. The time dependence enables

\(\text{\textsuperscript{2}}\text{Flow means plastic deformation. Flow occurs under shear stress.}\)

\(\text{\textsuperscript{3}}\text{Effective stress is defined in Section 3.1.}\)

\(\text{\textsuperscript{4}}\text{Isotropic hardening, kinematic hardening, and Bauschinger effect are described in Section 2.1.5.}\)
investigation of strain-rate effects. Unlike Inviscid theory, there is no yield surface. Yielding (plastic deformation) is directly included into the constitutive equations and it can be affected by work hardening.

James et. al. [13] reviewed four current Internal State elastic-viscoplastic models: Bodner, Krieg et. al., Schmidt and Miller, and Walker. Using these four models, James performed numerical simulations of experimentally tested Inconel 718 at 593°C. All models assumed that the total deformation rate could be separated into elastic and inelastic (plastic) components which are functions of state variables. The models also included isotropic and anisotropic (directional) hardening capabilities. According to James, Bodner's and Walker's models each exhibited an exponential flow law; however, their formulations are different and fairly complex with 12 and 11 material constants, respectively. James also indicated that Schmidt and Miller used a hyperbolic sine flow law, which models creep better but still requires 11 material constants, and that Krieg et. al. employed a simpler formulation, a power law for modeling the flow with only 8 material constants. James also included his own generic model with a power formulation and 7 material constants. From his study, James concluded that Bodner and Walker handled strain-rate sensitivity best. Also in his findings, James emphasized the importance of concise and simple models with regard to determining all the material constants.

Ghoneim [5] modeled isotropic viscoelastic-viscoplastic isothermal deformation based on the separability of the total strain-rate as mentioned previously. Both power and exponential formulations for viscoplastic behavior were investigated in which he concluded that the power formulation was easier to interpret and apply numerically. Based on that conclusion, a finite element code was developed by Ghoneim for simulation of axisymmetric boundary value problems.

- Endochronic theory, as discussed by Lin [14], was developed by Valanis [15]. It is similar to the Internal State theory except that equations are formulated in integral form. Also, the time dependence is not measured by wall clock time, but rather by a material property itself. This theory appears to be the least developed of the three.

---

5 A flow law describes how plastic deformation occurs. It relates the inelastic strain-rate to the stress.

6 Ghoneim's power formulation is a simplified version of James' generic power law. It has different work hardening capabilities.
1.3 Objectives and Scope

The goals of this study are:

1. to obtain a sound fundamental understanding of the modeling of elastic-viscoplastic materials;
2. to develop equations that are suitable for engineering applications;
3. to expand the equations into multidimensional form for implementation of the finite element method.

The primary objective in this work is to develop both qualitative and quantitative simulation capability of strain-rate dependency and work hardening behavior for a general class of elastic-viscoplastic materials. Qualitative simulation capability of creep and stress relaxation is a secondary objective. Exact agreement with actual material data is not expected. Only a good representation of the material's behavior is intended.

Constitutive equations are initially developed for the one-dimensional case. The generic power law formulation of James [13] provides the basis for further study because its concise and simple composition permits a firm qualitative understanding of elastic-viscoplastic behavior. VISCO, a program developed and written in ACSL\textsuperscript{7}, calculates numerical solutions for various 1-D problems. Uniaxial analysis using VISCO is performed for monotonic and cyclic loading, work hardening (both isotropic and kinematic), creep, and stress relaxation. Simulations of monotonic and cyclic tests for published data of real materials is compared on a qualitative and quantitative level.

The equations, except kinematic hardening, are expanded into multiple dimensions and formulations for finite element implementation are presented. Demonstration of the multidimensional capabilities of the constitutive equations is accomplished through solution of a numerical example. Two-dimensional finite element analysis of the elastic-viscoplastic compression of a constrained cylinder under uniformly applied end displacements is demonstrated by enhancing the finite element code of Ghoneim [5] to include isotropic hardening.

\textsuperscript{7}ACSL is Advance Continuous Simulation Language.
Chapter 2

ONE-DIMENSIONAL ANALYSIS

This chapter develops the one-dimensional form of the constitutive equations governing isothermal elastic-viscoplastic behavior. Throughout their development, the performance of the equations is studied. After the relationships evolve into their final form, actual monotonic and cyclic test data for several metals is simulated.

2.1 Development of the Model

Development and understanding of the elastic-viscoplastic stress-strain relationships require use of some basic principles and study in the areas of monotonic and cyclic loading, strain-rate sensitivity, yielding, isotropic and kinematic hardening, creep, and stress relaxation. Throughout the investigation, an assumption of quasi-static loading [7] is used. This assumes that there are negligible resonance effects, wave propagation effects, and/or inertial reactions within the specimen. This applies for strain-rates under 10 sec\(^{-1}\). Also, an assumption of small deformation is enforced. This implies that the higher order terms in the displacement gradient are negligible and that constitutive and equilibrium equations can be written with respect to the undeformed geometry. For complete accuracy, all constitutive and equilibrium equations should be written with respect to the deformed geometry of a structure (which is unknown in advance) [16]. If the deformations are small, then the constitutive and equilibrium equations can be written with respect to the original geometry and the resulting errors will be negligible. Exact values of what constitutes a small deformation depend on geometry and deformation.

2.1.1 The Basic Constitutive Equation

The equations developed here come, in part, from the Internal State theory discussed in Chapter 1. The formulations arise in a differential form where the history
of loading is incorporated through integration of the equations in time. The constitutive equation for the elastic-viscoplastic model is based on two principles:

1. Hooke's law for linear elasticity is always valid.

2. The total strain is equal to the sum of the elastic strain and the viscoplastic strain.

At first, statement 1 seems incorrect. That is because Hooke's law is usually written as

$$\sigma = E \varepsilon$$

where $\sigma$ is the stress, $E$ is the modulus of elasticity, and $\varepsilon$ is the total strain. When plasticity occurs, this would be invalid because stress is not linearly related to total strain during plastic deformation. Thus, the assumption of elastic deformation is implied to equate total strain to elastic strain when Hooke's law is presented in this manner.

Hooke's law actually is

$$\sigma = E \varepsilon$$

(2.1)

where $\varepsilon^e$ is the elastic strain and $\sigma$ is still the stress. If plastic deformation occurs, equation 2.1 is still valid. For this to be applied, the elastic strain must be known. Statement 2 provides a method of finding that value:

$$\varepsilon = \varepsilon^e + \varepsilon^{vp}$$

(2.2)

This states that the total strain is the sum of the elastic (recoverable) and plastic\(^1\) (irrecoverable) strains. This is easily seen in a uniaxial stress-strain curve for which a material is loaded through the elastic limit into the plastic range and then unloaded. The stress returns to zero, but the strain returns only partially towards zero because of the permanent deformation caused by the plasticity. Figure 2.1 displays a summary of what Hooke's law actually governs and the different components of strain. Rearrangement of equation 2.2 and substitution into 2.1 yields

$$\sigma = E (\varepsilon - \varepsilon^{vp})$$

(2.3)

For viscoplasticity, the strain-rate effect implies a time dependence. Thus, it is desired that the stress-rates as well as strain-rates be evaluated. Differentiating equation 2.3 with respect to time leads to

$$\dot{\sigma} = E (\dot{\varepsilon} - \dot{\varepsilon}^{vp})$$

(2.4)

\(^1\)Since all the plastic deformation in this study is strain-rate dependent, the term plastic is often used interchangeably with the term viscoplastic for brevity.
where $E$ is assumed constant with respect to time. The total strain-rate, $\dot{\varepsilon}$, is the actual rate at which a deformation occurs. In a tension test, $\dot{\varepsilon}$ is what one would measure experimentally.

From the forms presented in [13], the viscoplastic strain-rate, $\dot{\varepsilon}^{vp}$, can be written as a function of the stress, back stress, and drag stress. Mathematically, the flow rule for viscoplasticity is represented as

$$\dot{\varepsilon}^{vp} = f \left( \frac{\sigma - B}{D} \right)$$  \hspace{1cm} (2.5)

where $f$ represents a function, $\sigma$ is the stress, $B$ is the back stress, and $D$ is the drag stress. The back stress produces directional (kinematic) hardening. The drag stress, always a positive quantity, is related to the magnitude of a material’s elastic limit and can produce isotropic hardening. Both back stress and drag stress are discussed later. The form of $f$ can be a power law, exponential, or hyperbolic sine. In this study, the power law form of $f$ will be employed as discussed previously. This results in a modified version of James’ generic formulation [13]

$$\dot{\varepsilon}^{vp} = \dot{\varepsilon}_o \left( \frac{\sigma - B}{D} \right)^n$$  \hspace{1cm} (2.6)

where $n$ is the strain-rate sensitivity factor. The value of $n$ determines how much the model depends on variations in strain rate. The modification is the inclusion
of a parameter $\dot{e}_o$, a positive constant. In James’ model, $\dot{e}_o = 1$. Adding the parameter $\dot{e}_o$ provides more flexibility when numerically working with the equation. The flow law, equation 2.6, is based upon the velocity of dislocations during plastic deformation.

Care must be taken in using this equation because of the power formulation. Under tension the viscoplastic strain-rate is positive, but it is negative under compression. However, an even value for $n$ always produces a positive viscoplastic strain-rate using equation 2.6. This is corrected by forcing the viscoplastic strain-rate to follow the same sign as $\sigma - B$. Therefore, equation 2.6 should be rewritten as

$$\dot{e}^{vp} = \dot{e}_o \left| \frac{\sigma - B}{D} \right|^n \text{sgn} (\sigma - B)$$

(2.7)

The function $\text{sgn}$ returns the sign, positive or negative, of its argument. Since $D$ is always positive, it is not included in the argument.

Combining equations 2.4 and 2.7 provides the differential form of the constitutive equation for elastic-viscoplastic behavior

$$\dot{\sigma} = E \left[ \dot{\varepsilon} - \dot{e}_o \left| \frac{\sigma - B}{D} \right|^n \text{sgn} (\sigma - B) \right]$$

(2.8)

Since the viscoplastic term signifies plastic deformation, equation 2.8 appears to predict that plasticity (yielding) will occur continuously. In fact it does. However, when $\sigma - B < D$ (roughly speaking), the value of $\dot{e}^{vp}$ is extremely small because the value of $n$ is typically between 20–50. Under this condition, $\dot{e}^{vp}$ is negligible compared to $\dot{\varepsilon}$ and the model is considered elastic. Yielding in the model is defined as prominent plastic deformation. This occurs as $\sigma - B$ approaches $D$. Under this condition, the plastic portion begins to dominate due to the power formulation and causes the deformation to become predominately plastic. As a result, the power law allows for the solution to have two distinct regions; elastic and viscoplastic. This is roughly speaking because the actual stress value where this transition occurs depends on the ratio of $\dot{\varepsilon}/\dot{e}_o$. This is discussed in Section 2.1.4.

The stress-rate is related to the total and plastic strain-rates in a nonlinear form. Solution of stress is by integration through time which incorporates the history of the loading path. A closed form solution is difficult, if not impossible, because of the nonlinear form. Hence, equation 2.8 must be solved numerically.

A program, VISCO, written in ACSL has been developed to solve this equation. ACSL [17] uses a fourth order Runge-Kutta integration scheme and is capable of solving the simultaneous equations that result from work hardening. VISCO solves stress as a function of total strain. Simulations of strain rate jump tests, tension-compression cyclic loading, work hardening, creep tests, and stress relaxation tests are also available. The program listing for VISCO is in Appendix A.
2.1.2 Stress-Strain Curves

Since evaluation of the equations developed will be numerical, a discussion of stress-strain curves is appropriate. Two main types of curves will be plotted; monotonic and cyclic. A monotonic curve or simulation is when a material is deformed in one direction; such as tension or compression. There is no reversal in strain-rate (or strain). A cyclic curve or simulation has a reversal in strain-rate. Three types of cyclic loading are: tension-tension, compression-compression, and tension-compression. For tension-tension, a specimen is loaded in tension, unloaded partially or completely, and then reloaded in tension. The specimen is never placed in compression. A similar loading scheme is used in compression-compression. Tension-compression loading requires the material to be loaded in tension, unloaded, and then loaded into compression. Either direction can be initially applied. For our investigations, tension-compression cyclic loading will be performed and referred to simply as cyclic. The cyclic loading is valid only for a low number of cycles and is not intended to be a model for high cycle fatigue. In the discussions involving cyclic loading, the word monotonic is intended to imply either the tension or compression portion of a given cycle.

All stress plots, unless otherwise specified, are normalized with respect to the initial drag stress, $D_i$. This produces a better indication of the performance of the model. One exception, real test data, is not normalized since most published data is in a non-normalized form. The values of modulus of elasticity, initial drag, and strain-rate sensitivity used in simulating the equations for the development of the model are loosely based on annealed 304 stainless steel. The detailed simulation of this material is performed and shown in the section of numerical simulations of real materials.

For the uniaxial simulations performed, the maximum total true strain in any one direction (tension or compression) has been kept below 2.0% to avoid problems related to necking. For the cyclic tests, this would be equivalent to 4.0% total strain. With this limitation, certain assumptions can be made about the engineering and true values of stress and strain. The engineering values of stress and strain are defined as

$$
\sigma_{\text{eng}} = \frac{P}{A_0}
$$

$$
\epsilon_{\text{eng}} = \frac{\Delta L}{L_0}
$$

where $P$ is the applied load, $\Delta L$ is the change of length, and $A_0$ and $L_0$ are the original area and length. The true stress and strain values are found from [6]

$$
\sigma_{\text{true}} = \frac{P}{A}
$$
\[ \epsilon_{\text{true}} = \int_{L_0}^{L} \frac{dl}{l} = \ln (\epsilon_{\text{eng}} + 1) \]

where \( A \) and \( L \) are the current area and length.

With the limitation of 2.0% total strain, there is less than a 1.0% difference between \( \epsilon_{\text{eng}} \) and \( \epsilon_{\text{true}} \), and \( \sigma_{\text{eng}} = 0.98\sigma_{\text{true}} \). Therefore, in the uniaxial case, the engineering values and true values can be assumed to be equal. For the multidimensional simulations of Chapter 3, this is not valid because of non-uniform stress distributions in the multidimensional stress state.

It should also be noted that for uniaxial simulations, there is no distinction between compression and tension except for sign. This is valid under the strain limitations previously discussed. The monotonic simulations investigated in this chapter reflect a state of tension, but they also apply to a compressive state when appropriate signs are added to the results.

As discussed in [18], the values of proportional limit, elastic limit, and yield strength help define some major features of the stress-strain curve (and are often used incorrectly). Proportional limit is the stress value after which stress is no longer linear with strain. Elastic limit is the greatest stress a material can withstand before undergoing permanent deformation. This limit may be equal to or higher than the proportional limit. It defines the boundary for yielding and is considered the yield surface in the Inviscid theory. Yield strength is the stress related to a specified strain that is slightly higher than that associated with the elastic limit. The 0.2% offset is a common example. Since Hooke’s law is used in the development of the constitutive equation, elasticity is considered linear. This implies that solutions of equation 2.8 will produce equal values for the proportional limit and elastic limit. As a result, modeling of materials with nonlinear elastic regions such as Aluminum should be done with caution or avoided entirely.

As noted in Chapter 1, the model is macroscopic. Occurrences such as yield point phenomenon are not predicted by the model. This is mostly a microscopic effect caused by pinned dislocations occurring in many annealed metals. At high strain-rates, however, this often is not visible.

### 2.1.3 Strain-Rate Sensitivity, \( n \)

Strain-rate sensitivity describes dependency of a material’s behavior to the deformation rate. A material with high strain-rate sensitivity will have very different plastic deformation curves over a range of strain-rates whereas low sensitivity produces curves with little variation across a similar range. The parameter \( n \) in equation 2.8 allows control of the visco effects in the model.

To see how \( n \) effects the model, solutions are calculated for monotonic simulations at several strain-rates for two sensitivities and are shown in figure 2.2. Simulations of a jump test, a rapid change in strain-rates, are also displayed in fig-

\(^2\)These calculations are based on tension.
$n = 20$

Key:

- A $\dot{\varepsilon} = 10.0 \text{sec}^{-1}$
- B $\dot{\varepsilon} = 1.0 \text{sec}^{-1}$
- C $\dot{\varepsilon} = 0.1 \text{sec}^{-1}$
- D $\dot{\varepsilon} = 0.01 \text{sec}^{-1}$

All simulations: $E = 21.7 \times 10^8 \text{psi}$, $\dot{\varepsilon}_0 = 1.0 \text{sec}^{-1}$, $D = \text{constant}$, and $B = 0.0 \text{psi}$.

Figure 2.2: Strain-Rate Sensitivity
Rapid Increase in Strain-Rate

\[ \dot{\varepsilon}_2 = 1.0 \text{sec}^{-1} \]
\[ \dot{\varepsilon}_1 = 0.01 \text{sec}^{-1} \]

Rapid Decrease in Strain-Rate

\[ \dot{\varepsilon}_1 = 1.0 \text{sec}^{-1} \]
\[ \dot{\varepsilon}_2 = 0.01 \text{sec}^{-1} \]

All simulations: \( E = 21.7 \times 10^6 \text{psi} \), \( \dot{\varepsilon}_0 = 1.0 \text{sec}^{-1} \), \( n = 20 \), \( D = \text{constant} \), and \( B = 0.0 \text{psi} \).

Figure 2.3: Simulation of Strain-Rate Jump Tests

For these solutions, no hardening is considered; \( B \) is set to zero and \( D \) is kept constant.

For all curves, only the plastic portion is affected by the strain-rate change. In figure 2.2, a larger \( n \) produces less sensitivity. In the limit as \( n \) approaches \( \infty \), the model degenerates to elastic-perfectly plastic. In this limiting case, stress is not a function of strain-rate and yielding occurs at a value of \( D \), the drag stress. At the other end of the spectrum is \( n = 1 \), a viscoelastic model. For most metals, the sensitivity, \( n \), varies from 20–50.

The strain-rate jumps of figure 2.3 show the same behavior as Bodner [19] found under experimental evaluation for titanium. A rapid change in strain-rate (from \( \dot{\varepsilon}_1 \) to \( \dot{\varepsilon}_2 \)) causes the stress to jump from that produced by \( \dot{\varepsilon}_1 \) towards a new stress value. This new value of stress is the same stress value that occurs if \( \dot{\varepsilon}_2 \) is applied continuously from the start. This behavior is expected for some classes of elastic-viscoplastic metals. If \( D \) is not constant (work hardening), the stress after a strain-rate jump will be between comparable stress values obtained for \( \dot{\varepsilon}_1 \) and \( \dot{\varepsilon}_2 \) [20].

2.1.4 Yielding and Monotonic Saturation

Yielding of a material is the beginning of plasticity. A material yields when its elastic limit is exceeded and permanent deformation occurs. The drag, \( D \), largely influences when yielding occurs. Simple variation of the drag shows that increasing
the value of $D$ causes the material to yield at higher values as depicted in figure 2.4.\footnote{This stress-strain curve is not normalized to $D$ since $D$ is being varied.}

Hence, the value of $D$ is strongly related to the elastic limit of a material. Another parameter, $\dot{\epsilon}_o$, is related to saturation of a material and also influences when yielding occurs. Thus, investigation of the parameter $\dot{\epsilon}_o$ and the saturation condition is needed for complete understanding and application.

Monotonic saturation is when stress no longer increases with increased strain; $\dot{\sigma} = 0$. When monotonic saturation occurs, there is no hardening. As was the case for the simulations of figure 2.2, the model saturated almost immediately after becoming plastic. In general, real materials would harden with increasing plasticity before saturating. This effect is corrected by allowing $B$ and/or $D$ to change and is discussed in Section 2.1.5. However, first we will look at saturation.

Under monotonic saturation, the constitutive equation 2.8, with no hardening ($B = 0$ and $D$ is constant), degenerates to

$$0 = E \left[ \dot{\epsilon} - \dot{\epsilon}_o \left| \frac{\sigma_s}{D} \right|^n \text{sgn} (\sigma_s) \right]$$

where $\sigma_s$ denotes $\sigma$ at saturation. Rearranging this equation leads to

$$\left| \frac{\sigma_s}{D} \right|^n = \left( \frac{\dot{\epsilon}}{\dot{\epsilon}_o} \right) \text{sgn} (\sigma_s)$$

Since $\dot{\epsilon}_o$ is always positive, and the numerical signs of $\dot{\epsilon}$ and $\sigma_s$ are always the same under the saturation condition, the right hand side of this equation is always positive. However, the absolute value sign that is applied to the left side of the equation still provides some difficulty in solving for $\sigma_s$ since the solution has two

All simulations:

\[
\begin{align*}
E &= 21.7 \times 10^6 \text{ psi} \\
\dot{\epsilon} &= 1.0 \text{ sec}^{-1} \\
\dot{\epsilon}_o &= 1.0 \text{ sec}^{-1} \\
n &= 20 \\
B &= 0.0 \text{ psi}
\end{align*}
\]
possibilities, positive and negative. The key to the choice of positive or negative is that the sign of $\sigma_s$ is the same as the sign of $\dot{\varepsilon}$ (previously stated). Thus, under the saturation condition, the stress is

$$\sigma_s = D \left| \frac{\dot{\varepsilon}}{\dot{\varepsilon}_0} \right|^{1/n} \text{sgn}(\dot{\varepsilon})$$  \hspace{1cm} (2.9)

From equation 2.9, the effect of adding $\dot{\varepsilon}_o$ to James' original formulation can be realized. With this additional parameter, the saturation value can be controlled. The value of the epsilon ratio determines if saturation is above, equal to, or below the drag value.

$$\left| \frac{\dot{\varepsilon}}{\dot{\varepsilon}_0} \right| > 1 \Rightarrow |\sigma_s| > D$$

$$\left| \frac{\dot{\varepsilon}}{\dot{\varepsilon}_0} \right| = 1 \Rightarrow |\sigma_s| = D$$

$$\left| \frac{\dot{\varepsilon}}{\dot{\varepsilon}_0} \right| < 1 \Rightarrow |\sigma_s| < D$$

These results do not depend on the value of $n$, except for $n = \infty$ and $n = 0$. The results are illustrated in figure 2.5. Because of this behavior, $\dot{\varepsilon}_o$ is termed the saturation constant.

In all the monotonic simulations, the model is over-square; a sharp change from elasticity to plasticity. A real material would not have such an abrupt change in that transition. An attempt was made to overcome this problem by averaging
two different sensitivities. Although it is not shown clearly in figures 2.2 and 2.5, lower $n$ values produce smoother transitions. Unfortunately, the averaging scheme failed because one $n$ would override the other. A moderately successful method of smoothing the curves is to incorporate work hardening into the model. These effects are discussed in Section 2.2. Although over-square behavior is not generally realistic, it is useful in creating the numerical model.

The over-square behavior in the model can be used to determine the the elastic limit and many other parameters. With no hardening, the saturation value can be assumed to be the elastic limit because of the sharp transition. As shown, when $\dot{\epsilon}_0 = \dot{\epsilon}$, the stress saturates at the drag value, $D$. The drag and elastic limit have the same magnitude under this condition. With strain-rate variation, the stress values in figure 2.5 vary above and below this nominal value of elastic limit. As a result, $\dot{\epsilon}_0$ is chosen to be equal to the average strain-rate being investigated. Evaluation of the elastic limit at that average strain-rate provides the appropriate value of $D$. For figure 2.5, curve C would be the average strain-rate. The drag, $D$, would be set equal to the magnitude of the elastic limit for curve C and $\dot{\epsilon}_0$ would be set to the strain-rate of curve C. Then variations in strain-rate, $\dot{\epsilon}$, would produce curves A, B, D, and E. The sensitivity to the strain-rate variation would depend on the value of $n$ as discussed in Section 2.1.3.

This approach is a good and fast method of obtaining first cut values for the parameters. With the addition of work hardening, these values for the parameters may need to be adjusted slightly. It might also be necessary to fit the model to two or more strain-rate ranges depending on the materials' actual behavior.

As for the degeneration of the model from quasi-static to static, figure 2.5 shows the plastic region becoming less sensitive to strain-rate as $\dot{\epsilon}$ is lowered (as expected). However, the model will not reach a base static value for which there is no strain-rate effect. Real materials would exhibit this base value. As a result, the model does not fully degenerate to the static case and must be used only in areas where strain-rate sensitivity occurs.

### 2.1.5 Work Hardening

Work hardening, the strengthening of a material due to plastic deformation, is a mechanism of increasing the elastic limit of metals. When metals deform plastically, they become more resistant to plastic deformation and require a larger stress to produce further deformation. Until now, the investigation of equation 2.8 assumed ideally plastic behavior. It did not include work hardening. The difference between ideally plastic and work hardening behavior is displayed in figure 2.6.

In this study, two types of hardening are discussed and modeled: isotropic and kinematic hardening. As in [13], these two types of hardening are considered completely separable and controlled by different variables. During isotropic hardening, the strength of the material increases equally in all directions, regardless of the direction of the applied strain [21]. The microscopic causes include grain bound-
aries, subgrains, precipitate particles, and dislocation entanglements [22]. Kinematic hardening is directional. Prager's kinematic model assumes that the yield surface translates in the direction of the plastic deformation [21]. An increase in the tensile elastic limit from tensile plastic deformation would imply a decrease in the compressive elastic limit due to translation of the yield surface. Microscopic causes for kinematic hardening are dislocation pileups and bowing of pinned dislocations between their obstacles [22]. Actual hardening is assumed to be a combination of both isotropic and kinematic hardening.

Figure 2.7 displays isotropic and kinematic hardening. For a virgin material\(^4\), one which has not been deformed plastically, both tensile and compressive elastic limits will be equal (points A and B). If the material is loaded to point C and then released, point C becomes the new elastic limit in tension. For isotropic hardening, further loading in compression along line CDE will reveal that the elastic limit in compression is at point E. This value is the same magnitude as point C. Hence, the material's elastic limit has increased equally in both directions for isotropic hardening. Under the kinematic hardening model, the curve up to point C would be the same, but compression along line CDE would reveal that the elastic limit in compression is at point D. This value is less than both the value at point C and the value at point B (virgin compressive elastic limit). Here the elastic limit which defines the yield surface has translated. For this definition of kinematic hardening, the total elastic range of the material is considered constant.

The anisotropic nature of kinematic hardening allows material behavior such as \textit{Bauschinger effect} to be simulated. Bauschinger effect is defined as follows [18]:

\(^4\)A virgin material can also be produced by removing the residual stresses through annealing.
Virgin Material

\[
|\sigma_X^V| = |\sigma_X^C|
\]
before hardening

Isotropic Hardening

\[
|\sigma_X^I| = |\sigma_X^C|
\]
after hardening

Kinematic Hardening

\[
|\sigma_X^K| + |\sigma_Z^K| = |\sigma_X^C| + |\sigma_Z^C|
\]
before hardening after hardening

Figure 2.7: Isotropic and Kinematic Hardening
When a material is deformed plastically in one direction, its elastic limit in that direction is raised while its elastic limit in the opposite direction is lowered when compared to the original values before plastic deformation. As Datsko [18] has concluded after studying much experimental data, this is only valid for small strains. When large plastic deformations are analyzed, Datsko found that the elastic limits in the direction of loading as well as those in the opposite direction increase compared to the values for the virgin material. However, the elastic limit in the direction of loading is still greater than that in the opposite direction.

Using both isotropic and kinematic hardening, the observations of Datsko can be explained. For the small strain studies, most of the hardening was kinematic and resulted in the Bauschinger effect (as defined above). The isotropic hardening was not very pronounced. In the large plastic deformations, the isotropic hardening was more prevalent, and raised both tensile and compressive elastic limits above their original values. The kinematic hardening caused the elastic limit in the direction of loading to be raised higher than in the opposite direction. It should be noted that the large plastic deformations that Datsko studied violate the original assumptions in this study of 2.0% total strain maximum in one direction (4.0% for cyclic). However, the explanation is still valid within these assumptions. Hence, the equations being developed do have the power to simulate such behavior if it occurs within a total strain of 2.0% (monotonic).

**Isotropic Hardening Equation**

As seen in figure 2.4, variations in the drag, \( D \), produce changes in the elastic limit. By allowing \( D \) to change, isotropic hardening can be modeled. Two models (Model 1 and Model 2) will be developed that will allow \( D \) to change. For these investigations, \( \dot{\varepsilon}/\dot{\varepsilon}_0 = 1 \) which implies that the drag and elastic limit are equal in magnitude.\(^5\) Also, there is no kinematic hardening (\( B = 0 \)). The first model presented is not completely realistic but provides the basic comprehension needed for the investigations of Model 2.

- Model 1 is based on the assumption that the drag changes with time and can be represented as a simple 1\(^{st} \) order equation

\[
\dot{D} = (b_1 - a_1D) \quad D = D_i \text{ at } t = 0 \tag{2.10}
\]

where \( a_1 \) and \( b_1 \) are constants, \( D_i \) is the initial drag, and \( t \) is time. This equation is coupled with the constitutive equation 2.8 when simulating materials. Equation 2.10 describes how \( D \) in equation 2.8 changes. However, equation 2.10 is independent of equation 2.8. As a result, a closed form solution to equation 2.10 can be found. The solution to equation 2.10 is

\(^5\)This is done only as an aid for visual simplicity when viewing the figures.
This solution is plotted as a function of time and as a function of strain (for cyclic loading) in figure 2.8. The cyclic loading is used because it allows more insight into the model and is a realistic deformation process. The drag is always positive regardless of the direction of loading. The rise time is controlled by \( a_1 \) and the value of \( D \) at \( t = \infty \) is \( b_1 / a_1 \). The parameters \( a_1 \) and \( b_1 \) control how fast the drag changes and when the drag will saturate.

Figure 2.9 depicts three simulations using equations 2.8 and 2.10 where different sets of values for the parameters \( a_1 \) and \( b_1 \) are used. For all three simulations, the quantity \( b_1 / a_1 \) is the same to provide the same value for \( D \) at \( t = \infty \). Plot A has a slow rise time and the stress never reaches cyclic saturation\(^6\) within the finite number of cycles simulated. This is due to the continuous changing of drag. The rise time of plot B is faster and the stress begins to saturate. The onset of saturation is due to the decreasing change in the drag from the faster rise time. In plot C, the rise time is large enough that the drag eventually becomes constant causing cyclic saturation of the stress.

For Model 1, the drag (and elastic limit) are constantly increasing, regardless of elastic or plastic deformation (figure 2.9). Although not depicted, equation 2.10 also allows the drag and elastic limit to change even if a material is sitting unloaded. This is not very realistic because a material’s elastic limit does not increase if the specimen is only deformed elastically or not deformed at all.\(^7\)

---

\(^6\)Cyclic saturation occurs when the maximum stress per cycle no longer changes.
\(^7\)Over a period of months, Strain Aging could occur [23].
A) \( a_1 = 10.0 \text{sec}^{-1}, \ b_1 = 2.0 \times 10^5 \text{psi/sec} \)

B) \( a_1 = 50.0 \text{sec}^{-1}, \ b_1 = 1.0 \times 10^6 \text{psi/sec} \)

C) \( a_1 = 100.0 \text{sec}^{-1}, \ b_1 = 2.0 \times 10^6 \text{psi/sec} \)

All simulations: \( E = 21.7 \times 10^6 \text{psi}, \ \dot{\varepsilon} = 1.0 \text{sec}^{-1}, \ \dot{\varepsilon}_0 = 1.0 \text{sec}^{-1}, \ n = 20, \) and \( B = 0.0 \text{psi}. \)

Figure 2.9: Variation of Isotropic Hardening Parameters For Time-Dependent Hardening
• A more realistic model (Model 2) would assume that the drag changes with plastic deformation. This can be accomplished by modifying Model 1 such that \( D \) only changes with plastic deformation. One plausible equation for the drag-rate is

\[
\dot{D} = |\dot{\varepsilon}^{vp}| (b_1 - a_1 D) \quad D = D_i \text{ at } t = 0
\]

(2.12)

The parameters \( a_1 \) and \( b_1 \) are the same as before except that their units are different in order to keep dimensional consistency (see figures 2.9 and 2.10). The same conclusions regarding rise time and saturation that were stated for Model 1 are valid for Model 2. A closed form solution to equation 2.12 is not possible because the equation depends on equation 2.8. Equations 2.12 and 2.8 are coupled and must be solved simultaneously.

The absolute value sign in equation 2.12 is necessary because \( D \) changes regardless of strain-rate direction. The drag is either always increasing (work hardening) or always decreasing (work softening). James [13] used a similar growth equation but included an additional drag stress recovery term.

Equation 2.12 displays behavior similar to equation 2.10 with respect to rise time variations (parameters \( a_1 \) and \( b_1 \)) and resulting cyclic saturation characteristics (figures 2.9 and 2.10). The important difference is that the drag in equation 2.12 is constant during elastic deformations (\( \dot{\varepsilon}^{vp} = 0 \)) and only changes under plastic deformations (\( \dot{\varepsilon}^{vp} \neq 0 \)). In the simulations of figure 2.10, the drag is always constant (flat horizontal regions on drag curves) during any elastic deformation process. Although not depicted, the drag, \( D \), is also constant when the material is sitting because \( \dot{\varepsilon}^{vp} = 0 \) under this condition. Hence, equation 2.12 is more realistic than equation 2.10. Equation 2.12 is called the isotropic hardening equation.

**Kinematic Hardening Equation**

For kinematic hardening, one can assume that the hardening is a function of plastic deformation as in Model 2. The Back stress, \( B \), produces this behavior. An equation similar to equation 2.12 is

\[
\dot{B} = \dot{\varepsilon}^{vp} (b_2 - a_2 B) \quad B = 0 \text{ at } t = 0
\]

(2.13)

where \( a_2 \) and \( b_2 \) are constants similar to those of equation 2.12. The absolute value sign does not appear because kinematic hardening is not an additive effect. Over one cycle, the elastic limit should return to its original value since the yield surface is translated in one direction and then translated back in the opposite direction. The initial value of \( B \) at \( t = 0 \) is zero to ensure that, for a virgin material, there are equal values of the elastic limit in tension and in compression (\( B \) only changes during plastic deformation). This equation is similar to the equation James [13]
A) $a_1 = 10.0$, $b_1 = 2.0 \times 10^5 \text{psi}$

![Graph A](image1)

B) $a_1 = 50.0$, $b_1 = 1.0 \times 10^6 \text{psi}$

![Graph B](image2)

C) $a_1 = 100.0$, $b_1 = 2.0 \times 10^6 \text{psi}$

![Graph C](image3)

All simulations: $E = 21.7 \times 10^6 \text{psi}$, $\dot{\epsilon} = 1.0 \text{sec}^{-1}$, $\dot{\epsilon}_0 = 1.0 \text{sec}^{-1}$, $n = 20$, and $B = 0.0 \text{psi}$.

Figure 2.10: Variation of Isotropic Hardening Parameters For Viscoplastic-Dependent Hardening
used for kinematic hardening. The only difference is that James included thermal recovery.

Figure 2.11 depicts the simulation of cyclic loading with the kinematic hardening of equation 2.13 ($D$ is constant; no isotropic hardening). Since the yield points are merely translating back and forth, figure 2.11 could represent one cycle or ten cycles. The subtraction of $B$ from $\sigma$ in equation 2.8 causes the anisotropic behavior. Under tensile deformation, continued plastic deformation allows $B$ to increase (equation 2.13) which in turn forces $\sigma$ to increase since $B$ is subtracted. The net result is work hardening. However, if the strain rate is then reversed (cyclic loading), then $B$ initially becomes additive to $\sigma$ when the plastic region in compression is reached. This is because $\sigma$ is negative in compression and $B$ stays constant (a positive value) during the transition from tension to compression (an elastic deformation process). As a result of this additive effect, a lower value of stress, $\sigma$, is needed to produce yielding in compression than was required in tension. Continued compression causes $B$ to decrease ($\dot{\epsilon}^p$ is negative in compression) and eventually $B$ becomes negative. This in turn forces $\sigma$ to increase, producing work hardening in compression. This subtraction/addition provides for the translation of the elastic limits for kinematic hardening. If compression is applied first, then the opposite occurs. The compressive elastic limit becomes higher than the tensile elastic limit.

Figure 2.11 also shows how $B$ and $\sigma$ change with respect to time. The flat horizontal regions in the back stress plot are caused by elastic deformation (no hardening occurs during elastic deformation). The equation produces the Bauschinger effect but has a major flaw. The simulation in figure 2.11 is unstable under compression. The stress curve is concave down under compression instead of concave up. As seen in the time plots, the back stress, $B$, is asymptotically approaching infinity instead of asymptotically approaching a stable (finite) value. Also the mean value of $B$ is offset from zero. These undesirable results can be traced back to equation 2.13. Since $\dot{\epsilon}^p$ varies from positive values in tension to negative values in compression, the solution of the equation is basically tracing over itself when $\dot{\epsilon}^p$ is reversed, as shown in the time plots of figure 2.11.

A numerical correction of adding an absolute value sign can be made to equation 2.13 such that it will be stable in both tension and compression. Applying the correction yields the modified formulation

$$\dot{B} = \dot{\epsilon}^p b_2 - |\dot{\epsilon}^p| a_2 B \quad B = 0 \text{ at } t = 0 \quad (2.14)$$

Simulation of equation 2.14 is shown in figure 2.12 ($D$ is constant; no isotropic hardening). The model is stable in both tension and compression and produces no net change in elastic limit over 1 cycle. The Bauschinger effect is apparent and the yield translates back and forth equally as expected. The time plots of stress and back stress agree with those presented by Miller [22]. As a result, equation 2.14 is preferred for modeling kinematic hardening.
\[ \dot{\sigma} = E \left( \dot{\varepsilon} - \dot{\varepsilon}_* \right) \left( \frac{\sigma - B}{D} \right)^n \text{sgn}(\sigma - B) \]

\[ \dot{B} = \dot{\varepsilon}^\ast (b_2 - a_2 B) \]

All simulations:

- \( E = 21.7 \times 10^6 \text{psi} \)
- \( \dot{\varepsilon} = 1.0 \text{sec}^{-1} \)
- \( \dot{\varepsilon}_* = 1.0 \text{sec}^{-1} \)
- \( n = 20 \)
- \( a_2 = 400.0 \)
- \( b_2 = 2.5 \times 10^5 \text{psi} \)
- \( B_i = 0.0 \text{psi} \)
- \( D = \text{constant} \)

Figure 2.11: Kinematic Hardening Using the Formulation of James
\[ \dot{\varepsilon} = E \left[ \dot{\varepsilon} - \dot{\varepsilon}_o \right]^n \frac{\sigma - B}{D} \operatorname{sgn}(\sigma - B) \]

\[ \dot{B} = \dot{\varepsilon}^n b_2 - |\dot{\varepsilon}| a_2 B \]

All simulations:

- \( E = 21.7 \times 10^6 \text{psi} \)
- \( \dot{\varepsilon} = 1.0 \text{sec}^{-1} \)
- \( \dot{\varepsilon}_o = 1.0 \text{sec}^{-1} \)
- \( n = 20 \)
- \( a_2 = 400.0 \)
- \( b_2 = 2.5 \times 10^6 \text{psi} \)
- \( B_i = 0.0 \text{psi} \)
- \( D = \text{constant} \)

Figure 2.12: Kinematic Hardening Using a Modified Formulation
Cyclic Hardening and Cyclic Softening

Combining both isotropic and kinematic hardening allows for the simulation of both cyclic hardening and cyclic softening. In general, soft or annealed metals tend to cyclic harden while hard or cold-worked metals usually exhibit cyclic softening [10,24,25,26]. Figure 2.13 depicts both types of behavior.

Cyclic hardening has been seen previously in the simulations of figures 2.9 and 2.10. These were produced with only isotropic hardening (back stress was zero). Hence, cyclic hardening of an isotropic metal can be simulated with the constitutive equation 2.8 coupled to equation 2.12 for isotropic hardening.

For materials that exhibit the Bauschinger effect, kinematic hardening is also needed. Kinematic hardening cannot be used alone since it has no cumulative effects over a cycle. Isotropic hardening, as seen from the previous plots in figures 2.9 and 2.10 produces accumulative hardening effects over a cycle. Thus, three equations are employed—2.8, 2.12, and 2.14. A simulation of kinematic cyclic hardening is illustrated in figure 2.14. The time plots show that the drag increases with time and the back stress is periodic (varies back and forth) depicting translation of the yield point.

For cyclic softening, the maximum stress within a cycle decreases as the material experiences repeated cyclic deformation. However, in general, a metal usually appears to harden during each monotonic section of the deformation (figure 2.13). Modeling this type of material is difficult. Figures 2.15 and 2.16 depict two different simulations of cyclic softening. The simulation in figure 2.15 attempts to model the behavior using only isotropic hardening (equation 2.12). The simulation exhibits monotonic softening caused by an ever decreasing drag stress. This is generally undesirable. The simulation in figure 2.16 uses both isotropic and kinematic hardening (equations 2.12 and 2.14) to simulate cyclic softening. Since the kinematic equation produces no net effects over a cycle, the drag equation must be responsible
\[
\dot{\sigma} = E \left[ \dot{\varepsilon} - \dot{\varepsilon}_0 \left| \frac{\sigma - B}{D} \right|^n \text{sgn}(\sigma - B) \right]
\]
\[
\dot{D} = |\dot{\varepsilon}^{yp}| (b_1 - a_1 D)
\]
\[
\dot{B} = \dot{\varepsilon}^{yp} b_2 - |\dot{\varepsilon}^{yp}| a_2 B
\]

All simulations:

- \( E = 21.7 \times 10^6 \) psi
- \( \dot{\varepsilon} = 1.0 \text{sec}^{-1} \)
- \( \dot{\varepsilon}_0 = 1.0 \text{sec}^{-1} \)
- \( n = 20 \)
- \( a_1 = 50.0 \)
- \( b_1 = 1.0 \times 10^6 \) psi
- \( a_2 = 400.0 \)
- \( b_2 = 2.5 \times 10^6 \) psi
- \( B_i = 0.0 \) psi

Figure 2.14: Cyclic Hardening With Combined Kinematic and Isotropic Hardening
\[ \dot{\varepsilon} = E \left[ \dot{\varepsilon} - \dot{\varepsilon}_0 \frac{|\sigma - B|}{D} \right]^n \text{sgn}(\sigma - B) \]

\[ \dot{D} = |\dot{\varepsilon}| (b_1 - a_1 D) \]

All simulations:

- \( E = 21.7 \times 10^6 \text{psi} \)
- \( \dot{\varepsilon} = 1.0 \text{sec}^{-1} \)
- \( \dot{\varepsilon}_0 = 1.0 \text{sec}^{-1} \)
- \( n = 20 \)
- \( a_1 = 25.0 \)
- \( b_1 = 1.0 \times 10^6 \text{psi} \)
- \( B = 0.0 \text{psi} \)

Figure 2.15: Cyclic Softening With Only Isotropic Hardening
\[
\sigma = E \left[ \dot{\varepsilon} - \dot{\varepsilon}_0 \right] \frac{|\sigma - B|}{D} \text{ sgn}(\sigma - B)
\]
\[
\dot{D} = |\dot{\varepsilon}_0| (b_1 - a_1 D)
\]
\[
\dot{B} = \dot{\varepsilon}_0 b_2 - |\dot{\varepsilon}_0| a_2 B
\]

All simulations:
- \( E = 21.7 \times 10^6 \text{ psi} \)
- \( \dot{\varepsilon} = 1.0 \text{ sec}^{-1} \)
- \( \dot{\varepsilon}_0 = 1.0 \text{ sec}^{-1} \)
- \( n = 20 \)
- \( a_1 = 25.0 \)
- \( b_1 = 1.0 \times 10^5 \text{ psi} \)
- \( a_2 = 400.0 \)
- \( b_2 = 2.5 \times 10^6 \text{ psi} \)
- \( B_i = 0.0 \text{ psi} \)

Figure 2.16: Cyclic Softening With Combined Kinematic and Isotropic Hardening
\[
\dot{\varepsilon} = E \left[ \dot{\varepsilon}_0 \left| \frac{\sigma - B}{D} \right|^n \text{sgn} (\sigma - B) \right]
\]

\[
\dot{D} = |\dot{\varepsilon}_p| (b_1 - a_1 D)
\]

\[
\dot{B} = \dot{\varepsilon}_p b_2 - |\dot{\varepsilon}_p| a_2 B
\]

where:

\[
\dot{\varepsilon}_p = \dot{\varepsilon}_0 \left| \frac{\sigma - B}{D} \right|^n \text{sgn} (\sigma - B)
\]

<table>
<thead>
<tr>
<th>Type of Simulation</th>
<th>Equations Needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monotonic Loading</td>
<td>X X X</td>
</tr>
<tr>
<td>Cyclic Hardening</td>
<td>X X X</td>
</tr>
<tr>
<td>Cyclic Softening</td>
<td>X X X</td>
</tr>
</tbody>
</table>

X - must use equation
x - may or may not use equation

Figure 2.17: Summary of Equations For Modeling Work Hardening

for producing the cyclic effects of softening (similar to the case of cyclic hardening). The simulation of figure 2.16 exhibits a slight amount of monotonic hardening but still has difficulty at the end of each monotonic section. This is because in each monotonic section the increasing back stress initially overrides the decreasing drag stress to produce a slight amount of monotonic hardening. However, by the end of each monotonic section, the decreasing drag dominates resulting in some monotonic softening.

A summary of the equations necessary for modeling the various types of loading is displayed in figure 2.17. A constitutive equation is always required for modeling, obviously. Monotonic simulations can be performed with either isotropic and/or kinematic equations, or neither. Cyclic hardening requires at least the isotropic equation. The kinematic equation can also be used if Bauschinger effect is prominent. Cyclic softening requires all three equations and is limited in its success. In general, cyclic softening should be avoided with the current formulation of the model.

2.1.6 Creep and Stress Relaxation

Evaluation of the model's performance in simulating creep and stress relaxation is the last section in the development of the model. As stated at the beginning of this study, the formulation of the elastic-viscoplastic equations does not have the simulations of creep and/or stress relaxation as a primary motive. Hence, the evaluation here is only to determine how the current model reacts under these conditions.
Creep

Creep describes how a material deforms under a constant load (stress). It governs the change of strain as a function of time under the condition of constant stress. In general, there are three stages to creep; primary (I), secondary (II), and tertiary (III). In primary creep, the creep-rate decreases rapidly. Secondary creep has a constant creep-rate. The creep-rate in the tertiary stage increases rapidly as fracture becomes imminent. Figure 2.18 displays these three stages.

Since the stress is constant for creep, the stress rate is zero. Therefore, the constitutive equation 2.8 degenerates to

$$\dot{\varepsilon} = \dot{\varepsilon}_o \left| \frac{\sigma - B}{D} \right|^n \text{sgn}(\sigma - B)$$

From here, we see that the total strain-rate, $\dot{\varepsilon}$, depends on the viscoplastic strain-rate, $\dot{\varepsilon}^{vp}$ (the power law, equation 2.7). Under elastic deformation, $\dot{\varepsilon}^{vp}$ is zero. Hence, creep can only occur if the material is plastically deformed.

Under the conditions of no hardening ($D$ is constant and $B$ is zero), the right side of equation 2.15 is constant and can be integrated. Solving for the strain yields

$$\varepsilon = \dot{\varepsilon}_o \left| \frac{\sigma}{D} \right|^n \text{sgn}(\sigma)t + \varepsilon_i$$

where $\varepsilon_i$ is the initial total strain at which the creep begins. Subtracting $\varepsilon_i$ from the total strain, $\varepsilon$, defines the creep strain, $\varepsilon_c$. Hence, we arrive at the creep formulation for no hardening.
Figure 2.19: Simulation of Creep Test Showing Stage I and Stage II

\[ \epsilon_c = \dot{\epsilon}_c \left| \frac{\sigma}{D} \right|^n \text{sgn}(\sigma) t \]  

Equation 2.16 is linear with time since \( D \) was constrained to be constant. However, the viscoplastic strain actually changes during creep because the elastic strain stays constant due to the constant stress state. Therefore, \( B \) and \( D \) could change as governed by equations 2.12 and 2.14, causing the creep to be nonlinear. A closed form solution, similar to equation 2.16, for equation 2.15 is not possible (or difficult at best) since the right side of equation 2.15 is not constant with time if hardening is considered. Under the conditions of work hardening (not softening\(^8\)), the solution of equation 2.15 will yield stage I and stage II creep. The linear portion, stage II, comes from the fact that \( B \) and \( D \) will eventually saturate to constant values and then solution of equation 2.15 would reduce to a form similar to equation 2.16. The net result of modeling creep with both isotropic and kinematic hardening is a solution in the form shown in figure 2.19.

This solution poses one major problem. The time scale in figure 2.19 is in seconds. In general, creep in metal occurs over many hours (hundreds). The simulation could be scaled by lowering the value of the saturation constant, \( \dot{\epsilon}_c \), since it controls the slope of the solution. Numerical solutions of equation 2.15 would be

\(^8\)Work softening would yield stage I creep that is concave up instead of the normal concave down.
more accurate if this is done, but this would also pose two major problems. First, numerical solutions take time. In general, it takes the computer (VAX8650) about 1/3 to 3 times the real test time to simulate actual creep behavior (depending on your numerical integration step size). Hence, lowering $\dot{\epsilon}_p$ enough to quantitatively simulate a 100 hour creep test would take between 33–300 hours. Secondly, lowering the saturation constant, $\dot{\epsilon}_o$, would make modeling of monotonic simulations very difficult. As discussed in Section 2.1.4, $\dot{\epsilon}_o$ is used in determining an initial drag value. As a result, it is recommended that $\dot{\epsilon}_o$ not be modified for creep and that the model be used to simulate creep only qualitatively.

**Stress Relaxation**

Stress relaxation is basically the opposite of creep. Here, the change of stress under constant total strain as a function of time is of interest. This generally results in an exponential decay of stress with time (figure 2.20).

Under the condition of constant strain, the strain-rate, $\dot{\epsilon}$, is zero and the constitutive equation 2.8 degenerates to

$$\dot{\sigma} = E\dot{\epsilon}_o \left[ \left( \frac{\sigma - B}{D} \right) \right]^n \text{sgn} (\sigma - B)$$

(2.17)

From here, we see that the stress depends on the viscoplastic strain-rate, $\dot{\epsilon}^{vp}$ (the power law, equation 2.7) and $E$. Under elastic deformation, $\dot{\epsilon}^{vp}$ is zero. Hence, stress relaxation can only occur if the material is plastically deformed. Simulation of a material that is strained into plastic deformation and then held at a constant total strain is displayed in figure 2.21. The first plot shows the stress decreasing at the constant strain value (the vertical line at 0.10% strain). The lower plots show
All simulations:

\[ E = 21.7 \times 10^6 \text{ psi} \]
\[ \dot{\varepsilon} = 1.0 \text{ sec}^{-1} \]
\[ \varepsilon_0 = 1.0 \text{ sec}^{-1} \]
\[ n = 20 \]
\[ a_1 = 50.0 \]
\[ b_1 = 1.0 \times 10^6 \text{ psi} \]
\[ a_2 = 400.0 \]
\[ b_2 = 2.5 \times 10^5 \text{ psi} \]
\[ B_1 = 0.0 \text{ psi} \]

Figure 2.21: Simulation of Stress Relaxation
the variation of the drag, back stress, and total stress with time. The total stress exhibits an exponential decay with time. Both the drag and back stress are constant during the elastic deformation. During plastic deformation, before a constant strain is imposed, both drag and back stress are increasing rapidly (appears almost linear, but is not). Once a constant strain is imposed, the drag and back stress increase slowly and eventually approach constant values. Even though the strain is held constant, the drag and back stress still increase, indicating that the viscoplastic strain rate, \( \varepsilon^{vp} \), is positive during stress relaxation. This seems strange, but can be explained as follows. Since the stress decreases during relaxation, the elastic strain, \( \varepsilon^e \) must decrease (see discussion at beginning of Chapter 2 near equation 2.1). Thus, the viscoplastic strain, \( \varepsilon^{vp} \), must increase to keep the total strain, \( \varepsilon \), constant. This in turn forces both the drag and back stress to increase during stress relaxation.

\[
\frac{\varepsilon}{\text{constant}} = \frac{\varepsilon^e + \varepsilon^{vp}}{\downarrow \uparrow} \Rightarrow D \uparrow B \uparrow
\]

The same problem regarding the time scale that exists in the creep model exists for stress relaxation. For the same reasons discussed in the creep model, it is recommended that the model be used to simulate stress relaxation qualitatively, not quantitatively.

### 2.2 Numerical Simulations of Real Materials

Now that the model is developed for the uniaxial case, simulations of real materials will be presented. The materials simulated are AISI 1040 Steel, Commercially Pure Titanium, and Annealed Type 304 Stainless Steel. Simulations of stress-strain curves at different strain-rates will be presented for all materials. Tension-compression cyclic loading of the Titanium and 304 Stainless is also shown. All the curves in this section will not be normalized since the original published test data is not normalized.

The types of equations used for fitting the model to the test data vary slightly depending on the amount and type of data available (e.g. with no cyclic data, kinematic hardening can not be used). The constants needed for the necessary equations were found through iterative techniques based on educated guesses. James [13] provides a good discussion on other methods that can be used to calculate the required material constants for such simulations. Since the type and amount of experimental data were different for each of the metals, the procedure used for simulation is described separately for each material.

The equations used are relatively simple. Exact agreement with actual material data is not expected. Only a good representation of the materials behavior is intended with these equations.
2.2.1 AISI 1040 Steel

The first material modeled is AISI 1040 Steel. The experimental data comes from Meyers [7] and is shown in figure 2.22. The data displays the tensile response of the steel at three strain rates and is plotted as engineering stress versus engineering strain. The maximum strain level of 10.0% in Meyers' data violates the maximum total strain assumption of this study. Applying the maximum total strain assumption, the engineering stress and strain data up to 2.0% can be considered the true stress and strain. As a result, only this portion of Meyers' data is simulated (figure 2.23). The dotted lines depicting Meyers' data were produced by scaling data points from an enlarged copy of figure 2.22.

Since no cyclic data is available, only isotropic hardening is applied. Thus, equations 2.8 and 2.12 are used to create the simulation ($\dot{B} = 0$ and $B = 0$). They are listed again for convenience.

\[ \dot{\sigma} = E \left( \dot{\varepsilon} - \dot{\varepsilon}_0 \right) \left[ \frac{\sigma - B}{D} \right]^n \text{sgn}(\sigma - B) \]

\[ \dot{D} = |\dot{\varepsilon}_p| (b_1 - a_1 D) \]

Meyers data is unusually sharp in the transition region. While not certain, this might be from the method he used to produce his plots. The over-square nature is beneficial, however. As seen in figure 2.23, the simulation is very accurate.

The constants for the equations are determined as follows.

1. Set $B$ to zero (no kinematic hardening).
2. Set $\dot{\varepsilon}_0$ equal to the average strain-rate, $\dot{\varepsilon}_2$.
3. Set the initial drag, $D_i$, equal to the elastic limit of curve 2. Set $a_1$ and $b_1$ to zero initially (no isotropic hardening).
4. Start with an initial guess for $n$ and vary the strain-rate. Change $n$ until the correct sensitivity is obtained.
5. Once a proper sensitivity is obtained, introduce isotropic hardening ($a_1$ and $b_1$).
6. Modify parameter values as needed.

From this method, a fast and reasonably accurate model is produced.

---

9This is discussed in Section 2.1.2.
Figure 2.22: AISI 1040 Steel Data of Meyers [7]

Figure 2.23: Simulation of AISI 1040 Steel
2.2.2 Commercially Pure Titanium

The experimental data for Commercially Pure Titanium comes from two papers published by Bodner. The data shows strain-rate sensitivity of tensile specimens [19] and cyclic behavior [27]. Bodner plots the tensile data [19] as true engineering stress and true engineering strain with a maximum strain value of 10.0%. Again, this violates the assumptions in this study. Thus, only data up to 2.0% total strain is simulated. Bodner's tensile data [19] is scaled by the same method used for Meyers' data. The cyclic data of Bodner [27] (displayed later in figure 2.25) is untouched (except for reduction during photocopying).

Since cyclic data is available and the Titanium exhibits the Bauschinger effect, both isotropic and kinematic hardening is applied. Thus, equations 2.8, 2.12, and 2.14 (repeated here) are used to create the simulation.

\[
\dot{\sigma} = E \left[ \dot{\epsilon} - \dot{\epsilon}_0 \left| \frac{\sigma - B}{D} \right|^n \text{sgn} (\sigma - B) \right]
\]

\[
\dot{D} = |\dot{\epsilon}^{sp}| (b_1 - a_1 D)
\]

\[
\dot{B} = \dot{\epsilon}^{sp} b_2 - |\dot{\epsilon}^{sp}| a_2 B
\]

The simulations of the deformation of Titanium are shown in figures 2.24 and 2.25. Two methods of simulating the monotonic curves of figure 2.24 are displayed. For plot A, it is assumed that there is no kinematic hardening and the guidelines listed previously for the AISI 1040 Steel data simulation are used. The simulation is over-square and not very good. Plot B uses both isotropic and kinematic hardening as suggested by the cyclic data of figure 2.25. The constants for plot B are also used in simulating the cyclic data. Hence, plot B was created in conjunction with figure 2.25. This simulation is more realistic and fits the published data better. As seen in figure 2.25, the addition of the kinematic hardening produced increased curvature (the model is not over-square) in the transition region between purely elastic deformation and plastic deformation. This arises from the fact that the kinematic equation 2.14 allows B to change. Since there is a large Bauschinger effect in the cyclic data, the rise time of B needs to be fast. This change of B in combination with the addition/subtraction effect with \(\sigma\) produces the smooth transition. The values for the constants in plots A and B of figure 2.24 are different because different equations are used for each of the two plots.

The cyclic data of figure 2.25 shows that the Titanium reaches cyclic saturation after only a few cycles. Also, it exhibits a very large Bauschinger effect. The quick cyclic saturation is not difficult to model but a large Bauschinger effect is. The simulation matches cyclic saturation behavior well, but has some difficulty with the Bauschinger effect. The simulation does show the effect prominently, but is still over-square in the lower right and upper left portions of the plot.

The constants for the equations were determined as follows:
**A) Isotropic Hardening**

- **Simulation**
- **Experimental**

\[
\sigma = 3.2 \times 10^{-3} \text{sec}^{-1}
\]

\[
\sigma = 1.6 \times 10^{-4} \text{sec}^{-1}
\]

\[
\sigma = 1.6 \times 10^{-5} \text{sec}^{-1}
\]

\[
E = 1.2 \times 10^{5} \text{MPa}, \ \dot{\varepsilon}_0 = 1.6 \times 10^{-4} \text{sec}^{-1}, \ n = 40,
\]
\[a_1 = 20.0, \ b_1 = 6.5 \times 10^{3} \text{MPa}, \ D_i = 2.85 \times 10^{2} \text{MPa}, \]
and \(B = 0.0 \text{MPa}\).

**B) Isotropic and Kinematic Hardening**

- **Simulation**
- **Experimental**

\[
\sigma = 3.2 \times 10^{-3} \text{sec}^{-1}
\]

\[
\sigma = 1.6 \times 10^{-4} \text{sec}^{-1}
\]

\[
\sigma = 1.6 \times 10^{-5} \text{sec}^{-1}
\]

\[
E = 1.2 \times 10^{5} \text{MPa}, \ \dot{\varepsilon}_0 = 3.2 \times 10^{-4} \text{sec}^{-1}, \ n = 30,
\]
\[a_1 = 18.0, \ b_1 = 4.2 \times 10^{3} \text{MPa}, \ D_i = 2.1 \times 10^{2} \text{MPa}, \]
\[a_2 = 400.0, \ b_2 = 3.4 \times 10^{4} \text{MPa}, \] and \(B_i = 0.0 \text{MPa}\).

**Figure 2.24: Two Simulations of Titanium at Different Strain Rates**

**Figure 2.25: Actual Cyclic Test Data and Numerical Simulation of Titanium For 10 Cycles**
1. Initially, for monotonic simulation, set $B$ to zero (no kinematic hardening).

2. Set $\dot{\varepsilon}_0$ equal to the average strain-rate, $\dot{\varepsilon}_2$.

3. Set the initial drag, $D_t$, equal to the elastic limit of curve 2. Set $a_1$ and $b_1$ to zero initially (no isotropic hardening).

4. Start with an initial guess for $n$ and vary the strain-rate. Change $n$ until the correct sensitivity is obtained.

5. Once a proper sensitivity is obtained, introduce kinematic hardening ($a_2$ and $b_2$). Simulate for one cycle.

6. Once proper kinematic hardening is obtained, introduce isotropic hardening ($a_1$ and $b_1$).

7. Modify parameter values as needed. Most likely, the initial guesses for $D_t$ and $\dot{\varepsilon}_0$ will need to be changed.

This method is slightly more cumbersome than the previous method but is needed for obtaining kinematic effects.

2.2.3 Annealed Type 304 Stainless Steel

The experimental data for Annealed Heat 9T2796 Type 304 Stainless Steel at 1100°F comes from two papers. Strain-rate sensitivity data for the 0.2% offset yield comes from Steichen [28]. Cyclic data is published by Corum [29], but strain-rate sensitivity data is not available from Corum. It is recognized that using data from two different sources increases the uncertainty in the data, but it is assumed that the overall behavior shown by Steichen and Corum's data is still valid.

Since cyclic data is available and the metal exhibits the Bauschinger effect, both isotropic and kinematic hardening is applied. Thus, the same equations used to model the Titanium are used for the 304 Stainless. Simulations are shown in figures 2.26 and 2.27. The constants used in figure 2.26 are also used in the cyclic model of figure 2.27.

The 0.2% offset strain-rate sensitivity data published by Steichen is in graph format and precise values are not readily accessible. The graph contained several plots of data for 304 Stainless at various temperatures. The closest temperatures to 1100°F (temperature of Corum's data) are 1000°F and 1200°F. Only the higher temperature data (1200°F) appeared to show much strain-rate sensitivity. Since it is unknown what 1100°F would produce for strain-rate sensitivity, it is assumed that it would be similar to the 1200°F data. As a result, only values depicting trends can be compared. Figure 2.26 shows that the 0.2% offset values of the simulation varied in the same range and with the same trends as that published by Steichen.

The cyclic data of figure 2.27 shows cyclic loading of 304 Stainless Steel for three different total strain ranges. Bauschinger effect is exhibited, but to a lesser
Experimental Data of Steichen[28] at Several Temperatures

Simulation at 1100°F

Strain, $\varepsilon$ (%)

0.00 0.20 0.40 0.60

Stress, $\sigma$ (x10^6 psi)

0.00 0.50 1.00 1.50 2.00

$\dot{\varepsilon} = 8.3 \times 10^{-1} \text{sec}^{-1}$

$\dot{\varepsilon} = 8.3 \times 10^{-2} \text{sec}^{-1}$

$\dot{\varepsilon} = 8.3 \times 10^{-3} \text{sec}^{-1}$

0.2% offset

All simulations: $E = 21.7 \times 10^6$ psi, $\dot{\varepsilon}_0 = 8.3 \times 10^{-8}$ sec$^{-1}$, $n = 30$, $a_1 = 10.0$, $b_1 = 3.0 \times 10^5$ psi, $D_i = 9.0$ ksi, $a_2 = 850.0$, $b_2 = 2.4 \times 10^5$ psi, and $B_i = 0.0$ psi.

Figure 2.26: Simulations of 304 Stainless Steel at Different Strain-Rates

degree than that for Titanium. In all the plots, the material approaches, but never reaches cyclic saturation within 10 cycles. In general, the simulations agree with the published data of Corum. There is some over-squareness in the corners (transition regions). For simulations A and B, the final stress value after 10 cycles is slightly higher than actual. Simulation C has a value slightly lower than actual. Since no final cyclic saturation value is available from the published data, the value for the ratio $b_1/a_1$ is estimated. This estimation combined with a slightly inaccurate rise time (controlled by $a_1$), could cause some of the cyclic discrepancies.

The constants for the equations were determined in the same manner as for the Titanium simulations.

### 2.3 Summary of Uniaxial Model

In this chapter, the constitutive equations governing isothermal elastic-viscoplastic behavior have been developed and evaluated for the uniaxial case. These equations are based on Hooke's law, the separability of the total strain into elastic and plastic components, and the separability of hardening into isotropic and kinematic quantities. Assumptions of constant temperature, small strains, and quasi-static loading have been made.

An important goal in developing the equations was to keep them simple enough to be understandable and applicable for numerical modeling. Qualitative and quantitative agreement of strain-rate dependence and work hardening behavior for a
Experimental Data of Corum[29] at 1100°F  Simulation at 1100°F

All simulations: $\dot{\varepsilon} = 8.3\times10^{-5}\text{sec}^{-1}$, $\dot{\varepsilon}_0 = 8.3\times10^{-5}\text{sec}^{-1}$, $n = 30$, $a_1 = 10.0$, $b_1 = 3.0\times10^5\text{psi}$, $D_1 = 9.0\text{ksi}$, $a_2 = 850.0$, $b_2 = 2.4\times10^6\text{psi}$, and $B_1 = 0.0\text{psi}$.

Figure 2.27: Actual Cyclic Test Data and Numerical Simulation of 304 Stainless Steel at 0.4%, 0.6%, and 1.0% Total Strain Range
general class of elastic-viscoplastic materials was the primary objective. Qualitative agreement of creep and stress relaxation was a secondary objective. It should be remembered that the equations created are relatively simple. Exact agreement with actual material data is not expected. Only a good representation of the materials behavior is intended with these equations.

The formulation consists of three coupled differential equations; a power law measuring viscoplastic strain-rate and two first order equations for isotropic and kinematic hardening. A summary of the governing equations is as follows:

**Viscoplastic Strain Rate**

\[ \dot{\varepsilon}^{vp} = \dot{\varepsilon}_o \left| \frac{\sigma - B}{D} \right|^n \text{sgn}(\sigma - B) \]  

(2.7)

**Constitutive Equation**

\[ \dot{\sigma} = E \left[ \dot{\varepsilon} - \dot{\varepsilon}_o \left| \frac{\sigma - B}{D} \right|^n \text{sgn}(\sigma - B) \right] \]  

(2.8)

**Isotropic Hardening Equation**

\[ \dot{D} = |\dot{\varepsilon}^{vp}|(b_1 - a_1 D) \]  

(2.12)

**Kinematic Hardening Equation**

\[ \dot{B} = \dot{\varepsilon}^{vp} b_2 - |\dot{\varepsilon}^{vp}| a_2 B \]  

(2.14)

From the simulations and analysis presented, the following conclusions can be drawn for uniaxial modeling of elastic-viscoplastic behavior. The equations effectively model:

- Strain-rate sensitivity, both qualitatively and quantitatively
- Monotonic and cyclic loading (cyclic hardening), both qualitatively and quantitatively
- Isotropic and kinematic hardening, both qualitatively and quantitatively
- First and second stage creep, only qualitatively
- Stress relaxation, only qualitatively

The equations have difficulty in the following areas.

- They model only a range of strain-rate.
- Models are often over-square (when kinematic hardening is large over-squareness is lessened).
- Cyclic softening is difficult and not recommended.
- Nonlinear elastic materials such as Aluminum are difficult to model and not recommended.
Chapter 3

MULTIDIMENSIONAL ANALYSIS

This chapter extends the constitutive equations of chapter 2 (excluding kinematic hardening) to multidimensional forms\(^1\). These multidimensional forms will allow for the study and investigation of real structures under real loading conditions. The finite element method is then introduced as a means of solving actual continuum problems by creating approximate discrete solutions. Finally, a numerical example, compression of a constrained cylinder, is solved to demonstrate the capabilities of the constitutive equations when implemented into a finite element algorithm. For all the equations developed and discussed in this chapter, the assumptions of Chapter 2 remain; namely, all deformations are quasi-static, isothermal, and small.

3.1 Development of Elastic-Viscoplastic Constitutive Equations in Multiple Dimensions

This section develops and briefly describes the constitutive equations in their multidimensional forms. These equations are based on the same principles as those presented in the one-dimensional analysis of Chapter 2.

The origin of the derivation is Hooke’s law for linear elasticity (equation 2.1 for 1-D). This is represented in a matrix form as

\[
[\sigma] = [E] [\varepsilon] \tag{3.1}
\]

where \([\sigma]\) and \([\varepsilon]\) are the commonly known stress and elastic strain tensors, respectively. The elastic stiffness\(^2\) is defined by the matrix \([E]\) and contains values for the

---

\(^{1}\)A multidimensional form can govern 1-D, 2-D, and/or 3-D analysis.

\(^{2}\)The elastic stiffness is often denoted by the matrix \([D]\). It is denoted as \([E]\) in this study in order to avoid confusion with the drag stress, \(D\), defined previously.
modulus of elasticity, \( E \), and Poisson's ratio, \( \nu \). The strain relationship, equation 2.2 in 1-D, is necessary for the evaluation of elastic and plastic deformation. It is expressed in a matrix form as

\[
[\epsilon] = [\epsilon^e] + [\epsilon^p]
\]  
(3.2)

Combining these two equations and taking a time derivative leads to

\[
[\dot{\sigma}] = [E] ([\dot{\epsilon}] - [\dot{\epsilon}^p])
\]  
(3.3)

where \([E]\) is considered constant with respect to time. This is easily seen to be the matrix form of equation 2.4.

The viscoplastic strain-rate matrix, \([\dot{\epsilon}^p]\), is related to the stress through the flow rule of classical plasticity [5]. The flow rule is expressed as

\[
\dot{\epsilon}^p_{ij} = \dot{\lambda} S_{ij}
\]  
(3.4)

where \( S_{ij} \) is the *deviatoric stress tensor* defined by

\[
S_{ij} = \sigma_{ij} - \sigma_m \delta_{ij}
\]

and

\[
\sigma_m = \frac{1}{3} (\sigma_{11} + \sigma_{22} + \sigma_{33})
\]

The deviatoric stress tensor, \( S_{ij} \), represents all the shear stresses and therefore causes plasticity (distortion). The *hydrostatic* or *mean stress*, \( \sigma_m \), involves only pure tension or compression and produces volume changes only. The symbol \( \delta_{ij} \) is the Kronecker delta. In simple terms, equation 3.4 states that plastic deformation is produced by the deviatoric stress only. The hydrostatic pressure causes no plasticity.

The function \( \dot{\lambda} \) is not defined yet and is derived in the following manner. First equation 3.4 is squared and both sides are premultiplied by \( \frac{3}{2} \) to obtain

\[
\frac{3}{2} \dot{\epsilon}^p_{ij} \dot{\epsilon}^p_{ij} = \frac{3}{2} \dot{\lambda}^2 S_{ij} S_{ij}
\]  
(3.5)

Next, two quantities, the *effective stress* and the *effective viscoplastic strain-rate* are defined as

\[
\sigma^{\text{eff}} = \sqrt{\frac{3}{2} S_{ij} S_{ij}}
\]  
(3.6)

\[
\dot{\epsilon}^p = \sqrt{\frac{2}{3} \dot{\epsilon}^p_{ij} \dot{\epsilon}^p_{ij}}
\]  
(3.7)

Using these definitions in equation 3.5 and solving for \( \dot{\lambda} \) yields

\[^3\text{For notational simplicity, tensor notation will be used for the development of the viscoplastic strain-rate tensor (ie. } [\dot{\epsilon}^p] = [\dot{\epsilon}^p_{ij}].\]
\[ \lambda = \frac{3 \dot{\varepsilon}_{vp}^{\varepsilon}}{2 \sigma_{\text{eff}}} \]  

(3.8)

Taking this definition of \( \lambda \) and substituting into equation 3.4 produces the following form of the viscoplastic strain-rate:

\[ \dot{\varepsilon}_{ij}^{vp} = \frac{3 \dot{\varepsilon}_{vp}^{\varepsilon}}{2 \sigma_{\text{eff}}} S_{ij} \]  

(3.9)

Choosing a form similar to the 1-D power law (equation 2.6) for representation of the effective viscoplastic strain-rate, \( \dot{\varepsilon}_{vp}^{\varepsilon} \), yields

\[ \dot{\varepsilon}_{vp}^{\varepsilon} = \dot{\varepsilon}_o \left( \frac{\sigma_{\text{eff}}}{D} \right)^n \]  

(3.10)

where the parameters \( n, \dot{\varepsilon}_o \) and \( D \) are defined in Chapter 2. Kinematic hardening (back stress, \( B \)) is not expanded into a multidimensional form in this study. Multidimensional formulations do exist for kinematic hardening but are more complex because kinematic hardening is directional. This directionality requires that the kinematic hardening be represented by a matrix, unlike the isotropic drag, \( D \), which is still represented by a scalar (drag is not directional) in multidimensional analysis.

Substituting equation 3.10 into equation 3.9 provides the final multidimensional form of the viscoplastic strain-rate:

\[ \dot{\varepsilon}_{ij}^{vp} = \frac{3}{2} \dot{\varepsilon}_o \left( \frac{\sigma_{\text{eff}}}{D} \right)^n S_{ij} \sigma_{\text{eff}} \]  

(3.11)

where it is noted that if \( \sigma_{\text{eff}} = 0 \), then \( \dot{\varepsilon}_{ij}^{vp} = 0 \). Substituting equation 3.11 into equation 3.3 produces the complete three-dimensional matrix form of the constitutive equation for elastic-viscoplastic behavior.\(^4\)

\[ [\sigma] = [E] \left( [\varepsilon] - \frac{3}{2} \dot{\varepsilon}_o \left( \frac{\sigma_{\text{eff}}}{D} \right)^n \frac{1}{\sigma_{\text{eff}}} [S] \right) \]  

(3.12)

The first thing to note is that equation 3.12 degenerates exactly to equation 2.8 under the conditions of an uniaxial (1-D) stress state. Under this condition, the effective stress degenerates to the absolute value of the uniaxial stress, \( |\sigma| \), and the deviatoric stress becomes \( 2/3\sigma \) (including the correct sign). Then, the ratios of \( 3/2 \) and \( 2/3 \) cancel and the deviatoric stress divided by the effective stress becomes \( \text{sgn}(\sigma) \). This result is exactly the form developed in Chapter 2.

Also noted is the role of the effective stress, \( \sigma_{\text{eff}} \). The effective stress equals the von Mises Stress which comes from the Distortion Energy Theory. This theory states that yielding begins when the distortion energy is equal to the distortion energy at yield in simple tension. Since distortion is caused by deviatoric stress, this

\(^4\)The matrix \([S]\) is exactly the same as the indicial notation of \( S_{ij} \).
is consistent with the original statement of the flow rule of plasticity, equation 3.4, which states (in simple terms) that plastic deformation is caused by deviatoric stresses. In equation 3.12, plasticity (yielding) occurs when the effective stress approaches the drag stress, \(D\). The exact value of \(\sigma_{\text{eff}}\) for which plasticity occurs depends on the same parameters as those discussed in Sections 2.1.1 and 2.1.4.

As noted previously, only isotropic hardening is presented in this chapter. Since isotropic hardening is uniform in all directions, it is represented by a single scalar equation (i.e. no matrix is needed). Use of equation 2.12 from 1-D analysis poses a problem in that \(\dot{\varepsilon}^p\) is not defined as a single quantity in multiple dimensions. This problem is cured by replacing \(\varepsilon^p\) with the effective viscoplastic strain-rate, \(\dot{\varepsilon}^p\) (equation 3.10). The effective viscoplastic strain-rate degenerates to the magnitude of the one dimensional viscoplastic-strain rate under the conditions of an uniaxial stress state. Also, the effective viscoplastic strain-rate is uniform in all directions because it is a scalar. Hence, isotropic hardening in multidimensional analysis is governed by

\[
\dot{D} = \dot{\varepsilon}^p (b_1 - a_1D) \tag{3.13}
\]

For solution of boundary value problems (actual structures), the constitutive equations alone are not enough. They must be used in conjunction with the equilibrium equations of continuum mechanics. Since our equations are nonlinear and actual structures usually have complicated geometry and loading conditions, closed form solutions are difficult and rare. As a result, the method of finite elements is chosen for solving these boundary value problems.

### 3.2 Finite Element Implementation

This section discusses the finite element method and its implementation for modeling elastic-viscoplastic materials. Since the derivation of the basic equations used in the finite element method is rather extensive, only the major highlights are listed in this chapter. The reader is directed to Appendix B and the listed references for further detail.\(^6\)

The finite element method is an approximate numerical technique used in solving a wide variety of boundary value problems. In particular, engineers use finite elements to solve problems in the fields of continuum mechanics and heat transfer to name just a few. Other approximate numerical techniques such as finite difference have been developed to solve these problems also. Since finite difference creates difference equations for an array of grid points, the method has difficulty with the irregular geometries and/or unusual (nonuniform) boundary conditions that are often found in real structures [30]. Finite elements on the other hand divides the

\(^5\)No absolute value sign is applied to the effective viscoplastic strain-rate since it is always a positive quantity.

\(^6\)Equations listed in this section are listed in Appendix B with different equation numbers.
continuum into several interconnected subregions called elements. Approximation
functions known as shape functions (often polynomials) are then created for the
elements and incorporated into the variational form of the governing equations.
Assemblage of all the discrete elements then provides a piecewise approximation
to the governing equations which is capable of modeling the complex shapes and
boundary conditions found in real structures.

Using the finite element method, solutions to boundary value problems proceeds in an orderly step-by-step manner. The basic outline consists of discretizing the domain, selecting the shape functions, developing the element equations, as-
sembling the element equations into global (system) equations, solving the global
equations, and finally calculating additional results as desired (post processing such as stresses).

Starting with the principle of virtual work, the finite element formulation is
derived. However, since our constitutive equations are rate dependent, the virtual
work principle is modified slightly into a rate formulation. In the case of isothermal
quasi-static loading and negligible body forces, the principle of the rate of virtual
work for an element becomes

\[
\int_{\Omega} [\delta \epsilon]^T [\dot{\sigma}] \, d\Omega = \int_{S} [\delta U]^T [\dot{F}] \, dS \quad (3.14)
\]

Rate of Virtual Strain Energy \quad Rate of Virtual External Work

where \([\delta \epsilon]^T\) is the transpose of the variation of the strain, \([\dot{\sigma}]\) is the stress rate, \(\Omega\) is
the domain of the element (volume or area), \([\delta U]^T\) is the transpose of the variation
of the displacements, \([\dot{F}]\) is the rate of external forces, and \(S\) is the surface. This is
known as a weak or variational formulation. The formulation is called weak because
it need only be satisfied as an average value as denoted by the integral.

Combining the constitutive equation (equation 3.12), the shape functions (Ap-
pendix B), and equation 3.14 will eventually lead to the final matrix form of the
element equations.

\[
[K] \begin{bmatrix} \dot{U} \end{bmatrix} = \begin{bmatrix} \dot{F}^{\text{vp}} \end{bmatrix} + \begin{bmatrix} \dot{F}^{\text{ext}} \end{bmatrix} \quad (3.15)
\]

where:

\[
[K] = \int_{\Omega} [C]^T [E] [C] \, d\Omega
\]

\[
[\dot{F}^{\text{vp}}] = \int_{\Omega} [C]^T [E] [\dot{\epsilon}^{\text{vp}}] \, d\Omega
\]

\[
[\dot{F}^{\text{ext}}] = \int_{S} [\psi]^T [\dot{F}] \, dS
\]

\(^7\)These and the other matrix quantities are listed in Appendix B.
The matrix \([K]\) is the element stiffness and \([\dot{\mathbf{u}}]\) denotes the rates of nodal displacement for the element. The force rates on the right side represent the viscoplastic forces and external forces, respectively. The matrix \([\psi]\) contains the shape functions and the strain-displacement matrix\(^8\) is \([C]\).

The viscoplastic force-rate is a fictitious force-rate in the sense that it is not a physically applied force rate. It arises from a mathematical manipulation during the combining of equations 3.12 and 3.14. This type of manipulation is common in many nonlinear finite element algorithms. The viscoplastic term differentiates the purely elastic problem from the elastic-viscoplastic problem.

Global equations are assembled from these element equations using the standard methods such as those found in [31]. Once the global system is created, a solution is calculated using an iterative strategy. The iterations are necessary because of the nonlinear nature of the viscoplastic force-rate.

The solution also requires integration in space and time. The spacial integration comes from the integrals in the equations above. Since spacial integration is performed over two or three variables, it is often computed numerically using Gaussian Quadrature. Gaussian Quadrature evaluates the integral at special points known as Gaussian points and then adds up all the evaluations according to a specific weighting procedure. This method is discussed in [32]. In order to integrate in time, the exact time derivatives are approximated using any one of a number of numerical integration schemes commonly used for integration with respect to one variable (time). The solution advances in time by small time steps, \(\Delta t\). Using this time marching, an incremental solution is found. The procedure for obtaining the incremental solution in any given time step is as follows.

1. Increment time by a small time step, \(\Delta t\).
2. Solve the system of equations (spacial integration) with the viscoplastic force-rate equal to zero.
3. Calculate the viscoplastic force-rate using the newly acquired displacements, strains, and stresses.
4. Re-solve the system of equations (spacial integration) including the viscoplastic force-rate.
5. Check convergence with specified criteria.
6. If no convergence, return to step 3 and repeat.
7. If convergence, return to step 1 and repeat.

\(^8\)The strain-displacement matrix is often denoted by the matrix \([B]\). It is denoted as \([C]\) in this study in order to avoid confusion with the back stress, \(B\), defined previously.
Convergence criteria vary, but they usually check the absolute and relative differences between quantities in two successive iterations within a given time step. Once convergence occurs, the next time increment is applied and the method is repeated.

The displacements (also strains, stresses, etc.) are incremental values. The total value of any variable at any time \( t_i \) is obtained by adding the incremental value calculated at time \( t_i \) to the previous total value at time \( t_{i-1} \).

Using the technique outlined, many multidimensional elastic-viscoplastic problems can be investigated and solved. In order to demonstrate the capabilities of the constitutive equations when implemented into a finite element algorithm, a numerical example, compression of a constrained cylinder, is solved.

### 3.3 A Numerical Example

In this section, simulations of the compression of a constrained cylinder by uniformly applied end displacements demonstrate the implementation of the constitutive equations in a finite element algorithm. The intent here is not to investigate the behavior of the solution, but rather to demonstrate the capabilities of the method. The demonstration consists of three parts: an elastic solution to show general stress behavior, elastic-viscoplastic solutions to show strain-rate effects, and elastic-viscoplastic solutions to show isotropic work hardening effects. For all of the solutions shown, the stresses are normalized with respect to the initial drag stress, \( D_i \), which is assumed to be the same for the entire cylinder.

This problem is chosen for several reasons. Symmetry in the geometry and boundary conditions of the cylinder allow for some simplification from a full 3-D analysis to a 2-D axisymmetric analysis. More important, the problem itself is realistic. The compression of a cylinder between two plates that have friction causes the ends to be constrained from motion in the radial direction. This constraining produces stress concentrations at the outer edges (top and bottom) of the cylinder. These stress concentrations imposed on the nominal stresses create stress variations throughout the cylinder. Since there are variations in more than one direction, predicting the resulting behavior requires multidimensional analysis.

Finite element modeling of the problem (in this study) is performed using isoparametric four noded elements in a 2-D domain. The term isoparametric comes from the combination of the words iso and parametric. Iso means that both geometry and response (displacements) are represented by the same shape functions. Parametric means that the element is mapped back (by the shape functions) onto a biunit square domain, known as the parametric domain, for all integrations and evaluations. The element used is also bilinear meaning that the displacements are a product of two linear terms in the parametric domain. The two linear terms come from the definition of the shape functions. These features make it possible to have nonrectangular quadrilateral elements which are often needed for irregular
All simulations in this chapter: $E = 21.7 \times 10^6$ psi, $\dot{\varepsilon}_0 = 1.0$ sec$^{-1}$, $n = 20$, and $\nu = 0.33$.

Figure 3.1: Actual Constrained Cylinder and Its Axisymmetric Representation

gometry. The detailed equations for this element can be found in Appendix B.

A finite element program, FEPROG, is written in FORTRAN to solve the axisymmetric problem. The program, originally written by Ghoneim [5], has been updated and revised by this author to include (among other things) the capability of isotropic work hardening. The program solves axisymmetric problems under the condition of imposed displacements. The program's accuracy is verified in two steps. First, purely elastic solutions for the compression of a constrained cylinder are created by FEPROG and compared against solutions created by a widely accepted commercial code called ANSYS. Secondly, elastic-viscoplastic solutions for compression of an unconstrained cylinder are compared against the 1-D solutions of VISCO (the 1-D program). Since the latter verification is 1-D, the exact same solutions are reached whether one element or a hundred elements are used. A listing of FEPROG can be found in Appendix C.

A diagram of the actual constrained cylinder and its finite element representation is shown in figure 3.1. Due to the symmetry, a quarter slice of the cylinder is modeled to obtain a complete representation of the entire cylinder. The mesh shown is chosen because it produces adequate results for demonstration of the technique. Other meshing schemes which include refinement near the upper right hand corner

---

9Compression of an unconstrained cylinder is an 1-D (uniaxial) analysis.
would better represent the stress concentration. However, the goal here is only to
demonstrate the method and not thoroughly investigate the problem. This model
is used for all of the following solutions in this chapter.

### 3.3.1 Elastic Solution

In order to obtain a basic understanding of the problem and a base line for com-
parisons, an elastic solution of the problem is necessary. Figure 3.2 depicts the
four stress quantities found in the constrained cylinder under compression. They
are axial stress, \( \sigma_z \); radial stress, \( \sigma_r \); hoop stress, \( \sigma_\theta \); and shear stress, \( \tau_{rz} \). As is
the case for all the contour plots in this chapter, the \( x \) axis represents the radial
direction and the \( z \) axis represents the axial direction. In all the plots of figure 3.2,
the magnitude of the stresses are maximum in the upper right hand corner where
the stress concentration occurs.

Combining these stresses according to the recipe defined by equation 3.6 pro-
duces the effective stress contour displayed in figure 3.3. Again, the maximum stress
magnitude is where the stress concentration occurs. The effective stress, always a
positive quantity, is the main factor in determining the viscoplastic strain-rate as
noted in the discussion of equation 3.12. As a result, stress contours of the effective
stress provide insight into which portion of the model is elastic and which is plastic.
For the solution shown, all of the model is elastic and this is not a factor. However,
for the rest of the models which are elastic-viscoplastic, the effective stress is of
interest.

### 3.3.2 Strain-Rate Effects For Elastic-Viscoplastic Model

The elastic-viscoplastic formulation models the strain-rate dependency of a material.
In 1-D, it was easy to demonstrate this by monitoring the stress, \( \sigma \), and the strain,
\( \epsilon \), for various strain rates, \( \dot{\epsilon} \). In the 2-D axisymmetric case, there are four stresses
and four strains to monitor. The effective stress provides information on strain-rate
dependency in the form of stress contours. However, these contours are pictures at
a particular time or strain and we would need several contours at different strains
to obtain a continuous (fluent) representation.

As a result, engineering strain and engineering stress are chosen as additional
quantities to monitor since they provide a continuous representation and can actu-
ally be measured during an experiment. The engineering strain is defined as the
change in axial length over the original axial length. The engineering strain-rate
for monotonic loading is simply the engineering strain divided by the time. The
engineering stress is defined as the applied load in the axial direction divided by
the original cross sectional area of the cylinder. Since only enforced displacements
are used in the model, the applied load is not directly available and must be calcu-
lated. Since the applied displacements are on the end boundaries (top and bottom
of cylinder) in the axial direction, the force across any given radial plane (plane
Figure 3.2: Contours of The Four Stress Components For an Elastic Compression of 0.01% Strain
Figure 3.3: Resulting Effective Stress For an Elastic Compression of 0.01% Strain perpendicular to z axis) is the same. Thus, summing up all the axial forces for a row of elements in a radial plane provides the total axial force. The axial force within any given element is found by dividing the axial Gauss point stress in the element, \( \sigma_z \), by the discrete area on which the stress acts.

Variations in the engineering stress resulting from the compression of a cylinder at three engineering strain-rates are depicted in figure 3.4. Similar conclusions can be drawn as those stated in Chapter 2 for the similar 1-D figure (figure 2.2). Despite the overall similarity, there are some differences with regard to some values in figures 3.4 and 2.2. For plot B in figure 3.4, where the engineering strain-rate is equal to the saturation constant, \( \dot{\varepsilon}_e \), the normalized compressive engineering stress is *slightly greater* than the value 1.0. In the 1-D case, the normalized engineering stress\(^{10}\) value was *exactly equal* to 1.0. This discrepancy is due to the 2-D effects resulting from the constrained ends on the cylinder. Also, the comparison of \( \dot{\varepsilon}_e \) against only the engineering strain-rate in the 2-D analysis is not completely equivalent to comparing \( \dot{\varepsilon}_e \) against the true strain-rate from the 1-D analysis.

For these solutions, the initial time step is chosen large enough to almost span the entire elastic region in one step because the elastic solution is linear. After that, several small time steps are required because the nonlinear contributions of the

\(^{10}\)Under the assumptions stated in Chapter 2, the engineering stress equals the true stress for 1-D analysis only.
Figure 3.4: Strain Rate Effects For 2-D Model
viscoplastic strain-rate in the equations becomes pronounced. For the simulations that progress up to a maximum engineering strain of 0.15% at an engineering strain-rate of 1.0 sec\(^{-1}\), 264 time steps are used. Once plasticity starts to occur, roughly ten iterations per time step are required to obtain convergence. Convergence is based on obtaining four significant digits in effective stress values.

To see the stress variations within the cylinder, effective stress contours are employed. Figure 3.5 displays the effective stress contours for the three strain-rate simulations at 0.15% engineering strain. These plots correspond to a snapshot in time of the cylinder. As depicted, the simulations with higher strain-rates have higher overall stress contour values. The normalized effective stress value determines the regions of elasticity and plasticity within the contour plot. In plots A and C, it is difficult to define the regions exactly since their ratios of engineering strain-rate to saturation constant, \(\varepsilon_a\), are not equal to 1.0. For plot B, where this ratio is 1.0, it is assumed that the region of plasticity begins when the normalized effective stress is slightly greater than 1.0. An exact value of effective stress for defining the regions is not known for the reasons discussed previously in this section.

### 3.3.3 Isotropic Hardening For Elastic-Viscoplastic Model

The demonstrations of isotropic hardening in this section show the effects of work hardening on the engineering stress and the effective stress. The equation used to simulate isotropic hardening is equation 3.13. For all the plots in this section, the engineering strain-rate and saturation constant, \(\varepsilon_a\), are both equal to 1.0 sec\(^{-1}\). Hence, the plastic region is defined by values of the normalized effective stress that are slightly greater than 1.0.

Figure 3.6 depicts how the engineering stress in the 2-D model changes with various values of hardening parameters. The variations produce similar results to those investigated in the 1-D model of Chapter 2. Discrepancies (between 1-D and 2-D models) similar to those found during the strain-rate investigation are found here also. Figures 3.7 and 3.8 depict the normalized effective stress without hardening and with hardening at four discrete engineering strain levels during the simulation. The contours of figure 3.7 become constant somewhere after 0.05% engineering strain while the the contours in figure 3.8 are continuously changing due to work hardening. As expected, the plastic region, designated by normalized effective stress values greater than 1.0, is continuously growing for the case of work hardening.

Figure 3.9 demonstrates the variation in effective stress for various sets of hardening parameters. The rise time, controlled by \(a_1\), varies from a high value for the top contour plot to a low value for the bottom contour plot. All three contours have the same steady state value of drag, \(b_1/a_1\). As expected, the solutions with the faster rise times have the largest regions of plastic deformation.
Figure 3.5: Effective Stress Contours at 0.15% Engineering Strain For Engineering Strain Rates of 10.0 sec\(^{-1}\), 1.0 sec\(^{-1}\), and 0.1 sec\(^{-1}\)
Figure 3.6: Variation of Drag Parameters For 2-D Model
Parameters: $\dot{\varepsilon}_{\text{eng}} = 1.0\text{sec}^{-1}$, $D=$constant.

Figure 3.7: Effective Stress Contours Without Hardening at Engineering Strain Levels of 0.04%, 0.05%, 0.10%, and 0.15%
Parameters: $\dot{\varepsilon}_{\text{eng}} = 1.0\text{sec}^{-1}$, $a_1 = 100.0$, $b_1 = 2.0 \times 10^6\text{psi}$.

Figure 3.8: Effective Stress Contours With Isotropic Hardening at Engineering Strain Levels of 0.04%, 0.05%, 0.10%, and 0.15%
A) \( a_1 = 100.0, b_1 = 2.0 \times 10^6 \text{psi} \)

B) \( a_1 = 50.0, b_1 = 1.0 \times 10^6 \text{psi} \)

C) \( a_1 = 10.0, b_1 = 2.0 \times 10^5 \text{psi} \)

Key:

\[
\begin{align*}
.465 & = A \\
.518 & = B \\
.571 & = C \\
.624 & = D \\
.677 & = E \\
.730 & = F \\
.783 & = G \\
.836 & = H \\
.889 & = I \\
.942 & = J \\
.995 & = K \\
1.05 & = L \\
1.10 & = M \\
1.15 & = N \\
1.21 & = O
\end{align*}
\]

All simulations: \( \varepsilon_{\text{eng}} = 1.0 \text{sec}^{-1} \).

Figure 3.9: Effective Stress Contours at 0.15% Engineering Strain For Various Sets of Drag Parameters
Chapter 4

CONCLUSIONS

A constitutive model has been proposed to simulate the isothermal quasi-static mechanical behavior of elastic-viscoplastic materials subject to small deformations. The constitutive equations are based upon Hooke's law, the separation of the total strain into elastic and viscoplastic quantities, and the separation of work hardening into isotropic and kinematic quantities. The formulation consists of three coupled differential equations; a power law measuring viscoplastic strain-rate and two first order equations simulating isotropic and kinematic hardening.

An important goal in developing the equations was to keep them simple enough to be comprehensible and applicable for numerical modeling. The basic construction of the model was based on macroscopic physical behavior but does have roots to microscopic physical mechanisms. Qualitative and quantitative agreement of strain-rate dependence and work hardening behavior for a general class of elastic-viscoplastic materials was the primary objective. Qualitative agreement of creep and stress relaxation was a secondary objective. Exact agreement with actual material data was not expected. Only a good representation of the material's behavior was intended with these equations.

The constitutive equations were initially developed for the one-dimensional case. The viscoplastic strain-rate, governed by a power law, was assumed a function of uniaxial stress, back stress, and drag stress. The equations simulating isotropic hardening and kinematic hardening were considered dependent on the uniaxial viscoplastic strain-rate. VISCO, a program written in ACSL, was created to numerically solve the resulting nonlinear constitutive equations for various 1-D simulations. Several uniaxial simulations, including simulations of actual published material data for AISI 1040 Steel, Commercially Pure Titanium, and Annealed Type 304 Stainless Steel, were performed numerically. Study of these simulations revealed that the equations qualitatively and quantitatively model strain-rate sensitivity, monotonic and cyclic loading, isotropic hardening, and kinematic hardening (if not extremely severe). Creep and stress relaxation were simulated only qualitatively because of a large discrepancy in the time scale. The constitutive equations were also found to effectively govern only a finite (small) range of strain-rates. For a large range
of strain-rates, multiple sets of material constants may be required. Other difficulties and limitations included over-square behavior of models, unrealistic cyclic softening behavior, and a restriction to modeling materials with linear elastic behavior. Nonlinear elastic materials such as Aluminum are difficult to model and not recommended.

The constitutive equations were then expanded, excluding kinematic hardening\(^1\), into multidimensional forms for implementation into a finite element algorithm. The flow rule was adopted to expand the viscoplastic strain-rate into a multidimensional power law. The many components of the stress and viscoplastic strain-rate in two- and three-dimensions required several modifications to the one-dimensional forms of the equations. The viscoplastic strain-rate became a function of effective stress and drag stress. The equation simulating isotropic hardening was considered dependent on the effective viscoplastic strain-rate. These general multidimensional forms did degenerate to the uniaxial equations originally developed.

For solution of boundary value problems, the constitutive equations were implemented into a finite element algorithm. The finite element formulation developed was time dependent because of the rate dependency in the constitutive equations. Treatment of the nonlinear term resulting from the viscoplastic power law came in the form of a fictitious force-rate. Numerical solution of the resulting equations required an incremental iterative solution scheme that calculated solutions at several time increments. Applying the finite element method with time marching created a comprehensive technique that is capable of solving many actual elastic-viscoplastic engineering problems.

Demonstration of this capability was shown by solving the compression of a constrained cylinder under uniformly applied end displacements. Numerical solution with a FORTRAN program, FEPROG, displayed the ability to predict strain-rate dependency and isotropic hardening along with general two- and three-dimensional phenomena such as stress concentrations. The important role of the effective stress for multidimensional analysis was seen as a major factor for determining when plasticity begins and where plasticity has occurred within a specimen.

Future work can be directed in many areas. Elimination of the over-square behavior in the equations would produce increased accuracy in modeling. This would allow for better modeling of many real materials, such as Commercially Pure Titanium, which do not have an abrupt change in transition from elasticity to plasticity. Expanding kinematic hardening into multidimensions would create a more robust package to model real materials in two- and three-dimensions. Lastly, for practical use in industry, the constitutive equations developed need to be implemented into a commercial code. Use of a commercial code would provide much greater modeling capabilities than can be achieved by creating individual programs. One suggestion for its implementation is through the use of the DMAP command language of

---

\(^1\)Kinematic hardening is direction-dependent and, therefore, requires a matrix for multidimensional representation which adds increased difficulty and complexity into the analysis.
MSC/NASTRAN.
References


Appendix A

PROGRAM FOR ONE DIMENSIONAL ANALYSIS, VISCO

The following program is used for one dimensional analysis. The program numerically solves the 1-D form of the constitutive equation. VISCO solves stress as a function of total strain. Simulations of strain-rate jump tests, tension-compression cyclic loading, work hardening, creep tests, and stress relaxation tests are also available.

The main program, VISCO, is written in ACSL. ACSL is a program that takes a model definition (main program VISCO), and translates it into numerous FORTRAN statements which are incorporated into the ACSL programs and subroutines. ACSL [17] uses a fourth order Runge-Kutta integration scheme and is capable of solving simultaneous equations which is needed when work hardening is introduced.

The model definition is written using statements similar to FORTRAN but not exactly like FORTRAN. One noticeable difference is that IF statements are allowed but IF-ENDIF blocks are not. As a result, a subroutine, EPSILON, was written because numerical simulation of cyclic loading required an IF-ENDIF block. It should be noted that the subroutine can be (and is) written in FORTRAN.
"VISCOPLASTICITY MODEL"
"NUMERICAL MODELING OF VISCOPLASTIC BEHAVIOR"
"USING POWER FORMULA & ITS DERIVATIVES"
"PROGRAM NAME: VISCO"
"WRITTEN BY: TED DIEHL"
"REVISED : 4/25/88"

Program models 1-D elastic/plastic behavior by evaluating
the stress rate as a function of total and viscoplastic strain
rates. Cyclic loading is accomplished by reversing the strain
rate when a predetermined maximum strain level is reached. A
counter is also included to determine how many cycles have been
performed. A strain rate jump test, stress relaxation test, and
creep test can also be performed using this numerical model.

Hardening is modeled by both Isotropic and/or Kinematic.
Isotropic work hardening is modeled using 3 different models:
1. Time dependent work hardening
2. Total strain dependent work hardening
3. Accumulative plastic strain work hardening
Kinematic work hardening is modeled using a model similar to
no. 3 from Isotropic work hardening. Haislers model (KIN=0)" produces incorrect behavior. The correct model is KIN=1.

For JUMP TEST
set JUMP = 1 (every time jump test is run)
set CMAX = 0.5 (for monotonic test run)
set EPSJU = # (value of strain for jump to take place)
set EPDJU = # (value of new strain rate at jump)

For STRESS RELAXATION TEST
set RELAX = 1 (every time relaxation test is run)
set JUMP = 1 (every time relaxation test is run)
set EPSJU = # (value of constant strain for test)
set EPDJU = 0.0 (strain rate is zero)
set TMAX = # (time to let specimen relax)

For CREEP TEST
set CREEP = 1
set CIC = # (initial value of strain for creep test)
set RC = # (value of constant stress)
set SMAX = # (value of maximum strain, where to stop test)

For NORMALIZED STRESS DATA w.r.t. INITIAL DRAG

71
set ND = 1

TO GET GREATER RESOLUTION OF CURVES, MAKE CINT SMALLER. HOWEVER
IT WILL TAKE LONGER TO REACH A SOLUTION

SPECIAL NOTE:
Regardless of program order, ACSL evaluates the DERIVATIVE section first and then the DYNAMIC section. Also, the DERIVATIVE section has approximately 10 times the iterations as the DYNAMIC section.

Variable Definitions

A1 - Coeff. for Isotropic hardening
A2 - Coeff. for Kinematic hardening
B1 - Coeff. for Isotropic hardening
B2 - Coeff. for Kinematic hardening
B - Kinematic hardening stress factor
BACK - Back stress, same as B (for Kinematic hardening)
BD - Kinematic Stress rate
BIC - Initial value of BD
C - Counter of strain rate reversals
CEPD - Strain rate during creep test (not %)
CIC - Initial strain for start of Creep test
CINT - Communication interval
CMAX - Maximum number of strain rate reversals
CO1 - Saturation strain rate constant
CREEP - Flag to run creep test
D - Drag stress
DD - Drag stress rate
DIC - Initial value of Drag stress D
DRAG - Drag stress (same as D)
E - Modulus of Elasticity
EPD - Epsilon Dot, strain rate (used for reversals)
EPDJU - EPD after the JUMP for jump test
EPS - Epsilon, strain
EPSD - Epsilon Dot, strain rate
EPSJU - Strain value (EPS) where JUMP occurs for jump test
EPSMAX - Maximum value of EPS (point where reversal occurs)
ISO - Type of Isotropic hardening;
1=time dependent
2=total strain dep.
3=plastic strain dep.
JUMP - Flag for jump test
KIN - Type of Kinematic hardening:
0=Haisler incorrect mod.
91 " 1=plastic strain dep. "
92 " N - Strain rate sensitivity factor "
93 " ND - Flag for normalization of stresses to initial drag "
94 " 0 -> no normalization, 1 -> normalization "
95 " NORM - Normalization factor "
96 " R - Stress "
97 " RC - Stress for creep test "
98 " RD - Stress rate "
99 " RIC - Initial value of stress R "
100 " RELAX - Flag to run relaxation "
101 " SMAX - Maximum strain for creep test (%) "
102 " STRAIN - Strain (same as EPS*100) (%) "
103 " STRESS - Stress (same as R) "
104 " T - Time "
105 " TI - Time increment (as seen by this subroutine) "
106 " TIME - Time "
107 " TMAX - Time to let specimen relax "
108 " TO - Time at which Relaxation began "
109 "-----------------------------------------------"
110 PROGRAM VISCO
111 ""
112 " Communication interval "
113 CINTERVAL CINT = 0.0001
114 ""
115 " Defining preset variables "
116 " Program initially begins with jump test off and no hardening. "
117 ""
118 INTEGER N,ND,ISO,KIN,JUMP,RELAX,CREEP
119 CONSTANT DIC=10.0E3, RIC=0.0, BIC=0.0 ... 
120 ,E=21.7E6, EPSD=1.0, ND=1 ... 
121 ,C01=1.0, N=20 ... 
122 ,A1=0.0, B1=0.0 ... 
123 ,EPSMAX=0.005 ... 
124 ,CMAX = 3.5, ISO=3 ... 
125 ,JUMP=0, EPSJU=0.0035, EPDJU=10.0 ... 
126 ,KIN=1, A2=0.0, B2=0.0 ... 
127 ,RELAX=0, TMAX=1.0E-2 ... 
128 ,CREEP=0, CIC=0.0035, RC=1.0E4, SMAX=0.8 
129 ""
130 INITIAL
131 EPS = 0.0 
132 EPD = EPSD 
133 TI = CINT 
134 C = 0.0 
135 IF (ND .EQ. 0) NORM = 1.0
136 IF (ND.EQ. 1) NORM = DIC
137 IF (RELAX .EQ. 0) TO = 1.0E20
138 IF (CREEP .EQ. 0) SMAX = 1000
139 IF (CREEP .EQ. 1) R = RC
140 IF (CREEP .EQ. 1) CMAX=1.0E10
141 END $ "OF INITIAL"
142 DYNAMIC
143 DERIVATIVE
144 "-------- VISCOPLASTIC STRAIN RATE ---------------"
145 RC1 = C01*((R-B)/D)**N
146 "-------- STRESS RATE EQUATION -------------------"
147 IF (CREEP .EQ. 0) RD = E*(EPD - (SIGN(RC1,R-B)))
148 "-------- STRAIN RATE EQUATION (CREEP TEST) -----
149 IF (CREEP .EQ. 1) CEPD = SIGN(RC1,R-B)
150 PROCEDURAL (DD=IS0,RC1,D)
151 "-------- WORK HARDENING (ISOTROPIC) -----------"
152 IF (ISO .EQ. 1) DD = B1 - A1*D
153 IF (ISO .EQ. 2) DD = (ABS(EPD))*B1 - A1*D
154 IF (ISO .EQ. 3) DD = (ABS(RC1))*B1 - A1*D
155 END $ "OF PROCEDURAL"
156 "-------- WORK HARDENING (KINEMATIC) ------------"
157 PROCEDURAL (BD=KIN,RC1,B)
158 IF (KIN .EQ. 0) BD = SIGN(RC1,R-B)*(B2 - A2*B)
159 IF (KIN .EQ. 1) BD = SIGN(RC1,R-B)*B2 - ABS(RC1)*A2*B
160 END $ "OF PROCEDURAL"
161 "-------- INTEGRATING FOR DRAG & BACK STRESS -------"
162 D = INTEG(DD,DIC)
163 DRAG = D/NORM
164 B = INTEG(BD,BIC)
165 BACK = B/NORM
166 "-------- INTEGRATING FOR STRESS ------------------"
167 IF (CREEP .EQ. 0) R = INTEG(RD,RIC)
168 STRESS = R/NORM
169 "-------- INTEGRATING FOR STRAIN (CREEP TEST) ----
170 IF (CREEP .EQ. 1) STRAIN = INTEG(CEPD,CIC)*100.0
171 END $ " OF DERIVATIVE"
172 ""
173 "-------- CALCULATE STRAIN AND REVERSING IT IF NEEDED "
174 PROCEDURAL (EPD,EPS,C=T,TI,EPSMAX)
175 CALL EPSILON(T,TI,EPS,EPD,EPSMAX,C)
176 IF (CREEP .EQ. 0) STRAIN = EPS*100.0
177 ""
178 "-------- PERFORMING JUMP TEST -------------------"
179 IF (JUMP .EQ. 1 .AND. EPS .GE. EPSJU) EPD = EPDJU
180 "-------- TURN FLAG OFF FOR JUMP TEST ----------"
IF (JUMP .EQ. 1 .AND. EPS .GE. EPSJU) JUMP = 0

"-------- STRESS RELAXATION ---------------------"

IF (RELAX .EQ. 1 .AND. EPS .GE. EPSJU) TO = T

IF (RELAX .EQ. 1 .AND. EPS .GE. EPSJU) RELAX = 0

END $ "OF PROCEDURAL"

TIME = T

""

"-------- SPECIFYING TERMINATION CONDITION ------"

TERMT(C .GE. CMAX .OR. (T-TO) .GE. TMAX .OR. STRAIN .GE. SMAX)

END $ " OF DYNAMIC"

END $ " OF PROGRAM"
* Subroutine: EPSILON
* Written by: TED DIEHL
* Revised : 3/20/88

* Purpose
5 * To evaluate the strain (EPS) as a function of time (T)
6 * and strain rate (EPD). When the maximum strain level is
7 * reached, the strain rate (EPD) is reversed so that cyclic
8 * loading and unloading can be accomplished. A counter is
9 * also included to determine how many cycles have been
10 * performed.
11 *
12 * Special note:
13 * Although this subroutine is written as part of an ACSL program,
14 * the subroutine follows the rules of FORTRAN and not ACSL.
15 * That is why the comments are delegated by the * and not the
16 * quotes (""') as in the ACSL language.
17 *
18 * Variable Definitions:
19 * C - Counter for number of strain rate reversals.
20 * EPD - Epsilon Dot, strain rate; units: 1/sec
21 * EPS - Epsilon, strain; units: dimensionless
22 * EPSMAX - Maximum value of EPS (point where reversal occurs);
23 * units: dimensionless
24 * T - Time; units: seconds
25 * TI - Time increment (as seen by this subroutine);
26 * units: seconds
27 *
28 *
29 *
30 SUBROUTINE EPSILON(T,TI,EPS,EPD,EPSMAX,C)
31 *
32 REAL T,TI,EPS,EPD,EPSMAX,C
33 *
34 PROGRAM
35 *
36 * Evaluate strain
37 IF (T.EQ. 0.0) THEN
38 EPS = 0.0 ! initialize strain
39 ELSE
40 EPS = EPD*TI + EPS
41 ENDIF
42 *
43 * Check to see if strain rate should be reversed
44 * Note: (-TI*ABS(EPD)/2.0) takes care of roundoff error from ACSL
45 IF (ABS(EPS) .GT. (EPSMAX-(TI*ABS(EPD)/2.0))) THEN
46 \[ C = C + 0.5 \] !counter of loops
47 \[ EPD = -EPD \] !reversing strain rate
48 ENDIF
49 RETURN  !return to mainprogram VISCO
50 END
Appendix B

DERIVATIONS OF FINITE ELEMENT EQUATIONS

This appendix contains the derivations of the finite element equations used in this work. The basis for the derivations comes from several sources, namely [5,16,31,32,33]. Appropriate modifications are made to the standard finite element derivations in order to incorporate the elastic-viscoplastic constitutive equations. The derivations are general with respect to the major steps in their development but several of the equations, particularly variable definitions, are specific to the axisymmetric case and a four noded bilinear isoparametric element.

The following equations are element equations; they are valid for a particular element. Global equations for the entire system are created using the standard methods listed in the references.

Starting with the principle of the rate of virtual work applied to an element and assuming isothermal quasi-static loading with negligible body forces, we have

\[
\int_\Omega [\delta \varepsilon]^T [\dot{\sigma}] \, d\Omega = \int_S [\delta U]^T [\dot{F}] \, dS
\]

Rate of Virtual Strain Energy \hspace{1cm} Rate of Virtual External Work

where \([\delta \varepsilon]^T\) is the transpose of the variation of the strain, \([\dot{\sigma}]\) is the stress rate, \(\Omega\) is the domain of the element, \([\delta U]^T\) is the transpose of the variation of the general displacements, \([\dot{F}]\) is the rate of external forces, and \(S\) is the surface. All of these quantities are element quantities.

For the axisymmetric case, the strain and stress-rate are defined as:

\[
[\varepsilon] = \begin{bmatrix} \varepsilon_z \\ \varepsilon_r \\ \varepsilon_\theta \\ \gamma_{rz} \end{bmatrix} = \begin{bmatrix} \frac{\partial u}{\partial z} \\ \frac{\partial u}{\partial r} \\ \frac{\partial v}{\partial r} \\ \frac{\partial u}{\partial z} + \frac{\partial v}{\partial r} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \frac{\partial}{\partial r} & 0 \\ \frac{1}{r} & 0 \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = [L] [U]
\]

(B.2)
\[
\begin{bmatrix}
\dot{\sigma}_z \\
\dot{\sigma}_r \\
\dot{\sigma}_\theta \\
\dot{\tau}_{rz}
\end{bmatrix}
\]  
(B.3)

The matrix \([L]\) in equation B.2 is called the linear differential operator. The general displacements within an element are denoted by \([U]\) where \(u\) is the radial component and \(v\) is the axial component. The four stress-rate components denoted are axial, radial, hoop, and shear.

Within an element, the stress is related to the elastic strain via

\[
[\sigma] = [E] [\varepsilon]
\]  
(B.4)

The elastic stiffness\(^1\) for an isotropic material is defined by the matrix \([E]\).

\[
[E] = \begin{bmatrix}
(2G + \lambda) & \lambda & \lambda & 0 \\
\lambda & (2G + \lambda) & \lambda & 0 \\
\lambda & \lambda & (2G + \lambda) & 0 \\
0 & 0 & 0 & G
\end{bmatrix}
\]

where:

\[
G = \frac{E}{2(1 + \nu)} \quad \lambda = \frac{\nu E}{(1 + \nu)(1 - 2\nu)}
\]

The variables \(E\) and \(\nu\) denote the modulus of elasticity and Poisson’s ratio. The total strain is related to elastic and viscoplastic strain by the relationship

\[
[\varepsilon] = [\varepsilon^e] + [\varepsilon^{vp}]
\]  
(B.5)

Combining equations B.4 and B.5 and taking a derivative with respect to time yields

\[
[\dot{\sigma}] = [E] ([\dot{\varepsilon}] - [\dot{\varepsilon}^{vp}])
\]  
(B.6)

where \([E]\) is considered constant and \([\dot{\varepsilon}^{vp}]\) is defined in Chapter 3.

Within each element, we assume the displacement field (general displacements) is related by the shape functions to the nodal displacements in each element.\(^2\)

\[
U = \sum_{i=1}^{N} \psi_i \bar{U}_i
\]  
(B.7)

The total number of nodes in an element is represented by \(N\) and \(\bar{U}_i\) is the nodal displacement of the element at node \(i\). The shape functions are represented by \(\psi_i\). For the isoparametric element, this same equation also defines the geometry of the element.

Substituting equation B.7 into equation B.2 yields the matrix equation

\(^1\)The elastic stiffness is often denoted by the matrix \([D]\). It is denoted as \([E]\) in this study in order to avoid confusion with the drag stress, \(D\), defined previously.

\(^2\)The tilde in all the following equations denotes a nodal quantity.
\[ \epsilon = [L] \psi [\bar{U}] \]  

(B.8)

For a four noded element, the shape function matrix and nodal displacement vector are:

\[ [\psi] = \begin{bmatrix} \psi_1 & 0 & \psi_2 & 0 & \psi_3 & 0 & \psi_4 & 0 \\ 0 & \psi_1 & 0 & \psi_2 & 0 & \psi_3 & 0 & \psi_4 \end{bmatrix} \]

and

\[ [\bar{U}] = \begin{bmatrix} \tilde{u}_1 \\ \tilde{v}_1 \\ \tilde{u}_2 \\ \tilde{v}_2 \\ \tilde{u}_3 \\ \tilde{v}_3 \\ \tilde{u}_4 \\ \tilde{v}_4 \end{bmatrix} \]

For the isoparametric bilinear quadrilateral, the four shape functions, \( \psi_i \), are:

\[ \psi_1 = \frac{1}{4} (1 - \xi)(1 - \eta) \]

\[ \psi_2 = \frac{1}{4} (1 + \xi)(1 - \eta) \]

\[ \psi_3 = \frac{1}{4} (1 + \xi)(1 + \eta) \]

\[ \psi_4 = \frac{1}{4} (1 - \xi)(1 + \eta) \]

The shape functions map the element from the physical coordinates of the physical domain to the natural coordinates of the parametric domain (a biunit square). The coordinates \( \xi \) and \( \eta \) are called the natural coordinates. An example of this mapping for a quadrilateral element along with an example of the bilinear shape function is seen in figure B.1. The coordinates \( x \) and \( y \) in the figure denote the physical coordinates of the element.

Since the differential operator, \([L]\), premultiplies the shape functions, \([\psi]\), the partial derivatives of the shape functions are needed. For the isoparametric bilinear quadrilateral, the derivatives are defined by

\[ \frac{\partial \xi}{\partial \eta} \frac{\partial \xi}{\partial r} \frac{\partial \xi}{\partial z} \] = \frac{1}{J} \begin{bmatrix} \frac{\partial z}{\partial \eta} & -\frac{\partial \eta}{\partial \eta} \\ -\frac{\partial z}{\partial \xi} & \frac{\partial \xi}{\partial \xi} \end{bmatrix} \]

where \( J \) is the determinant of the Jacobian.
Figure B.1: Isoparametric Mapping and Shape Function For Bilinear Quadrilateral Element
\[
    j = \frac{\partial r}{\partial \xi} \frac{\partial z}{\partial \eta} - \frac{\partial r}{\partial \eta} \frac{\partial z}{\partial \xi}
\]

To reduce notation, the linear differential operator and the shape functions are combined and defined as

\[
    [C] = [L] [\psi] \tag{B.9}
\]

The resulting matrix \([C]\) relates the strains in the element to the nodal displacements and is called the strain-displacement matrix. This matrix is often denoted as \([B]\) in the literature but is denoted by \([C]\) here to avoid confusion with the back stress, \(B\), defined previously.

Substitution of equation B.9 into equations B.8 and B.6 provides a new form of the constitutive equation.

\[
    [\dot{\sigma}] = [E] \left( [C] \left[ \dot{U} \right] - [\dot{\varepsilon}^{vp}] \right) \tag{B.10}
\]

Combining equations B.1, B.7, B.8, B.9, and B.10 and further simplifying eventually leads to the final form of the element equation.

\[
    [K] \left[ \dot{U} \right] = [\dot{F}^{vp}] + [\dot{F}^{ext}] \tag{B.11}
\]

where:

\[
    [K] = \int_{\Omega} [C]^T [E] [C] \, d\Omega
\]

\[
    [\dot{F}^{vp}] = \int_{\Omega} [C]^T [E] [\dot{\varepsilon}^{vp}] \, d\Omega
\]

\[
    [\dot{F}^{ext}] = \int_{S} [\psi]^T [\dot{F}] \, dS
\]

For the special axisymmetric case investigated, the element domain is part of a cylinder. Hence, the element domain is defined as

\[
    d\Omega = 2\pi r \, dr \, dz
\]

Derivation of \(dS\) is a bit more tedious and is not derived here since all studies used displacement control.

Global equations are assembled from these element equations using the standard methods found in the references. Once the global system is created, a solution is calculated using the iterative strategy discussed in Chapter 3.

Integration for these equations is performed in space (over the domain of the element) and in time. Since the element could be nonrectangular, the spatial integration over each element is performed in the parametric domain. For the two-dimensional case, the integration formula for an element is
\[ \int_{\Omega} f(x, y) \, d\Omega = \int_{-1}^{1} \int_{-1}^{1} f(x(\xi, \eta), y(\xi, \eta)) j(\xi, \eta) \, d\xi \, d\eta \]  
(B.12)

where \( f \) is a function and \( j \) is the Jacobian determinant. This exact integral is approximated using Gaussian quadrature.

\[ \int_{\Omega} f(x, y) \, d\Omega \approx \sum_{i=1}^{M} f(x(\xi_{i}, \eta_{i}), y(\xi_{i}, \eta_{i})) j(\xi_{i}, \eta_{i}) W_{i} \]  
(B.13)

Here, the exact integral is approximated by evaluating the integrand at special Gauss points and summing the resultant evaluations according to a particular weighting scheme. The values of \( x(\xi_{i}, \eta_{i}) \) and \( y(\xi_{i}, \eta_{i}) \) are computed using the shape functions. The total number of Gauss points is \( M \), the location of each Gauss point is the coordinates \( (\xi_{i}, \eta_{i}) \), and the weighting factors are \( W_{i} \) for each Gauss point \( l \). The Gauss point locations and weighting factors for several quadrature schemes are found in [32].

In order to integrate in time, the exact time derivatives are approximated using any one of a number of numerical integration schemes. For this study, the numerical method used is the rectangular rule [5] which states

\[ \int_{t_{o}}^{t_{o}+\Delta t} f(t) \approx \Delta t f \left( t_{o} + \frac{\Delta t}{2} \right) \]  
(B.14)

Here, \( f \) is the function or variable to be integrated. Using this integration method, incremental solutions in time are created. Applying this technique to the element equation (equation B.11) provides the incremental form of the finite element method.

\[ \int_{\Omega} [C]^{T} [E] [C] \, d\Omega \, [\Delta \bar{U}] = \int_{\Omega} [C]^{T} [E] [\Delta \epsilon^{vp}] \, d\Omega + \int_{S} [\psi]^{T} [\Delta F] \, dS \]  
(B.15)

where \([\Delta \epsilon^{vp}]\) is the incremental form of equation 3.11

\[ \Delta \epsilon_{ij}^{vp} = \Delta t \frac{3}{2} \dot{\epsilon}_{o} \left( \left( \frac{\sigma_{eff}}{D} \right)^{n} \frac{S_{ij}}{\sigma_{eff}} \right) \Delta t/2 \]

The subscript \( \Delta t/2 \) represents the fact that the quantity enclosed within the braces is evaluated at the midpoint of the time step. The total value of displacement, \( [\bar{U}] \), is found by adding the incremental value, \( [\Delta \bar{U}] \), to the previous total value.
Appendix C

AXISYMMETRIC FINITE ELEMENT PROGRAM, FEPROG

The finite element program, FEPROG, is written in FORTRAN and solves the axisymmetric problem. The program was originally written by Ghoneim [5] and has been updated and revised to include (among other things) the capability of simulating isotropic work hardening. The program currently uses four node isoparametric elements and can be modified for higher order elements. Incremental solutions are calculated at each time step and the total solution is obtained by adding the incremental value to the previous total value.

The program consists of several modules. A general flow chart for the program is displayed in figure C.1. The basic function of each module is as follows:

FEPROG Main program; controls all subroutines

QUEST Subroutine; prompts user for input and output file names and for types of stress output required

GDATA Subroutine; reads in material, nodal, and element data and produces, if requested, linear or quadratic mesh pattern

FORMK Subroutine; creates global stiffness

STIFT1 Subroutine; creates element stiffness

SHAPEF Subroutine; performs mapping and transformations of shape functions

PGAUSS Subroutine; determines the Gauss points and weighting factors for Gaussian quadrature

BC Subroutine; reads in boundary conditions

BNDCND Subroutine; places boundary conditions into the matrix formulation
Figure C.1: Flow Chart For Finite Element Program
LOADS Subroutine; calculates the viscoplastic loads

SOLVE Subroutine; solves global equations for incremental displacements using a special Gauss elimination technique

OUTPUT Subroutine; calculates strains and stresses based on the incremental displacements and checks for convergence

PLOT Subroutine; calculates nodal stress values based on simple averaging of Gauss point stresses

A FORTRAN listing of each of these modules appears in the following pages. Some of the variables in the modules are different than the variables used in the equations of Chapters 1–4 (i.e., strain-rate sensitivity, \( n \), is \( M_\phi \) in the program). Also listed at the end is a sample input file. The sample input file does not contain numbers, but rather the variables and logic that the program reads and uses during execution. The variables \( N \), \( I \), and \( K \) are counters.
Program FEPROG

0001  * Elastic-Viscoplastic constitutive equation.
0002  * Using Power Law Formulation
0003  *
0004  IMPLICIT REAL*8(A-H,O-Z)
0005  INCLUDE 'COMMON,FOR/LIST'
0006  * MAX - Maximum value for variable
0007  INTEGER JTMAX,MATMAX,NEMAX,NNMAX,NBMAX,NLDMAX
0008  INTEGER EN,TEX,BCFLAG,KFLAG,ETMAX
0009  PARAMETER (JTMAX=20,MATMAX=5,NEMAX=200,NNMAX=400,NBMAX=100)
0010  PARAMETER (NLDMAX=600,ETMAX=7)
0011  INTEGER TIMES(NLDMAX)
0012  INTEGER TYPE(ETMAX),GP(ETMAX),ENUM(ETMAX),ETOTAL
0013  REAL*8 LENGTH,DELTAL,NORM(MATMAX)
0014  REAL*8 DI(2,NNMAX),DELTAT(NLDMAX),DELTAU(NLDMAX)
0015  REAL*8 NSTRESS(NEMAX),SKS(2*NNMAX,NBMAX)
0016  COMMON TYPE,GP,ENUM,ETOTAL
0017  COMMON NP,NE,NB,NLD,NDF,NMAT,NSZF,NBAND,KFLAG
0018  COMMON IL,IT,IP,NDF,COUNT(NNMAX),IS,TEX,BCFLAG
0019  COMMON PIE,ERC,ERRELC,RRT,DT,JTERM,LG
0020  COMMON ORI(MATMAX,10),CORD(NNMAX,2),NOP(NEMAX,8)
0021  COMMON IMAT(NEMAX),ISNE(NEMAX),MPB(MATMAX)
0022  COMMON NBC(NNMAX),NFX(NNMAX),UBC(NNMAX,2)
0023  COMMON SKS(2*NNMAX,NBMAX),SK(2*NNMAX,NBMAX)
0024  COMMON RI(2*NNMAX),RL(2*NNMAX),UL(2*NNMAX)
0025  COMMON LENGTH,DELTAL,STRESS,STRAIN,NORM
0026  COMMON YO(NEMAX,9),YY1(NEMAX,9),BKO(NEMAX,9),BK1(NEMAX,9)
0027  COMMON SIG(NEMAX,9,4),SIGO(NEMAX,9,4)
0028  COMMON EVF(NEMAX,9),SEF(NEMAX,9),SEFM(NEMAX,9)
0029  COMMON EPSI(NEMAX,9,4),EPSO(NEMAX,9,4)
0030  COMMON DSV(NEMAX,9,4),ESTIFM(16,16)
0031  COMMON ERG(JTMAX),ERRELJ(JTMAX)
0032  * * * * * * * * * * ..........Prerequisite data
0033  * BKO(EN,I) - Effective viscoplastic stress in element EN
0034  * at Gauss point I at end of load step
0035  * BK1(EN,I) - Effective viscoplastic stress in element EN
0036  * at Gauss point I at current time
0037  * DELTAL - Total change in length of cylinder for uniform
0038  * compression or tension
0039  * DSV(EN,I,K) - Incremental viscous stress in element EN at
0040  * Gauss point I
0041  * EN - Element number
0042  * EPSO(EN,I,K) - Total strain in element EN at Gauss point I at
0043  * end of load step
0044  * EPSI(EN,I,K) - Total strain in element EN at Gauss point I
0045  * EVF(EN,I) - Effective strain rate
0046  * I - Counter
0047  * IL - Counter for load steps
0048  * IQ - # of load steps from last print (when IQ=IP -> print)
0049  * IT - Counter for iterations during a load step
0050  * K - K=1 -> Z, K=2 -> R, K=3 -> THETA, K=4 -> RZ
0051  * KFLAG - Indicates if [K] matrix needs to be calculated
0052  * every load step. KFLAG=1 -> only once
0053  * MP - Power index for viscoplastic strain rate power formula
0054  * NE - Total number of elements
0055  * NLD - Total number of load steps
0056  * SEF(EN,I) - Effective stress for element EN at Gauss point I

87
0057 * SEFM(EN,I) - Same as SEF(EN,I) but at midpoint in time (used
0058 * for time integration)
0059 * SIG(EN,I,K) - Stress for element EN at Gauss point I
0060 * SIGO(EN,I,K) - Stress for element EN at Gauss point I at end of
0061 * load step
0062 * YYO(EN,I) - Drag stress in element EN at Gauss point I
0063 * at end of load step
0064 * YYI(EN,I) - Drag stress in element EN at Gauss point I
0065 * at current time step
0066 *
0067 *
0068 *........Input and output file designations
0069 * for005 - Input from screen
0070 * for006 - Screen output for error messages
0071 * UNIT 15 - data input *.FE
0072 * UNIT 16 - complete output of program (used mainly for debug)
0073 * FE.OUT
0074 * UNIT 17 - output file w/connectivity data for PATPLOT
0075 * * .con where * is same name as input file name
0076 * UNIT 26 - Engineering stress and strain at every load step
0077 * *.ENG
0078 * UNIT 36 - Nodal stress contours at the frequency of data output
0079 * *.CNT where *=3 letter name and #=load step number
0080 * UNIT 47+ - Element quantities that can be followed for an
0081 * individual element for all the load increments.
0082 * T4GP%.S where %=stress type (0-6), %=gauss point
0083 * to follow, and %=element number.
0084 *
0085 * CALL QUEST(l) ! get name of input file
0086 * CALL GDATA ! gets material data
0087 * CALL QUEST(2) ! normalize data?, and element quantities
0088 * to be followed.
0089 *
0090 *........Initialization (zero matrices)
0091 DO EN=1,NE
0092 DO I=1,9
0093 YYO(EN,I) = ORT(IMAT(EN),5)
0094 YYI(EN,I) = YYO(EN,I)
0095 BKO(EN,I) = 0.0
0096 BK1(EN,I) = 0.0
0097 EVF(EN,I) = 0.0
0098 SIG(EN,I,K) = 0.0
0099 SIGO(EN,I,K) = 0.0
0100 SEFM(EN,I) = 0.0
0101 DO K=1,4
0102 EPST(EN,I,K) = 0.0
0103 SIG(EN,I,K) = 0.0
0104 DS(EN,I,K) = 0.0
0105 SIGO(EN,I,K) = 0.0
0106 EPSO(EN,I,K) = 0.0
0107 END DO
0108 END DO
0109 END
0110 END DO
0111 IQ = 0
0112 DELTAL = 0.0
0113 * * * Classical procedures
0114 IF(KFLAG .EQ. 1) CALL FORMK
0115 DO IL=1, NLD ! for each load step
0116 IT = 1
0117
0117  IQ = IQ + 1
0118  *
0119  IF(IQ .EQ. IP .OR. IL .EQ. 1) CALL QUEST(3)
0120  IF(KFLAG .NE. 1) CALL FORMK
0121  CALL BC
0122  DO WHILE(IT .NE. 0) ! until convergence
0123      CALL LOADS
0124      CALL SOLVE
0125      CALL OUTPUT
0126  *
0127      IF(convergence or termination criterior is met, IT -> -1
0128      IT = IT + 1
0129  *
0130      IF(IQ .EQ. 0 .OR. IL .EQ. 1) CALL QUEST(4)
0131  END DO
0132  *
0133  close contour file
0134  *
0135  IF(IQ .EQ. 0 .OR. IL .EQ. 1) CALL QUEST(4)
0136  CALL QUEST(5) ! close all remaining files
0137  *
0138  STOP
0139  END
Subroutine QUEST

0001  *
0002  *
0003  *
0004  *
SUBROUTINE QUEST(FLAG)
0005  *
0006  * This routine asks for file names and other input
0007  *
0008  *
IMPLICIT REAL*8(A-H,O-Z)
0009  CHARACTER*1 C1,C2
0010  CHARACTER*3 NODAL,C3
0011  CHARACTER*4 CONVERT
0012  CHARACTER*7 INPUT,ENGINEER
0013  INTEGER FLAG,FLAG1
0014  INCLUDE 'COMMON.FOR/LIST'
0015  *
0016  * MAX - Maximum value for variable
0017  INTEGER JTMAX,MATMAX,NEMAX,NNMAX,NBMAX,NLDMAX
0018  INTEGER EN,TEX,BCFLAG,KFLAG,ETOTAL
0019  INTEGER PARAMETER (JMAX=20,MATMAX=5,NEMAX=200,NNMAX=400,NBMAX=100)
0020  INTEGER PARAMETER (NLDMAX=600,ETMAX=7)
0021  INTEGER TIMES(NLDMAX)
0022  INTEGER ETMAX,GP(ETMAX),ENUM(ETMAX),ETOTAL
0023  REAL*8 LENGTH,DELTAL,NORM(MATMAX)
0024  REAL*8 DIS(2,NNMAX),DELTAT(NLDMAX),DELTAL,STRESS(NNMAX)
0025  REAL*8 Nlag(EN),SIG(2*NEMAX,9),SIGO(2*NEMAX,9)
0026  REAL*8 REAL*8 LENGTH,DELTAL,NORM(MATMAX)
0027  REAL*8 LENGTH,DELTAL,NORM(MATMAX)
0028  REAL*8 LENGTH,DELTAL,NORM(MATMAX)
0029  COMMON TYPE,GP,ENUM,ETOTAL
0030  COMMON NP,NE,NB,NLD,NDF,NMAT,NSZF,NBAND,KFLAG
0031  COMMON IL,IT,IP,IQ,COUNT(NNMAX),IS,TEX,BCFLAG
0032  COMMON IL,IT,IP,IQ,COUNT(NNMAX),IS,TEX,BCFLAG
0033  COMMON ORT(MATMAX,10),CORD(NNMAX,2),NOP(NEMAX,8)
0034  COMMON IMAT(NEMAX),ISNE(NEMAX),MPB(MATMAX)
0035  COMMON NBC(NNMAX),NFix(NNMAX),UBC(NNMAX,2)
0036  COMMON SK(2*NNMAX,NBMAX),SKS(2*NNMAX,NBMAX)
0037  COMMON Nlag(EN),SIG(2*NEMAX,9),SIGO(2*NEMAX,9)
0038  COMMON Nlag(EN),SIG(2*NEMAX,9),SIGO(2*NEMAX,9)
0039  COMMON SK(2*NNMAX,NBMAX),SKS(2*NNMAX,NBMAX)
0040  COMMON ORT(MATMAX,10),CORD(NNMAX,2),NOP(NEMAX,8)
0041  *
0042  * Variables used
0043  * CONVERT - converts load step to character
0044  * ENGINEER - File name for output engineering stress
0045  * ENUM(I) - Element number to be followed
0046  * ETMAX - Maximum number of element quantities that
0047  * can be followed
0048  * ETOTAL - Number of element quantities to follow
0049  * FLAG - control flag
0050  * FLAG1 - control flag
0051  * GP(1) - Gauss point to follow
0052  * I - Counter of element quantities
0053  * IL - Load step
0054  * INPUT - File name of input file and output
0055  * connectivity file
0056  * LG - Number of Gauss integration points in one dir

90
* N - Counter for material types
* NLD - Total number of load steps
* NMAT - Number of material types
* NODAL - File name of nodal stress contours
* NORM(N) - Normalization factor (Initial Drag value)
* ORT(N,5) - Initial drag stress
* TYPE(I) - Type of stress to be followed

GOTO(1,2,3,4,5) FLAG

WRITE(6,*) 'ENTER NAME IN QUOTES (7 LETTERS MAX) OF ',
READ(5,*), 'INPUT FILE. EXTENSION IS .FE BY DEFAULT'

WRITE(6,*) 'ENTER NAME IN QUOTES (7 LETTERS MAX) OF ',
READ(5,*), 'EXTENSION IS .ENG BY DEFAULT'

WRITE(6,*) 'ENTER NAME IN QUOTES (3 LETTERS MAX) OF ',
READ(5,*), 'OUTPUT FILE FOR NODAL STRESSES. THE LOAD NUMBER WILL BE ADDED AND THE ',
WRITE(6,*), 'EXTENSION IS .CNT BY DEFAULT'

READ(5,*), NODAL

OPEN (UNIT-15, FILE=INPUT//'.FE',STATUS='UNKNOWN')
OPEN (UNIT-16, FILE=FE//'.OUT',STATUS='UNKNOWN')
OPEN (UNIT-17, FILE=INPUT//'.CON',STATUS='UNKNOWN')
OPEN (UNIT-26, FILE=ENGINEER//'.ENG',STATUS='UNKNOWN')
RETURN

*........normalizing data and choosing element quantities to follow

WRITE(6,600)
READ(5,*), FLAG1
DO N=1,NMAT
   IF(FLAG1.EQ.1) THEN
      NORM(N) = ORT(N,5)
   ELSE
      NORM(N) = 1.0
   ENDIF
END DO
WRITE(6,610)
WRITE(6,*), 'MAX IS ',ETMAX
READ(5,*), ETOTAL
IF(ETOTAL.GT.ETMAX) THEN
   WRITE(6,*) 'TOO MANY, MAX IS ',EMAX
STOP
ENDIF
DO I=1,ETOTAL
   WRITE(6,*) 'ENTER ELEMENT #, GAUSS POINT, AND TYPE'
   WRITE(6,*), 'TYPE = 0 -> EFFECTIVE STRESS, SEF'
   WRITE(6,*), 'TYPE = 1 -> Z STRESS, SIG(Z)'
   WRITE(6,*), 'TYPE = 2 -> R STRESS, SIG(R)'
   WRITE(6,*), 'TYPE = 3 -> THETA STRESS, SIG(THETA)'
   WRITE(6,*), 'TYPE = 4 -> RZ STRESS, SIG(RZ)'
   WRITE(6,*), 'TYPE = 5 -> DRAG STRESS, YY1'
   WRITE(6,*), 'TYPE = 6 -> EFFECTIVE VISCOPLASTIC STRAIN, B
   READ(5,*), ENUM(I),GP(I),TYPE(I)
   IF(ENUM(I).GT.NE) THEN
      WRITE(6,*) 'ERROR, ELEMENT # > MAXIMUM # OF ELEMENTS'
      STOP
   ENDIF
END DO

91
ENDIF
0118 IF(GP(I).GT.LG*LG) THEN
    WRITE(6,*) 'ERROR, GAUSS > MAXIMUM # OF GAUSS POINTS'
    STOP
ENDIF
0122 IF(TYPE(I).GT.6.OR.TYPE(I).LT.0) THEN
    WRITE(6,*) 'ERROR, TYPE > 6 OR < 0'
    STOP
ENDIF
0126 ENCODE(1,110,C1)
0127 ENCODE(1,120,C2)
0128 ENCODE(3,130,C3)
0129 OPEN (UNIT=46+I,FILE='T'/C1//'GP'//C2//'.'//C3,
     +STATUS='UNKNOWN')
0132 END DO
0134 WRITE(16,*) 'ELASTIC-VISCOPLASTIC FE SOLUTION'
0135 WRITE(26,*) 'USING POWER LAW FORMULATION'
0137 DO I=1,ETOTAL
    WRITE(46+I,*) 'ELEMENT DATA'
    END DO
0140 WRITE(26,*), NLD+1, ' = TOTAL NUMBER OF LOADS + 1'
0142 DO I=1,ETOTAL
    WRITE(46+I,*) NLD+1, ' = TOTAL NUMBER OF LOADS + 1'
    IF(TYPE(I).LT.5.OR.TYPE(I).EQ.6) THEN
        WRITE(46+I,*) 0.0,0.0, OUT ! all stress assumed zero
    ELSE
        OUT = ORT(IMAT(ENUM(I)),5)/NORM(IMAT(ENUM(I)))
        WRITE(46+I,*) 0.0,OUT ! initial drag
    END IF
END DO
0154 3 ENCODE (4,150,CONVERT) IL
0155 OPEN (UNIT=36,FILE=NODAL//CONVERT//'.CNT',STATUS='UNKNOWN')
0157 END
0158 4 CLOSE (UNIT=36)
0162 END
0165 CLOSE(UNIT=15)
0166 CLOSE(UNIT=16)
0167 DO I=1,ETOTAL
    CLOSE(UNIT=46+I)
END DO
0171 110 FORMAT(I1.1)
0172 120 FORMAT(I1.1)
0173 130 FORMAT(I3.3)
0174 150 FORMAT(I4.4)
0175 600 FORMAT(’0’, ’WOULD YOU LIKE ALL STRESSES NORMALIZED BY INITIAL ’,
'DRAG STRESS ?',/,' (YES=1, NO=2)')
FORMAT('0','HOW MANY DIFFERENT ELEMENTS/QUANTITIES ',
'WOULD YOU LIKE TO FOLLOW?')
END
Subroutine GDATA

This section reads in material data, nodal locations and element mesh information.

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

`REAL*8 MX,MY,IX,IY`

`INTEGER EI,GN(NN),TEY,XINC,YINC,XTEM(8),YTEM(8),FLAG`

* MAX - Maximum value for variable

`INTEGER JTMAX,MATMAX,NEMAX,NNMAX,NBMAX,NLDMAK`

`INTEGER EN,TEX,BNFLAG,ETYPE,ETMAX`

`PARAMETER (JTMAX=20,MATMAX=5,NEMAX=200,NNMAX=400,NBMAX=100)`

`PARAMETER (NLDMAK=600,ETMAX=7)`

`INTEGER TIMES(NLDMAK)`

`INTEGER TYPE(ETMAX),GP(ETMAX),ENUM(ETMAX),ETOTAL`

`REAL*8 LENGTH,DELTAL,NORM(MATMAX)`

`REAL*8 DIS(2,NNMAX),DELTAT(NLDMAK),DELTAU(NLDMAK)`

`REAL*8 NSTRESS(NNNMAX),SRT(2*NNMAX,NBMAX)`

`COMMON TYPE,GP,ENUM,ETOTAL`

`COMMON NP,NE,NB,NLD,DFK,NSZ,FNBAND,KFLAG`

`COMMON IL,IT,IP,ICOUNT(NNMAX),IS,TEX,BNFLAG`

`COMMON PIE,ERC,ERRELC,RRT,DT,JTERM,LAG`...
NOTE: For this program, the X direction is same as the R direction and the Y direction is the same as the Z direction

Material data

- **ORT(1)** = EE - Elastic modulus
- **ORT(2)** = PS - Poisson ratio
- **ORT(5)** = YS - The initial static drag stress
- **ORT(6)** = Al - Rise time constant for drag stress
- **ORT(7)** = Bl - Constant for drag stress rate equation
ORT(10) = GM  - Strain rate saturation constant

* .......Reading in initial parameters
READ(15,*) IP, IS, LG, NBAND, JTERM, ERC, ERRELC
PI = 3.1415926

* .......Control variables
READ(15,*) NP, NE, NB, KFLAG, BCFLAG, NLD, NDF, NMAT, II

IF(NP .GT. NNMAX) THEN
  WRITE(6,*), 'INPUT ERROR, NP> MAX ALLOWED'
  WRITE(6,*), 'CHECK COMMON BLOCK FOR MAX ALLOWED'
ENDIF

IF(NE .GT. NEMAX) THEN
  WRITE(6,*), 'INPUT ERROR, NE> MAX ALLOWED'
  WRITE(6,*), 'CHECK COMMON BLOCK FOR MAX ALLOWED'
ENDIF

IF(NB .GT. NBMAX) THEN
  WRITE(6,*), 'INPUT ERROR, NB> MAX ALLOWED'
  WRITE(6,*), 'CHECK COMMON BLOCK FOR MAX ALLOWED'
ENDIF

IF(JTERM .GT. JTMAX) THEN
  WRITE(6,*), 'INPUT ERROR, JTERM> MAX ALLOWED'
  WRITE(6,*), 'CHECK COMMON BLOCK FOR MAX ALLOWED'
ENDIF

IF(NLD .GT. NLDMAX) THEN
  WRITE(6,*), 'INPUT ERROR, NLD> MAX ALLOWED'
  WRITE(6,*), 'CHECK COMMON BLOCK FOR MAX ALLOWED'
ENDIF

IF(NMAT .GT. MATMAX) THEN
  WRITE(6,*), 'INPUT ERROR, NMAT> MAX ALLOWED'
  WRITE(6,*), 'CHECK COMMON BLOCK FOR MAX ALLOWED'
ENDIF

NSZF = NP*NDF  !total DOF (this defines length of [K])

* .........Material data
READ(15,*) (N, (ORT(N, I), I=1,5), N-1, NMAT)
READ(15,*) (N, (ORT(N, I), I=6,10), MPB(N), N-1, NMAT)
DO N=1, NMAT
  IF(ORT(N,5) .LE. 0.0) THEN
    WRITE(6,*), 'INITIAL DRAG .LE. 0, THIS IS INVALID'
    WRITE(16,*), 'INITIAL DRAG .LE. 0, THIS IS INVALID'
    STOP
  ENDIF
ENDDO

* .........Read Coordinate points of the nodes (in global X,Y)
READ(15,*) RRT, LENGTH  !radius & length of cylinder
IF(RRT .LE. 0.0) THEN
  WRITE(6,*), 'RADIUS RRT < or = 0, THIS IS INVALID'
  WRITE(16,*), 'RADIUS RRT < or = 0, THIS IS INVALID'
  STOP
ENDIF

IF(LENGTH .LE. 0.0) THEN
  WRITE(6,*), 'LENGTH < or = 0, THIS IS INVALID'
  WRITE(16,*), 'LENGTH < or = 0, THIS IS INVALID'
  STOP
ENDIF

* The following section has four options!
READ(15,*) FLAG
DO WHILE (FLAG .NE. -1)  !FLAG=-1 -> end of nodal input
  READ(15,*) N1, X1, Y1, X2, Y2, NXT, NYT
ENDDO

96
IF(NXT .EQ. 1 .OR. NYT .EQ. 1) THEN
WRITE(6,*) 'INPUT ERROR, FLAG=1 & NXT OR NYT =1'
WRITE(6,*) 'THIS IS A CONFLICT'
WRITE(16,*) 'INPUT ERROR, FLAG=1 & NXT OR NYT =1'
WRITE(16,*) 'THIS IS A CONFLICT'
STOP
ENDIF

DX = (X2 - X1)/(NXT - 1.0)
DY = (Y2 - Y1)/(NYT - 1.0)
NN = N1 node number
YTMP = -DY + Y1
DO I=1,NYT !total # of nodes in the y direction
  YTMP = YTMP + DY
  XTMP = -DX + X1
  DO J=1,NXT !total # of nodes in the x direction
    XTMP = XTMP + DX
    CORD(NN,1) = XTMP ! R coordinate
    CORD(NN,2) = YTMP ! Z coordinate
    NN = NN + 1
  END DO
END DO

*.......Rectangular nodal input (quadratic)
ELSEIF(FLAG .EQ. 2) THEN
READ(15,*) N1,X1,Y1,X2,Y2,NXT,NYT
IF(NXT .EQ. 1 .OR. NYT .EQ. 1) THEN
WRITE(6,*) 'INPUT ERROR, FLAG=2 & NXT OR NYT =1'
WRITE(6,*) 'THIS IS A CONFLICT'
WRITE(16,*) 'INPUT ERROR, FLAG=2 & NXT OR NYT =1'
WRITE(16,*) 'THIS IS A CONFLICT'
STOP
ENDIF

CHANGEX = MOD(NXT,2) + 1.0 ! determine odd or even # of
CHANGEY = MOD(NYT,2) + 1.0 ! determine odd or even # of
IX = (X2 - X1)/(NXT - 1.0)
IY = (Y2 - Y1)/(NYT - 1.0)
RX = (X2 - X1)/2.0
RY = (Y2 - Y1)/2.0
NN = N1 node number
MY = 1.0
CY = 0.0
DY = 0.0
DO I=1,NYT !total # of nodes in the y direction
  IF(DY .GT. RY) THEN
    MY = -1.0
    CY = 2.0*RY
    DY = DY - CHANGEY*IY
  ENDIF
  YTMP = (MY*DY**2)/RY + CY + Y1
  DY = DY + IY*MY
  MX = 1.0
  CX = 0.0
  DX = 0.0
  DO J=1,NXT !total # of nodes in the x direction
    IF(DX .GT. RX) THEN
      MX = -1.0
      CX = 2.0*RX
      DX = DX - CHANGEX*IX
    ENDIF
    XTMP = (MX*DX**2)/RX + CX + X1
    DX = DX + IX*MX
  END DO
END
CORD(NN,1) = XTMP ! R coordinate
CORD(NN,2) = YTMP ! Z coordinate
NN = NN + 1
END DO
END

* ........Nodes input for 1 row or column
ELSEIF(FLAG .EQ. 3) THEN
READ(15,*) N1,X1,Y1,X2,Y2,NT,NINC
IF(NT .EQ. 1) THEN
WRITE(6,*) 'INPUT ERROR, FLAG=3 & NT=1'
WRITE(6,*) 'THIS IS A CONFLICT'
WRITE(16,*) 'INPUT ERROR, FLAG=3 & NT=1'
WRITE(16,*) 'THIS IS A CONFLICT'
STOP
ENDIF
DO I=1,NT
XTMP = -DX + X1
YTMP = -DY + Y1
CORD(NN,1) = XTMP ! R coordinate
CORD(NN,2) = YTMP ! Z coordinate
NN = NN + NINC
END DO

* ........Individual node input
ELSEIF(FLAG .EQ. 4) THEN
READ(15,*) NN,X1,Y1
CORD(NN,1) = X1 ! R coordinate
CORD(NN,2) = Y1 ! Z coordinate
ENDIF

* READ(15,*) FLAG
END DO

* .......Element connection
* Zero quantities
DO EN=1,NE
ISNE(EN) = 0
DO J=1,8 ! maximum # of nodes/element is eight.
NOP(EN,J) = 0
END DO
END DO

* READ(15,*) FLAG
DO WHILE(FLAG .NE. -1) ! flag=-1 -> end of element input
* .......Simple 4 node element input
IF(FLAG .EQ. 1) THEN
READ(15,*) El,NEL,MAT,(GNN(I),I=1,NEL),XINC,YINC,TEX,TEY
EN = El
DO I=1,NEL
YTEMP(I) = GNN(I) - YINC
END DO
DO J=1,TEY
DO I=1,NEL
YTEMP(I) = YTEMP(I) + YINC
XTEMP(I) = YTEMP(I) - XINC
END DO
END DO
DO K=1,TEX
DO I=1,NEL
  XTEMP(I) = XTEMP(I) + XINC
  NOP(EN,I) = XTEMP(I)  ! global node #
END DO
ISNE(EN) = NEL
IMAT(EN) = MAT
EN = EN + 1
END DO
ENDIF
END

READ(*,*) EN, (NOP(EN,I),I=1,NEL), IMAT(EN), ISNE(EN)

DO 16,17,600
16 WRITE(16,600) EN, (NOP(EN,I),I=1,NEL), IMAT(EN), ISNE(EN)
17 END

WRITE(*,16) NP, NE, NB, NLD, NDF, NMAT, NN
WRITE(*,16) (N, (ORT(N,I) = 1.5), N=1,NMAT)
WRITE(*,16) (N, (ORT(N,I) = 6.10), MPB(N), N=1, NMAT)
WRITE(*,16) (NN, (CORD(NN,M), M=1,2), NN=1, NP)
WRITE(*,16) EN
DO 10 EN=1,NE
10 WRITE(*,16) EN, (NOP(EN,I),I=1,NEL), IMAT(EN), ISNE(EN)
16 END

WRITE(*,16) NP, ' = NUMBER OF NODES'
DO NN=1, NP
WRITE(*,16) NN, (CORD(NN,M), M=1,2)
END DO
WRITE(*,16) NE, ' = NUMBER OF ELEMENTS'
DO EN=1,NE
WRITE(*,16) EN, (NOP(EN,I),I=1,NEL), IMAT(EN), ISNE(EN)
END DO

FORMAT(16,*),//5X,'INPUT DATA',//5X,'CONTROL PARAMETERS')
FORMAT(5X,'MATERIAL PROPERTIES')
FORMAT(5X,'NODAL POINTS')
FORMAT(5X,'ELEMENT CONNECTION')
RETURN
END
**Subroutine FORMK**

0001 *
0002 *
0003 SUBROUTINE FORMK
0004 * This section creates the global stiffness \([K]\) but stores
0005 * only half of the matrix as a banded matrix SK. The diagonals
0006 * of \([K]\) are stored as columns in SK.
0007 IMPLICIT REAL*8(A-H,O-Z)
0008 INCLUDE 'COMMON.FOR/LIST,'
0009 1 * MAX - Maximum value for variable
0010 1 INTEGER JTMAX,MATMAX,NEFIX,NNMAX,NBMAX,NLDMAX
0011 1 INTEGER EN,TEX,B CFLAG,KFLAG,ETMAX
0012 1 PARAMETER (JTMAX=20,MATMAX=5,NEFIX=200,NNMAX=400,NBMAX=100)
0013 1 PARAMETER (NLDMAX=600,ETMAX=7)
0014 1 INTEGER TIMES(NLDMAX)
0015 1 INTEGER TYPE(ETMAX),GP(ETMAX),ENUM(ETMAX),ETOTAL
0016 1 REAL*8 LENGTH,DELTA,NORM(MATMAX)
0017 1 REAL*8 DIS(2,NNMAX),DELTA(NLDMAX),DELTAU(NLDMAX)
0018 1 REAL*8 NSTRESS(NNMAX),SKT(2*NNMAX,NBMAX)
0019 1 COMMON TYPE,GP,ENUM,ETOTAL
0020 1 COMMON FPE,NE,NB,NNL,SMAT,NSF,NSBAND,KFLAG
0021 1 COMMON IL,IF,IX,COUNT(NNMAX),IS,TEX,B CFLAG
0022 1 COMMON PIE,ERC,ERREL,RC,DT,JTERM,LG
0023 1 COMMON ORT(MATMAX,10),CORD(NNMAX,2),NOP(NMax,8)
0024 1 COMMON IMAT(NMAMAX),ISNE(NMAX),KPB(MATMAX)
0025 1 COMMON NB(NNMAX),NFIX(NMAX),UBC(NMAX,2)
0026 1 COMMON SK(2*NNMAX,NBMAX),SKS(2*NNMAX,NBMAX)
0027 1 COMMON RL(2*NNMAX),RL(2*NNMAX),VL(2*NNMAX)
0028 1 COMMON LENGTH,DELTA,STRESS,STRAIN,NORM
0029 1 COMMON YO(NMAX,9),Y(1)(NMAX,9),BK1(NMAX,9)
0030 1 COMMON SIG(NMAX,9,4),SIGO(NMAX,9,4)
0031 1 COMMON EVF(NMAX,9),SIG(NMAX,9,4),SEFM(NMAX,9)
0032 1 COMMON EPS(NMAX,9,4),EPSO(NMAX,9,4)
0033 1 COMMON DSV(NMAX,9,4),ESTIFM(16,16)
0034 1 COMMON ERG(JTMAX),ERRELJ(JTMAX)
0035 * ..........Variables
0036 1 * EN - Element number
0037 1 * ESTIFM(I,L) - Local stiffness for row I and collumn L
0038 1 I - Local stiffness row number
0039 1 * ISNE(NE) - Number of nodes/element (a vector)
0040 1 J - Counter for each DOF in a row
0041 1 JJ - Counter for local node numbers
0042 1 K - Counter for each DOF in a collumn
0043 1 KK - Counter for collumn numbers
0044 1 L - Local stiffness collumn number
0045 1 M - Counter
0046 1 MM - Counter
0047 1 NDF - DOF for a node
0048 1 NE - Total number of elements
0049 1 NEL - Number of nodes in element
0050 1 * NCOL - Diagonal # of global \([K]\), collumn number of SK
0051 1 NCOLB - Collumn number
0052 1 * NOP(EN,M) - Global node number for element EN and local node M
0053 1 NROWB - Row number of global \([K]\), row number of SK
0054 1 * SK(NROWB,NCOL) - Banded stiffness
0055 1 * SKS(NROWB,NCOL) - Copy of SK (used if KFLAG=1 since subroutine
0056 1 BNDCN is destructive to SK)
*Initialization (zero matrices)
DO M=1,NSZF
  DO MM=1,NBAND
    SK(M,MM) = 0.0
  END DO
END DO
*Form stiffness matrix in a band form
DO 100 EN=1,NE
  CALL STIFTKEN(EN) ! compute local stiffness, ESTIFM
  NEL = ISNE(EN)
  DO 60 JJ=1,NEL ! counting node number local
    NROWB = (NOP(EN,JJ) - 1)*NDF
    DO J=1,NDF ! for each DOF in a row, ie R,Z
      NROWB = NROWB + 1
    I = (JJ - 1)*NDF + J
    DO KK=1,NEL ! counting column number
      NCOLB = (NOP(EN,KK) - 1)*NDF
      DO K=1,NDF ! for each DOF in a column, ie L
        I = (KK - 1)*NDF + K
        NCOL = NCOLB + K + 1 - NROWB
        IF (NCOL) 40,40,30
        SK(NROWB,NCOL) = SK(NROWB,NCOL) + ESTIFM(I,L)
        CONTINUE
      END DO
    END DO
  END DO
100
CONTINUE
END DO
END DO
END DO
END DO
END DO
END DO
END DO
Copy SK if it is only made once since sub BNDCOND is destructive
IF (KFLAG .EQ. 1) THEN
  DO I=1,NSZF
    DO J=1,NBAND
      SKS(I,J) = SK(I,J)
    END DO
  END DO
ENDIF

write(6,*)'global stiffness'
do i=1,nband
do j=1,nszf
write(6,*) sk(j,i)
end do
do end do
RETURN
END
Subroutine STIFT1

0001  *
0002  *
0003  * SUBROUTINE STIFT1(EN)
0004  * Produces local stiffness
0005  * IMPLICIT REAL*8(A-H,O-Z)
0006  * INCLUDE 'COMMON.FOR/LIST',
0007  1  ! *MAX - Maximum value for variable
0008  1  INTEGER JMAX,MATMAX,NMAX,NBMAX,NLDMAX
0009  1  INTEGER EN,TEX,BCFLAG,KFLAG,ETMAX
0010  1  PARAMETER (JMAX=20,MATMAX=5,NMAX=200,NBMAX=400,NMAX=100)
0011  1  PARAMETER (NLDMAX=600,ETMAX=7)
0012  1  INTEGER TIMES(NLDMAX)
0013  1  INTEGER TYPE(ETMAX),GP(ETMAX),ENUM(ETMAX),ETOTAL
0014  1  REAL*8 LENGTH,DELTAL,NORM(MATMAX)
0015  1  REAL*8 DIS2(NMAX),DELTAT(NLDMAX),DELTAU(NLDMAX)
0016  1  REAL*8 NSTRESS(NMAX),SKT(2*NMAX,NBMAX)
0017  1  COMMON TYPE,GP,ENUM,ETOTAL
0018  1  COMMON NP,NE,NNM,NDF,NMAT,NSZ,F,NBMAX,KFLAG
0019  1  COMMON IL,IT,IP,IQ,COUNT(NMAX),IS,TEX,BCFLAG
0020  1  COMMON PIE,ERC,ERRELC,RRT,DT,JTERM,LM
0021  1  COMMON ORI(MATMAX,10),CORD(NMAX,2),NOP(NMAX,8)
0022  1  COMMON IMAT(NMAX),ISNE(NMAX),MBF(MATMAX)
0023  1  COMMON NBC(NMAX),NFIX(NMAX),UBC(NMAX,2)
0024  1  COMMON SK(2*NMAX,NBMAX),SKS(2*NMAX,NBMAX)
0025  1  COMMON R1(2*NMAX),RL(2*NMAX),UL(2*NMAX)
0026  1  COMMON LENGTH,DELTAL,STRESS,STRAIN,NORM
0027  1  COMMON YO(NMAX,9),Y1(NMAX,9),BKO(NMAX,9),BK1(NMAX,9)
0028  1  COMMON SIG(NMAX,9,4),SIGO(NMAX,9,4)
0029  1  COMMON EVF(NMAX,9),SEF(NMAX,9),SEFM(NMAX,9)
0030  1  COMMON EPST(NMAX,9,4),EPSO(NMAX,9,4)
0031  1  COMMON DSV(NMAX,9,4),ESTIFM(16,16)
0032  1  COMMON ERG(JTMAX),ERRELG(JTMAX)
0033  1  DIMENSION XL(2,8),SHF(3,8),SG(9),TG(9),WG(9)
0034  1  DIMENSION D(3)
0035  *
0036  *.......Variables
0037  * CORD(NN,1) - X coordinate for node NN
0038  * CORD(NN,2) - Y coordinate for node NN
0039  * D(1) - First element in stress-strain matrix (2G + lamda)
0040  * D(2) - Second element in stress-strain matrix (lamda)
0041  * D(3) - Last element in stress-strain matrix (G)
0042  * D## - Elements in stress-strain matrix
0043  * DX - Differential volume
0044  * EE - Elastic modulus
0045  * EN - Element number
0046  * ESTIFM(I,J) - Local Stiffness for row I and column J
0047  * G1 - Shear modulus
0048  * I - Counter
0049  * IMAT(EN) - Material type number (a vector)
0050  * ISNE(EN) - Number of nodes/element for element EN (a vector)
0051  * J - Counter
0052  * k - Counter
0053  * LG - Number of Gauss integration points in one dir
0054  * LINT - Total number of Gauss points
0055  * LL - Material type number
0056  * M - Local node number
0057 * NEL - Number of nodes in element
0058 * NL - Twice NEL (for producing element stiffness)
0059 * NN - Node number (global)
0060 * NOP(EN,M) - Global node number for element EN and local node M
0061 * ORT(LL,1) - Matrix containing material data for each LL material t
0062 * PS - Poisson's ratio
0063 * REV - A revolution (2.0 * Pie * RR)
0064 * RR - Radius (X)
0065 * SG - Natural coordinate (radial or zeta direction)
0066 * SHP(1,I) - X derivative of shape function (I=1,8),
0067 * SHP(2,I) - Y derivative of shape function (I=1,8),
0068 * SHP(3,I) - Shape function in S,T coordinates
0069 * TG - Natural coordinate (longitudinal or eta direction)
0070 * WG - Weighting factor for gauss point
0071 * XL(I,M) - X and Y Coordinates of node, I=1 -> X, I=2 -> Y
0072 * XSJ - Jacobian determinant

*.......Prerequisite data
0075 NEL = ISNE(EN)
0076 NL = NEL*NDF
0077 LL = IMAT(EN)
0078 EE = ORT(LL,1)
0079 PS = ORT(LL,2)
0080 G1 = EE/(2.0*(1.0 + PS))
0081 *.......Initialization (zero matrices)
0082 DO J=1,8
0083 XL(1,J) = 0.0
0084 XL(2,J) = 0.0
0085 DO I=1,3
0086 SHP(I,J) = 0.0
0087 END DO
0088 END DO
0089 DO I=1,NL
0090 DO J=1,NL
0091 ESTIFM(I,J) = 0.0
0092 END DO
0093 END DO
0094 *.......Array of nodal coordinates , XL
0095 DO M=1,NEL
0096 NN = NOP(EN,M)
0097 XL(1,M) = CORD(NN,1) ! X coordinate of local node M
0098 XL(2,M) = CORD(NN,2) ! Y coordinate of local node M
0099 END DO
0100 *.......Stress/strain element matrix (there are only 3 distinct values)
0101 D(1) = EE*(1.0 - PS)/((1.0 + PS)*(1.0 - 2.0*PS))
0102 D(2) = D(1)*PS/(1.0 - PS)
0103 D(3) = EE/(2.0*(1.0 + PS))
0104 *.......Integration points
0105 CALL PGAUSS(LL,LINT,SG,TG,WG)
0106 *.......Evaluate D at each integration point
0107 DO 1000 II=1,LINT ! for each integration point
0108 CALL SHAPEF(SG(II),TG(II),XL,NEL,XSJ,SHP)
0109 RR = 0.0 ! initializing before summing
0110 * for each local node, XL(1,K) is the local RR value
0111 * and DV = 2.0*PIE*RR*XSJ*WG
0112 DO K=1,NEL ! summing over each node
0113 RR = RR + SHP(3,K)*XL(1,K)
0114 END DO
0115 DV = XSJ*WG(II)
0116 REV = 2.0*PIE*RR
*.... Compute [D][B] for each node

DO J=1,NEL ! for each shape function (l/node) 

DB11 = D12*SHP(1,J) + D12*SHP(3,J)/RR
DB21 = D11*SHP(1,J) + D11*SHP(3,J)/RR
DB31 = D12*SHP(1,J) + D12*SHP(3,J)/RR
DB41 = D44*SHP(2,J)

DB12 = D11*SHP(2,J)
DB22 = D12*SHP(2,J)
DB32 = DB22
DB42 = D44*SHP(1,J)

*..... Compute [B] transpose[D][B] for each node

* local stiffness upper 1/2

DO I=1,J

Kij i=1,3,5,7 j=1,3,5,7
ESTIFM(I+I-1,J+J-1) = ESTIFM(I+I-1,J+J-1) +
SHP(1,I)*DB21 + SHP(3,I)*DB31/RR +SHP(2,I)*DB41

Kij i=1,3,5,7 j=2,4,6,8
ESTIFM(I+I-1,J+J ) = ESTIFM(I+I-1,J+J ) +
SHP(1,I)*DB22 + SHP(3,I)*DB32/RR +SHP(2,I)*DB42

Kij i=2,4,6,8 j=1,3,5,7
ESTIFM(I+I,J+J-1) = ESTIFM(I+I,J+J-1) +
SHP(2,I)*DB11 + SHP(1,I)*DB41

Kij i=2,4,6,8 j=2,4,6,8
ESTIFM(I+I,J+J ) = ESTIFM(I+I,J+J ) +
SHP(2,I)*DB12 + SHP(1,I)*DB42

END

END DO

END DO

1000 END DO

*.... Compute lower triangle part via symmetry

DO I=2,NL
DO J=1,I
ESTIFM(I,J) = ESTIFM(J,I)
END DO

END DO

* write(6,*), 'local stiffness for en=',en
do i=1,16
do j=1,16
write(6,*), estifm(j,i)
end do

end do

RETURN
END
Subroutine SHAPEF

0010 * Subroutine SHAPEF
0011 SUBROUTINE SHAPEF(S,T,XL,NEL,XSJ,SHP)
0012 * This subroutine performs mapping and transformations of
0013 * the shape functions (3 or 4 nodes in 2-D)
0014 IMPLICIT *B(A-H,O-Z)
0015 DIMENSION SHP(3,8),XL(2,8),XS(2,2),SI(4),TI(4)
0016 * data for creating shape functions and their derivatives
0017 (first 4 our values fill SI vector, second 4 fill TI vector)
0018 DATA SI, TI/-0.5, 0.5, 0.5, -0.5, -0.5, 0.5, 0.5/
0019 *
0020 * Variables
0021 I - Counter
0022 J - Counter
0023 K - Counter of local node number
0024 NEL - Number of nodes/element
0025 S - Natural coordinate (radial or zeta direction)
0026 SHP(1,I) - S derivative of shape function (I=1,4),
0027 SHP(2,I) - T derivative of shape function (I=1,4),
0028 SHP(3,I) - Shape function in S,T coordinates
0029 SI - Data for creating shape functions
0030 T - Natural coordinate (longitudinal or eta direction)
0031 TEMP - Temporary storage of X derivative of shape function
0032 TI - Data for creating shape functions
0033 XL(I,K) - X and Y Coordinates of node, I=1 -> X, I=2 -> Y
0034 XS(I,J) - Jacobian transpose,
0035 XSJ - Jacobian determinant
0036 * Shape functions & derivatives in natural S,T coordinates
0037 DO I=1,4 ! for each shape function
0038 SHP(3,I) = (0.5 + SI(I)*S)*(0.5 + TI(I)*T)
0039 SHP(1,I) = SI(I)*(0.5 + TI(I)*T)
0040 SHP(2,I) = TI(I)*(0.5 + SI(I)*S)
0041 END DO
0042 * Triangle via colapsing
0043 IF(NEL .EQ. 3) THEN
0044 DO I=1,3 ! for the shape function, S deriv, & T deriv
0045 SHP(I,3) = SHP(I,3) + SHP(I,4)
0046 END DO
0047 ENDIF
0048 * Quadratic isoparametric element
0049 C IF (NEL .GT. 4) CALL SHAP2(S,T,NEL,SHP) ! listed for program exp
0050 * Jacobian array transposed
0051 DO I=1,2 ! I=1 -> X, I=2 -> Y
0052 DO J=1,2 ! J=1 -> S, J=2 -> T
0053 XS(I,J) = 0.0
0054 DO K=1,NEL ! for each node, local node number
0055 XS(I,J) = XS(I,J) + XL(I,K)*SHP(J,K)
0056 END DO
0057 END DO
0058 * Jacobian determinant
0059 XSJ = XS(1,1)*XS(2,2) - XS(1,2)*XS(2,1)
0060 * Transform natural S,T derivatives to X,Y derivatives
0061 * To save space, X,Y derivatives are stored in old S,T

105
derivative locations. Therefore must use TEMP so SHP(1,I)
does not become incorrect.

DO I=1,NEL ! for each node

TEMP = ( XS(2,2)*SHP(1,I) - XS(2,1)*SHP(2,I))/XSJ
SHP(2,I) = (-XS(1,2)*SHP(1,I) + XS(1,1)*SHP(2,I))/XSJ
SHP(1,I) = TEMP

END DO

RETURN

END
Subroutine PGAUSS

* SUBROUTINE PGAUSS(LG,LINT,R,Z,W)
* This subroutine performs gaussian quadrature
* IMPLICIT REAL*8(A-H,O-Z)
* DIMENSION LR(9),LZ(9), LW(9),R(1),Z(1),W(1)
* data for radial (zeta or S) integration
* DATA LR/-1,1,1,-1,0,1,0,-1,0/
* data for longitudinal (eta or T) integration
* DATA LZ/-1,-1,1,-1,0,1,0,0/
* weighting factor data, 4*25 means 25,25,25,25
* DATA LW/4*25,4*40,64/
* Variables
* G - Constant for gauss point evaluation
* H - Constant for gauss point evaluation
* I - Counter
* LG - Number of gauss points in a direction
* LINT - total number of gauss points
* LR - Data for radial (zeta or S) gauss point evaluation
* LW - Data for weighting factor of gauss point
* LZ - Data for longitudinal (eta or T) gauss point eval.
* R - Radial (zeta or S) location of gauss point
* W - weighting factor of gauss point
* Z - Longitudinal (eta or T) location of gauss point
* Total number of gauss points
* LINT = LG*LG
* .......1*1 integration
* IF(LG .EQ. 1) THEN
  R(I) = 0.0
  Z(I) = 0.0
  W(I) = 4.0
  RETURN
* .......2*2 integration
* ELSEIF(LG .EQ. 2) THEN
  G = 1.0/SQRT(3.0)
  DO I=1,4
    R(I) = G*LR(I)
    Z(I) = G*LZ(I)
    W(I) = 1.0
  END DO
  RETURN
* .......3*3 integration
* ELSEIF(LG .EQ. 3) THEN
  G = SQRT(0.6)
  H = 1.0/81.0
  DO I=1,9
    R(I) = G*LR(I)
    Z(I) = G*LZ(I)
    W(I) = H*LW(I)
  END DO
  RETURN
ELSE
WRITE(6,*) 'ERROR IN GAUSS POINT INTEGRATION'
WRITE(6,*) 'GAUSS POINTS MUST BE 1*1, 2*2, OR 3*3'
WRITE(16,*) 'ERROR IN GAUSS POINT INTEGRATION'
WRITE(16,*) 'GAUSS POINTS MUST BE 1*1, 2*2, OR 3*3'
STOP
ENDIF
END
Subroutine BC

0001 *
0002 *
0003 SUBROUTINE BC
0004 IMPLICIT REAL*8(A-H,O-Z)
0005 INCLUDE 'COMMON.FOR/LIST'
0006 1 *
0007 *MAX - Maximum value for variable
0008 INTEGER JTMAX,MATMAX,NEMAX,NNMAX,NBMAX,NLDMAX
0009 INTEGER EN,TX,BCFLAG,KFLAG,ETMAX
0010 PARAMETER (JTMAX=20,MATMAX=5,NEMAX=200,NNMAX=400,NBMAX=100)
0011 PARAMETER (NLDMAX=600,ETMAX=7)
0012 INTEGER TIMES(NLDMAX)
0013 INTEGER TYPE(ETMAX),GP(ETMAX),ENUM(ETMAX),ETOTAL
0014 REAL*8 LENGTH,DELTAL,NORM(MATMAX)
0015 REAL*8 DIS(2,NNMAX),DELTAT(NLDMAX),DELTAL(NLDMAX)
0016 REAL*8 NSTRESS(NNMAX),STRESS(NNMAX,2)
0017 COMMON REAL*8(A-H,0-Z)
0018 COMMON EN,TX,BCFLAG
0019 COMMON IL,IT,IP,IQ,COUNT(NNMAX),IS,TX,BCFLAG
0020 COMMON PI,EC,ERRELC,RRT,DT,JTERM,LL
0021 COMMON ORT(MATMAX,10),CORD(NNMAX,2),NDF(NMAT,8)
0022 COMMON IMAX(NMAT),ISNE(NEMAX),MPB(MATMAX)
0023 COMMON NBC(NNMAX),NUFIX(NNMAX),UBC(NNMAX,2)
0024 COMMON Sk(2*NNMAX,NBMAX),SKS(2*NNMAX,NBMAX)
0025 COMMON Rl(2*NNMAX),RL(2*NNMAX),UL(2*NNMAX)
0026 COMMON LENGTH,DELTAL,STRESS,STRAIN,NORM
0027 COMMON YO(NEMAX,2),YY1(NEMAX,9),BKO(NEMAX,9),BK1(NEMAX,9)
0028 COMMON SIG(NEMAX,9,4),SIGO(NEMAX,9,4)
0029 COMMON EPS(NEMAX,9,4),EPSO(NEMAX,9,4)
0030 COMMON DSV(NEMAX,9,4),ESTIFM(16,16)
0031 COMMON ERG(JTMAX),ERRELC(JTMAX)
0032 INTEGER BCFIX,FLAG
0033 DIMENSION R(2)
0034 *
0035 * .. Variables
0036 * BCFIX - Flag for b.c. data entry, BCFIX=1 -> disp control
0037 * BCINC - Number of large time increments after 1st increment
0038 * DELTAU(I) - Increment of displacement
0039 * DELTAT(I) - Increment of time
0040 * DELTAL - Total change in length of cylinder for uniform compression or tension
0041 * DT - Time increment
0042 * ERG(I) - Error vector used in subroutine OUTPUT
0043 * FLAG - Flag for types of input
0044 * I - Counter for nodes containing b.c.’s
0045 * IC - Position number in global displacement or force vector
0046 * IL - Load step
0047 * J - Counter
0048 * JTERM - Number of steps at which iteration is terminated
0049 * K - Counter
0050 * LENGTH - Length of cylinder
0051 * M - Counter
0052 * NB - Total number of nodes that contain b.c.’s
0053 * NBC(I) - Global node number for b.c. I
0054 * NDF - Number of nodal DOF
0055 * NFIX(I) - Code for displacement type for b.c. I
0056 * NFIX=01 -> Y b.c., NFIX=10 -> X b.c., NFIX=11 -> X Y

109
* NQ - Global node number
* R(1) - Force added in X dir
* R(2) - Force added in Y dir
* RL(IC) - Force value
* STRAIN - Engineering strain in Z dir (%)
* TIMES(K) - Number of small time increments within 1
* large time increment (BINC)
* UBC(I,1) - X displacement value of b.c. I
* UBC(I,2) - Y displacement value of b.c. I

*........Initialization
DO M=1,JTERM
  ERG(M) = 0.0
END DO
DO I=1,NSZF
  UL(I) = 0.0
  RL(I) = 0.0
END DO

* IF(IL .EQ. 1 .OR. BCFLAG .NE. 1) THEN
*........Read time step
  READ(15,*), DT
  WRITE(16,600) IL,DT
*........Force b.c.
  WRITE(16,610)
  READ(15,*), NQ,(R(K),K=1,NDF)
  WRITE(16,*), NQ,(R(K),K=1,NDF)
  DO K=1,NDF
    IC = (NQ-1)*NDF + K
    RL(IC) = R(K)
  END DO
  READ(15,*), FLAG
END DO

*........Displacement b.c.
  READ(15,*), DELTAL
  STRAIN = 200.0*DELTAL/LENGTH  ! engineering strain
  WRITE(16,620)
  READ(15,*) (NBC(I),NFIX(I),(UBC(I,K),K=1,2),I=1,NB)
  WRITE(16,630) (NBC(I),NFIX(I),(UBC(I,K),K=1,2),I=1,NB)
ELSEIF(IL .EQ. 2 .AND. BCFLAG .EQ. 1) THEN
  READ(15,*), BCINC
  DO K=1,BCINC
    READ(15,*), TIMES(K),DELTAT(K),DELTAU(K)
  END DO
ENDIF

* IF(IL .GT. 1 .AND. BCFLAG .EQ. 1) THEN
  IF(IL .EQ. 2) THEN
    K = 1
    ITIME = 1
  ENDIF
  IF(ITIME .GT. TIMES(K)) THEN
    K = K + 1
    ITIME = 1
  ENDIF
  IF(ITIME .EQ. 1) THEN
    DT = DELTAT(K)
    DO I=1,NB

110
IF(UBC(I,2).NE. 0.0) THEN
  UBC(I,2) = DELTAU(K)
ENDIF
END DO

ITIME = ITIME + 1
WRITE(16,600) IL,DT
DELTA = DELTAU(K) + DELTAL
STRAIN = 200.0*DELTAL/LENGTH

CALL BNDCND

FORMAT(/20X,'LOAD CASE',I3,
  /20X,'---------',
  //5X,'TIME STEP=',F13.7)
FORMAT(/5X,'LOADS')
FORMAT(/5X,'DISPL')
FORMAT('0',2I5,2F10.4,10X,2I5,2F10.4)
RETURN
END
Subroutine BNDCND

0001 * 
0002 * 
0003 * SUBROUTINE BNDCND 
0004 * This section places the applied boundary conditions into the matrix formulation. 
0005 * 
0006 IMPLICIT REAL*8(A-H,O-Z) 
0007 INCLUDE 'COMMON.FOR/LIST' 
0008 1 * *MAX - Maximum value for variable 
0009 1 INTEGER JTMAX,MATMAX,NMAX,NNMAX,NBMAX,NLDMAX 
0010 1 INTEGER EN,TEX,BCFLAG,KFLAG,ETMAX 
0011 1 PARAMETER (JTMAX=20, MATMAX=5, NMAX=200, NNMAX=400, NBMAX=100) 
0012 1 PARAMETER (NLDMAX=500, ETMAX=7) 
0013 1 INTEGER TIMES(NLDMAX) 
0014 1 INTEGER TYPE(ETMAX), GP(ETMAX), ENUM(ETMAX), ETOTAL 
0015 1 REAL*8 LENGTH, DELTAL, NORM(MATMAX) 
0016 1 REAL*8 DIS(2,NNMAX), DELTAT(NLDMAX), DELTAU(NLDMAX) 
0017 1 REAL*8 NSTRESS(NNNMAX), SKT(2*NMAX, NBMAX) 
0018 1 COMMON TYPE, GP, ENUM, ETOTAL 
0019 1 COMMON NP, NE, NB, NLD, NDF, NMAT, NSZF, NBAND, KFLAG 
0020 1 COMMON IL, IT, IP, IQ, COUNT(NNMAX), IS, TEX, BCFLAG 
0021 1 COMMON PIE, ERC, ERRELG, RRT, DT, JTERM, LG 
0022 1 COMMON ORT(MATMAX,10), CORD(NNMAX,2), NOP(NMAX,8) 
0023 1 COMMON IMAT(NMAX), ISNE(NMAX), MPB(MATMAX) 
0024 1 COMMON NBC(NNNMAX), NFIX(NNNMAX), UBC(NNNMAX,2) 
0025 1 COMMON SK(2*NNNMAX, NBMAX), SKS(2*NNNMAX, NBMAX) 
0026 1 COMMON RL(2*NNNMAX), RL(2*NNNMAX), UL(2*NNNMAX) 
0027 1 COMMON LENGTH, DELTAL, STRESS, STRAIN, NORM 
0028 1 COMMON YO(NMAX,9), Y1(NMAX,9), BK0(NMAX,9), BK1(NMAX,9) 
0029 1 COMMON SIG(NMAX,9,4), SIGO(NMAX,9,4) 
0030 1 COMMON EVF(NMAX,9), SEF(NMAX,9), SEFM(NMAX,9) 
0031 1 COMMON EPS(NMAX,9,4), EPSO(NMAX,9,4) 
0032 1 COMMON DSV(NMAX,9,4), ESTIFM(16,16) 
0033 1 COMMON ERG(JTMAX), ERRELG(JTMAX) 
0034 INTEGER FLAG 
0035 * ...... Variables 
0036 * I - Counter for nodes containing b.c.'s 
0037 * IC - Position number in global displacement or force vector 
0038 * ICON - Integer used in evaluating b.c. type 
0039 * IL - Load case 
0040 * J - Counter 
0041 * JH - Counter 
0042 * K - Counter 
0043 * M - Counter 
0044 * NB - Total number of nodes that contain b.c.'s 
0045 * NBC(I) - Global node number for b.c. I 
0046 * NCODE - Code for displacement type, same as NFIX(I) 
0047 * NDF - Number of nodal DOF 
0048 * NFIX(I) - Code for displacement type for b.c. I 
0049 * NFIX=01 -> Y b.c., NFIX=10 -> X b.c., NFIX=11 -> X Y 
0050 * NQ - Global node number 
0051 * NR - Counter 
0052 * NROWB - Counter for rows 
0053 * NSZF - Total DOF for system, length of SK 
0054 * NX - Integer used to evaluate b.c. type 
0055 * RL(IC) - Force value 
0056 * SK(NROWB, J) - Banded stiffness
* SKS(NROWB,J) - Copy of SK (used if KFLAG=1 since subroutine
* BNDCNDS is destructive to SK
* UBC(I,1) - X displacement value of b.c. I
* UBC(I,2) - Y displacement value of b.c. I
* UL(IC) - Displacement value
* *
* *....Global displacements
* DO I=1,NB ! for each node that has a b.c.
  NQ = NBC(I)
  DO K=1,NDF ! for each nodal DOF
    IC = (NQ - 1)*NDF + K
    UL(IC) = UBC(I,K)
  END DO
END DO

*.....Replace copy of SKS into SK if KFLAG=1
IF(KFLAG .EQ. 1) THEN
  DO I=1,NSZF
    DO J=1,NBAND
      SK(I,J) = SKS(I,J)
    END DO
  END DO
ENDIF

*.....Insert b.c.
DO 200 N=1,NB
  NX = 10**((NDF - 1)
  I = NBC(N)
  NROWB = (I - 1)*NDF
  NCODE = NFIX(N)
  DO 90 M=1,NDF
    NROWB = NROWB + 1
    ICON = NCODE/NX ! note: for integers, 3/4=0, 3/3=1, 4/3=
    checking if SK and RL need modification for adding b.c.
    if icon<0 goto 80, if icon=0 goto 80, if icon>0 goto 60
    IF(ICON) 80,80,60
    80,80,60
    SK(NROWB,1) = 1.0
    60
    RL(NROWB) = UL(NROWB)
    DO 70 J=2,NBAND
      making rows zero
      JH = NROWB + J - 1
      IF(JH .LE. NSZF) THEN
        RL(JH) = RL(JH) - SK(NROWB,J)*UL(NROWB)
        SK(NROWB,J) = 0.0
      ENDIF
      making columns zero
      NR = NROWB + 1 - J
      IF(NR .GT. 0) THEN
        RL(NR) = RL(NR) - SK(NR,J)*UL(NROWB)
        SK(NR,J) = 0.0
      ENDIF
    70
  END DO
  NCODE = NCODE - NX*ICON
  80
  NX = NX/10
  90
END DO
DO 200
  END DO

*.....Format
RETURN
Subroutine LOADS

0001 *  
0002 *  
0003 * SUBROUTINE LOADS  
0004 * This section calculates the viscous load  
0005 IMPLICIT REAL*8(A-H,O-Z)  
0006 INCLUDE 'COMMON.FOR/LIST'  
0007 1 *  
0008 1 INTEGER JTMAX, MATMAX, NEMAX, NNMAX, NBMAX, NLDMAX  
0009 1 INTEGER EN, TEX, BCFLAG, KFLAG, ETMAX  
0010 1 PARAMETER (JTMAX=20, MATMAX=5, NEMAX=200, NNMAX=400, NBMAX=100)  
0011 1 PARAMETER (NLDMAX=600, ETMAX=7)  
0012 1 INTEGER TIMES(NLDMAX)  
0013 1 INTEGER TYPE(ETMAX), GP(ETMAX), ENUM(ETMAX), ETOTAL  
0014 1 REAL*8 LENGTH, DELTAL, NORM(MATMAX)  
0015 1 REAL*8 DIS(2,NNMAX), DELTAT(NLDMAX), DELTAU(NLDMAX)  
0016 1 REAL*8 NSTRESS(NNMAX), SKT(2*NNMAX, NBMAX)  
0017 1 COMMON TYPE, GP, ENUM, ETOTAL  
0018 1 COMMON NP, NE, NB, NLD, NDF, NMAT, NSZF, NBAND, KFLAG  
0019 1 COMMON IL, IT, IP, IQ, COUNT(NNMAX), IS, TEX, BCFLAG  
0020 1 COMMON PIE, ERC, ERRELC, RT, DT, JTERM, LG  
0021 1 COMMON ORT(MATMAX, 10), CORD(NNMAX, 2), NIP(NMAT, 8)  
0022 1 COMMON IMAT(NEMAX), ISNE(NEMAX), MPB(MATMAX)  
0023 1 COMMON NBC(NNMAX), NFIX(NNMAX), UBC(NNMAX, 2)  
0024 1 COMMON SK(2*NNMAX, NBMAX), SRS(2*NNMAX, NBMAX)  
0025 1 COMMON RL(2*NNMAX), RL(2*NNMAX), UL(2*NNMAX)  
0026 1 COMMON LENGTH, DELTAL, STRESS, STRAIN, NORM  
0027 1 COMMON YYO(NEMAX, 9), YY1(NEMAX, 9), BKO(NMAT, 9), BK1(NMAT, 9)  
0028 1 COMMON SIG(NE, 9, 4), SIGO(NMAT, 9, 4)  
0029 1 COMMON EVF(NMAT, 9, 9), EPSO(NMAT, 9, 9)  
0030 1 COMMON EPSI(NEMAX, 9, 4), EPSO(NMAT, 9, 4)  
0031 1 COMMON DSV(NMAT, 9, 4), ESTIFM(16, 16)  
0032 1 COMMON DSV(JTMAX), ERRELG(JTMAX)  
0033 1 DIMENSION XL(2, 8), SHP(3, 8), SG(9), TG(9), WG(9)  
0034 1 DIMENSION FV(16)  
0035 1 *----------Variables  
0036 1 CORD(NN, 1) - X coordinate for node NN  
0037 1 CORD(NN, 2) - Y coordinate for node NN  
0038 1 DSV(EN, II, J) - Incremental viscous stress for element EN,  
0039 1 gauss point II, and direction J  
0040 1 DV - Differential volume  
0041 1 EN - Element number  
0042 1 FV(I) - Viscous force vector  
0043 1 I - Local row number  
0044 1 II - Counter for Gauss points  
0045 1 ICON - Integer used in evaluating b.c. type  
0046 1 ISNE(EN) - Number of nodes/element for element EN (a vector)  
0047 1 J - Counter  
0048 1 JJ - Counter  
0049 1 K - Counter  
0050 1 LG - Number of Gauss integration points in one dir  
0051 1 LINT - Total number of Gauss points  
0052 1 M - Counter  
0053 1 NB - Total number of nodes that contain b.c.'s  
0054 1 NBC(N) - Global node number for b.c. N  
0055 1 NCODE - Code for displacement type, same as NFIX(I)  

114
0057 * NDF - Number of nodal DOF
0058 * NE - Total number of elements
0059 * NEL - Number of nodes in element
0060 * NFIX(N) - Code for displacement type for b.c. N
0061 * NFIX=01 -> X b.c., NFIX=10 -> Y b.c., NFIX=11 -> X Y
0062 * NL - Twice NEL (for producing element stiffness)
0063 * NN - Node number (global)
0064 * NOP(EN,M) - Global node number for element EN and local node M
0065 * NROWB - Counter for rows
0066 * NSZF - Total DOF for system, length of SK
0067 * NX - Integer used to evaluate b.c. type
0068 * R1(K) - Force vector
0069 * REV - A revolution (2.0 * Pie * RR)
0070 * RR - Radius (X)
0071 * SG - Natural coordinate (radial or zeta direction)
0072 * SHP(1,I) - X derivative of shape function (I=1,8),
0073 * SHP(2,I) - Y derivative of shape function (I=1,8),
0074 * SHP(3,I) - Shape function in S,T coordinates
0075 * TG - Natural coordinate (longitudinal or eta direction)
0076 * WG - Weighting factor for gauss point
0077 * XL(I,M) - X and Y Coordinates of node, I=1 -> X, I=2 -> Y
0078 * XSJ - Jacobian determinant
0079 *
0080 *.......Initialization (zero vector)
0081 DO K=1,NSZF
0082 R1(K) = 0.0
0083 END DO
0084 DO 2000 EN=1,NE
0085 *.......Prerequisite data
0086 NEL = ISNE(EN)
0087 NL = NEL + NEL
0088 *.......Initialization (zero matrices)
0089 DO J=1,8
0090 XL(1,J) = 0.0
0091 XL(2,J) = 0.0
0092 DO M=1,3
0093 SHP(M,J) = 0.0
0094 END DO
0095 END DO
0096 DO K=1,NL
0097 FV(K) = 0.0
0098 END DO
0099 *.......Array of nodal coord. xl
0100 DO M=1,NEL
0101 NN = NOP(EN,M)
0102 XL(1,M) = CORD(NN,1) ! local X coordinate
0103 XL(2,M) = CORD(NN,2) ! local Y coordinate
0104 END DO
0105 *.......Gauss numerical integration
0106 CALL PGAUSS(LG,LINT,SG,TG,WG)
0107 DO 1000 II=1,LINT ! for each gauss point
0108 CALL SHAPEF(SIG(II),TG(II),XL,NEL,XSJ,SHP)
0109 RR = 0.0
0110 DO K=1,NL
0111 RR = RR + SHP(3,K)*XL(1,K)
0112 END DO
0113 DV = XSJ*WG(II)
0114 REV = 2.0*Pie*RR
0115 DV = DV*REV
0116 *.......Viscous load at each node (local)
DO KS=1, NEL ! for each node in the element
    K2 = 2*KS ! Z direction
    K1 = K2 - 1 ! R direction
    FV(K1) = FV(K1) + (SHP(1, KS)*DSV(EN, II, 2))
    + SHP(3, KS)*DSV(EN, II, 3)/RR
    + SHP(2, KS)*DSV(EN, II, 4))*DV
    FV(K2) = FV(K2) + (SHP(2, KS)*DSV(EN, II, 1))
    + SHP(1, KS)*DSV(EN, II, 4))*DV
END DO

* load at each node global
DO JJ=1, NEL ! for each local node number
NROWB = (NOP(EN, JJ) - 1)*NDF ! global row number
DO J=1, NDF
    NROWB = NROWB + 1
    I = (JJ - 1)*NDF + J ! local row number
    RL(NROWB) = RL(NROWB) + FV(I)
END DO
END DO

* insert b.c. (for the sake of accuracy)
* If viscous force appears at node where displacement was
* specified, make RL=0 so that b.c. is upheld.
DO 100 N=1, NB
    NX = 10**((NDF - 1)
    I = NBC(N)
    NROWB = (I - 1)*NDF
    NCODE = NFIX(N)
    DO M=1, NDF
        NROWB = NROWB + 1
        ICON = NCODE/NX
        IF(ICON .GT. 0) THEN
            RL(NROWB) = 0.0
        NCODE = NCODE - NX*ICON
        ENDIF
    END DO
    NX = NX/10
    100 END DO

* RETURN
END
Subroutine SOLVE

0001 *
0002 *
0003 SUBROUTINE SOLVE
0004 * This section uses a special version of Gauss Eleimination,
0005 * set up for a banded matrix stored diagonally, to find the
0006 * incremental displacements
0007 IMPLICIT REAL*8(A-H,O-Z)
0008 INCLUDE 'COMMON.FOR/LIST'
0009 1* MAX - Maximum value for variable
0010 1 INTEGER JTMAX,MATMAX,NMAX,NBMAX,NLMAX
0011 1 INTEGER EN,TEX,BCFLAG,KFLAG,ETMAX
0012 1 PARAMETER (JTMAX=20,MATMAX=5,NMAX=200,NBMAX=400,NLMAX=100)
0013 1 PARAMETER (NLMAX=600,ETMAX=7)
0014 1 INTEGER TIMES(NLMAX)
0015 1 INTEGER TYPE(ETMAX),GP(ETMAX),ENUM(ETMAX),ETOTAL
0016 1 REAL*8 LENGTH,DELTA,NORM(MATMAX)
0017 1 REAL*8 DIS(2,NMAX),DELTA(NLMAX),DELTAT(NLMAX)
0018 1 REAL*8 NSTRESS(NMAX),SKT(2*NMAX,NBMAX)
0019 1 COMMON TYPE,GP,ENUM,ETOTAL
0020 1 COMMON NF,NE,NB,NLD,NDF,NNAT,NSZF,NBAND,KFLAG
0021 1 COMMON IL,IT,IP,IQ,COUNT(NMAX),IS,TEX,BCFLAG
0022 1 COMMON PIE,ERC,ERRELG,RRT,DT,JTERM,LG
0023 1 COMMON ORT(MATMAX,10),CORD(NMAX,2),NOP(NMAX,8)
0024 1 COMMON IMAT(NMAX),ISNE(NMAX),MPB(MATMAX)
0025 1 COMMON NBC(NMAX),NFIK(NMAX),UBC(NMAX,2)
0026 1 COMMON SK(2*NMAX,NBMAX),SKS(2*NMAX,NBMAX)
0027 1 COMMON RL(2*NMAX),RL(2*NMAX),UL(2*NMAX)
0028 1 COMMON LENGTH,DELTA,NORM(MATMAX),SK(NMAX,NBMAX)
0029 1 COMMON YO(NMAX,9),YY1(NMAX,9),BKO(NMAX,9),BKL(NMAX,9)
0030 1 COMMON SIG(NMAX,9,4),SIGO(NMAX,9,4)
0031 1 COMMON EPS(NMAX,9,4),EPSO(NMAX,9,4)
0032 1 COMMON EPST(NMAX,9,4),EPSO(NMAX,9,4)
0033 1 COMMON ESF(NMAX,9,4),ESTIFM(16,16)
0034 1 COMMON ERG(JTMAX),ERRELG(JTMAX)
0035 * ....... Variables
0036 * C - Value of Element in SKT
0037 * I - Counter
0038 * J - Counter
0039 * JH - Counter
0040 * K - Counter
0041 * L - Counter
0042 * N - Counter
0043 * NBAND - Band width
0044 * NH - Counter
0045 * NSZF - Total DOF for system, length of SK
0046 * RL - Load from BNDCND routine
0047 * RL - Loads (at beginning of routine)
0048 * Incremental displacements (at end of routine)
0049 * SK(I,J) - Banded stiffness
0050 * SKT(I,J) - Banded stiffness (copy of SK since gauss is destructiv
0051 * ** Special note: R1 comes into this program as a force (rhs of equation
0052 * but leave the program as the incremental
0053 * displacements (solution)
0054 * **
0055 * .......Total load R1 and copy SK matrix to SKT matrix

117
since gauss is destructive.

DO NH=1,NSZF

* total load = loads from BNDCND + viscous loads from LOADS

RL(NH) = RL(NH) + RL(NH)

DO JH=1,NBAND

SKT(NH,JH) = SK(NH,JH)

END DO

END DO

* .......Solution via Gauss elimination

DO 300 N=1,NSZF

I = N

DO 290 L=2,NBAND

I = I + 1

IF(SKT(N,L).NE.0.0) THEN

C = SKT(N,L)/SKT(N,1)

J = 0

DO K=L,NBAND

J = J + 1

IF(SKT(N,K).NE.0.0) THEN

SKT(I,J) = SKT(I,J) - C*SKT(N,K)

ENDIF

END DO

SKT(N,L) = C

R1(I) = R1(I) - C*R1(N)

ENDIF

END DO

R1(N) = R1(N)/SKT(N,1)

END DO

DO WHILE(N.GT.0)

L = N

DO 100 K=2,NBAND

L = L + 1

IF(SKT(N,K).NE.0.0) THEN

R1(N) = R1(N) - SKT(N,K)*R1(L)

ENDIF

END DO

N = N - 1

END DO

RETURN

END
Subroutine OUTPUT

* IMPLICIT REAL*8(A-H,O-Z)
* INCLUDE 'COMMON.FOR/LIST'

* *MAX - Maximum value for variable

* INTEGER JMAX, MATMAX, NMAX, NNMAX, NBMAX, NLDMAX
* INTEGER EN, TEX, B CFLAG, KFLAG, ETMAX
* INTEGER TIMES(NLDMAX)

* REAL*8 LENGTH, DELTAL, NORM(MATMAX)
* REAL*8 DIS(2, NNMAX), DELTA(NLDMAX), DELTAU(NLDMAX)
* REAL*8 NSTRESS(NNMAX, SKT(2*NNNMAX, NBMAX)

* COMMON TYPE, GP, ENUM, ETOTAL
* COMMON NP, NE, NB, NLD, NDF, NMAT, NSZF, N BAND, KFLAG
* COMMON IL, IT, IP, IQ, COUNT(NNMAX), IS, TEX, B CFLAG
* COMMON PIE, ERC, ERRELC, RRT, DT, JTMAX, LG

* COMMON ORT(MATMAX, 10), CORD(NNMAX, 2), NOP(NMAX, 8)
* COMMON IMAT(NMAX), ISN E(NMAX), MPB(MATMAX)
* COMMON BNC(NNMAX), N FIX(NNMAX), UBC(NNMAX, 2)

* COMMON SK(2*NNNMAX, NBMAX), SKS(2*NNNMAX, NBMAX)
* COMMON R1(2*NNNMAX), RL(2*NNNMAX), UL(2*NNNMAX)

* COMMON LENGTH, DELTAL, STRESS, STRAIN, NORM
* COMMON YYO(NMAX, 9), Y11(NMAX, 9), BK0(NMAX, 9), BK1(NMAX, 9)
* COMMON SIG(NMAX, 9, 4), SIG0(NMAX, 9, 4)

* COMMON EVF(NMAX, 9), SEF(NMAX, 9)
* COMMON EPS(NMAX, 9, 4), EPSO(NMAX, 9, 4)

* COMMON DSV(NMAX, 9, 4), ESTIFM(16, 16)

* COMMON ERG(JMAX), ERRELG(JTOTAL)

* EQUIVALENCE (DIS(1), Ri(1))

* DIMENSION XL(2, 8), SHP(3, 8), SG(9), TG(9), WG(9)

* DIMENSION R(16), D(3)

* DIMENSION EPS(4), DSIG(4), SIGM(4), SD(4), SDO(4), DEVP(4)

* ....... Variables

* BKO(EN, I) - Effective viscoplastic stress in element EN
* at Gauss point I at end of load step

* BK1(EN, I) - Effective viscoplastic stress in element EN
* at Gauss point I at current time

* BKM - Effective viscoplastic strain at midpoint in time

* CORD(NN, 1) - X coordinate for node NN

* CORD(NN, 2) - Y coordinate for node NN

* D(1) - First element in stress-strain matrix (2G + lamda)
* D(2) - Second element in stress-strain matrix (lamda)

* D(3) - Last element in stress-strain matrix (G)

* DEVP(K) - Incremental viscoplastic strain

* DSIG(K) - Incremental stress

* DSV(EN, I, K) - Incremental viscous stress in element EN at
* Gauss point I

* DT - Time increment

* EN - Element number

* EPS(K) - Incremental strain

* EPSO(EN, I, K) - Total strain in element EN at Gauss point I at
* end of load step

* EPST(EN, I, K) - Total strain in element EN at Gauss point I
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERO</td>
<td>Maximum absolute error up till previous iteration</td>
</tr>
<tr>
<td>ERL1</td>
<td>Maximum absolute error in current iteration</td>
</tr>
<tr>
<td>ERC</td>
<td>Error criterion for termination iteration (absolute)</td>
</tr>
<tr>
<td>ERG(IT)</td>
<td>Maximum absolute error up till current iteration</td>
</tr>
<tr>
<td>ERRELO</td>
<td>Maximum relative error up till previous iteration</td>
</tr>
<tr>
<td>EREL1</td>
<td>Maximum relative error in current iteration</td>
</tr>
<tr>
<td>ERRELG(IT)</td>
<td>Maximum relative error up till current iteration</td>
</tr>
<tr>
<td>ERELT</td>
<td>Relative error in effective stress SEFM</td>
</tr>
<tr>
<td>ERV</td>
<td>Absolute error in effective strain rate ERV</td>
</tr>
<tr>
<td>ERRELV</td>
<td>Relative error in effective strain rate ERV</td>
</tr>
<tr>
<td>EVF(EN,I)</td>
<td>Effective strain rate</td>
</tr>
<tr>
<td>GI</td>
<td>Shear modulus</td>
</tr>
<tr>
<td>I</td>
<td>Counter</td>
</tr>
<tr>
<td>IL</td>
<td>Counter for load steps</td>
</tr>
<tr>
<td>IMAT(EN)</td>
<td>Material type number (a vector)</td>
</tr>
<tr>
<td>IP</td>
<td>Frequency of printing after first load step</td>
</tr>
<tr>
<td>IQ</td>
<td># of load steps from last print (when IQ=IP -&gt; print)</td>
</tr>
<tr>
<td>ISNE(EN)</td>
<td>Number of nodes/element for element EN (a vector)</td>
</tr>
<tr>
<td>IT</td>
<td>Counter for iterations during a load step</td>
</tr>
<tr>
<td>JTERM</td>
<td># of steps at which iteration is terminated</td>
</tr>
<tr>
<td>K</td>
<td>K=1 -&gt; Z, K=2 -&gt; R, K=3 -&gt; THETA, K=4 -&gt; RZ</td>
</tr>
<tr>
<td>LG</td>
<td># of Gauss points for Gauss integration (in one direct</td>
</tr>
<tr>
<td>LH</td>
<td>Counter for Gauss points</td>
</tr>
<tr>
<td>LINT</td>
<td>Total number of Gauss points</td>
</tr>
<tr>
<td>LL</td>
<td>Material type number</td>
</tr>
<tr>
<td>MP</td>
<td>Power index for viscoplastic strain rate power formula</td>
</tr>
<tr>
<td>MPB(LL)</td>
<td>Vector of MP for different material types</td>
</tr>
<tr>
<td>NBAND</td>
<td>Half band width of the global stiffness matrix</td>
</tr>
<tr>
<td>NDF</td>
<td>Number of degrees of freedom at a node</td>
</tr>
<tr>
<td>NE</td>
<td>Total number of elements</td>
</tr>
<tr>
<td>NEL</td>
<td>Number of nodes in an element</td>
</tr>
<tr>
<td>NL</td>
<td>Number of DOF for an element</td>
</tr>
<tr>
<td>NLD</td>
<td>Total number of load steps</td>
</tr>
<tr>
<td>NN</td>
<td>Node number (global)</td>
</tr>
<tr>
<td>NOP(EN,M)</td>
<td>Global node number for element EN and local node M</td>
</tr>
<tr>
<td>NORM</td>
<td>Normalization factor (Initial Drag value)</td>
</tr>
<tr>
<td>NP</td>
<td>Total number of nodal points</td>
</tr>
<tr>
<td>NDF</td>
<td>Total degrees of freedom for system</td>
</tr>
<tr>
<td>ORT(LL,I)</td>
<td>Matrix containing material data for each LL material t</td>
</tr>
<tr>
<td>RE1</td>
<td>X coordinate of third local node</td>
</tr>
<tr>
<td>RE2</td>
<td>X coordinate (2/3 between local node 4 and 1)</td>
</tr>
<tr>
<td>RE3</td>
<td>X coordinate (1/3 between local node 4 and 1)</td>
</tr>
<tr>
<td>RE4</td>
<td>X coordinate of fourth local node</td>
</tr>
<tr>
<td>REM</td>
<td>X coordinate (midpoint between local node 3 and 4)</td>
</tr>
<tr>
<td>RR</td>
<td>Radius (X)</td>
</tr>
<tr>
<td>RRT</td>
<td>Radius of cylinder</td>
</tr>
<tr>
<td>SD(K)</td>
<td>Deviatoric stress tensor</td>
</tr>
<tr>
<td>SDO(K)</td>
<td>Deviatoric stress tensor at last step in time</td>
</tr>
<tr>
<td>SEF(EN,I)</td>
<td>Effective stress for element EN at Gauss point I</td>
</tr>
<tr>
<td>SEFM(EN,I)</td>
<td>Same as SEF(EN,I) but at midpoint in time (used for time integration)</td>
</tr>
<tr>
<td>SG</td>
<td>Natural coordinate (radial or zeta direction)</td>
</tr>
<tr>
<td>SHP(1,I)</td>
<td>X derivative of shape function (I=1,8),</td>
</tr>
<tr>
<td>SHP(2,I)</td>
<td>Y derivative of shape function (I=1,8),</td>
</tr>
<tr>
<td>SHP(3,I)</td>
<td>Shape function in S,T coordinates</td>
</tr>
<tr>
<td>SIG(EN,I,K)</td>
<td>Stress for element EN at Gauss point I</td>
</tr>
<tr>
<td>SIGM(K)</td>
<td>Stress at midpoint in time</td>
</tr>
<tr>
<td>SIGO(EN,I,K)</td>
<td>Stress for element EN at Gauss point I at end of</td>
</tr>
</tbody>
</table>
0117 * load step
0118 * SM  - Mean stress
0119 * STRAIN  - Engineering strain in Z dir (%)
0120 * STRESS  - Engineering stress in Z dir
0121 * SUM  - Summation
0122 * SUMM  - Total force across cylinder
0123 * TE1  - Old value (last iteration) of SEFM
0124 * TE2  - New value (current iteration) of SEFM
0125 * TEX  - Total number of elements in X dir.
0126 * TG  - Natural coordinate (longitudinal or eta direction)
0127 * V1  - Old value (last iteration) of EVF
0128 * V2  - New value (current iteration) of EVF
0129 * WG  - Weighting factor for gauss point
0130 * XL(I,M)  - X and Y Coordinates of node, I=1 -> X, I=2 -> Y
0131 * XSJ  - Jacobian determinant
0132 * YYO(EN,I)  - Drag stress in element EN at Gauss point I
0133 * at end of load step
0134 * YY1(EN,I)  - Drag stress in element EN at Gauss point I
0135 * at current time step
0136 * YYM  - Drag stress at midpoint in time
0137 * ZZ  - Length (Y)
0138 *
0139 * .........Material data
0140 * ORT(1) = EE  - Elastic modulus
0141 * ORT(2) = PS  - Poisson ratio
0142 * ORT(5) = YS  - The initial static drag stress
0143 * ORT(6) = Al  - Rise time constant for drag stress
0144 * ORT(7) = Bl  - Constant for drag stress rate equation
0145 * ORT(10) = GM  - Strain rate saturation constant
0146 *
0147 * DO 1100 EN=1,NE
0148 * .........Prerequisite parameters and variables
0149 NEL = ISNE(EN)
0150 NL = NEL*NDF
0151 LL = IMAT(EN)
0152 EE = ORT(LL,1)
0153 PS = ORT(LL,2)
0154 Al = ORT(LL,6)
0155 Bl = ORT(LL,7)
0156 GM = ORT(LL,10)
0157 MP = MPB(LL)
0158 Gl = 0.5*EE/(1.0 + PS)
0159 D(1) = EE*(1.0 - PS)/((1.0 + PS)*(1.0 - 2.0*PS))
0160 D(2) = D(1)*PS/(1.0 - PS)
0161 D(3) = Gl
0162 IF(IL.EQ.1 .AND. IT .EQ. 1 .AND. EN .EQ. 1) THEN
0163 WRITE(16,600)
0164 END IF
0165 * .........Initialization
0166 DO K=1,NL
0167 R(K) = 0.0
0168 END DO
0169 DO M=1,NEL
0170 XL(1,M) = 0.0
0171 XL(2,M) = 0.0
0172 DO I=1,3
0173 SHP(I,M) = 0.0
0174 END DO
0175 END DO
0176 * .........Array of nodal coordinates
DO M=1,NEL
   NN = NOP(EN,M)
   XL(1,M) = CORD(NN,1)
   XL(2,M) = CORD(NN,2)
   J = (M-1)*NDF
   DO I=1,NDF
      K = J + I
      R(K) = DIS(I,NN)
   END DO
   END DO

*.......Radial distances required for engineering stress
IF(EN.EQ.1) SUMM=0.0
RE4 = XL(1,3) ! x coordinate of third local node
RE1 = XL(1,4) ! x coordinate of fourth local node
RE2 = (2.0*RE1 + RE4)/3.0
RE3 = (RE1 + 2.0*RE4)/3.0
REM = (RE1 + RE4)/2.0

*.......Numerical integration
CALL PEAUSS(LG,LINT,SG,TG,WG)
DO 1000 LH=1,LINT
   CALL SHAPEF(SG(LH),TG(LH),XL,NEL,XS,J,SHP)
1000继续

*.......Coordinates RR,ZZ & incremental strains EPS
   RR = 0.0
   ZZ = 0.0
   DO I=1,4
      EPS(I) = 0.0
   END DO
   DO J=1,NEL
      RR = RR + SHP(3,J)*XL(1,J)
      ZZ = ZZ + SHP(3,J)*XL(2,J)
      EPS(1) = EPS(1) + SHP(2,J)*R(2*J)
      EPS(2) = EPS(2) + SHP(1,J)*R(2*J-1)
      EPS(3) = EPS(3) + SHP(3,J)*R(2*J-1)
      EPS(4) = EPS(4) + SHP(2,J)*R(2*J-1)+SHP(1,J)*R(2*J)
   END DO
   EPS(3) = EPS(3)/RR
   DO K=1,4
      EPST(EN,LH,K) = EPSO(EN,LH,K) + EPS(K)
   END DO
   IF(IL.EQ.1 .AND. IT.EQ.1) THEN
      WRITE(16,610) EN,LH,RR,ZZ
   END IF

*.......Effective strain rate EVF
   SUM = 0.0
   DO K=1,4
      SUM = SUM + (EPS(K)/DT)**2
   END DO
   SUM = SUM - 0.5*(EPS(4)/DT)**2
   EPS(4) = angular distortion (sig/g)
   V1 = EVF(EN,LH) ! old value, last iteration
   EVF(EN,LH) = DSQRT(0.5*SUM)
   V2 = EVF(EN,LH) ! new value, current iteration
   ERV = DABS(V2-V1) ! absolute error
   ERRELV = 1.0E10 ! relative error default if V2=0
   IF(V2 .NE. 0.0) THEN
      ERRELV = ERV/V2 ! relative error
   END IF
   ENDIF

*.......Incremental stress DSIG & total stress SIG
   DSIG(1) = D(1)*EPS(1) + D(2)*EPS(2) + D(2)*EPS(3)
   DSIG(2) = D(2)*EPS(1) + D(1)*EPS(2) + D(2)*EPS(3)
DSIG(3) = D(2)*EPS(1) + D(2)*EPS(2) + D(1)*EPS(3)

DSIG(4) = D(3)*EPS(4)

DO K=1,4
    DSIG(K) = DSIG(K) - DSV(EN,LH,K)
    SIG(EN,LH,K) = SIGO(EN,LH,K) + DSIG(K)
END DO

*.....Deviatoric stress SD & mean stress SM (at current step in time)
SM = (SIG(EN,LH,1) + SIG(EN,LH,2) + SIG(EN,LH,3))/3.0

DO K=1,3
    SD(K) = SIG(EN,LH,K) - SM
END DO
SD(4) = SIG(EN,LH,4) ! off diagonal

*.....Effective stress SEF (at current time step)
SUM = 0.0
DO K=1,3
    SUM = SUM + SD(K)*SD(K)
END DO
SUM = SUM + 2.0*SD(4)*SD(4)
SEF(EN,LH) = DSQRT(1.5*SUM)

*.....Deviatoric stress SDO & mean stress SM (at last step in time)
SM = (SIGO(EN,LH,1) + SIGO(EN,LH,2) + SIGO(EN,LH,3))/3.0

DO K=1,3
    SDO(K) = SIGO(EN,LH,K) - SM
END DO
SDO(4) = SIGO(EN,LH,4) ! off diagonal

*.....Effective stress SEFM (at mid point in time)
DO K=1,4
    SIGM(K) = (SIG(EN,LH,K) + SIGO(EN,LH,K))/2.0
END DO
SM = (SIGM(1) + SIGM(2) + SIGM(3))/3.0

DO K=1,3
    SD(K) = SIGM(K) - SM
END DO
SD(4) = SIGM(4) ! off diagonal
SUM = 0.0
DO K=1,3
    SUM = SUM + SD(K)*SD(K)
END DO
SUM = SUM + 2.0*SD(4)*SD(4)

TE1 = SEFM(EN,LH) ! last iteration (time step midp
SEFM(EN,LH) = DSQRT(1.5*SUM)
TE2 = SEFM(EN,LH) ! current iteration (time step m
ERT = DABS(TE2 - TE1) ! absolute error
ERRELT = 1.0E10 ! relative error default if TE2=
IF(TE2 .NE. 0.0) THEN
    ERRELT = ERT/TE2 ! relative error
ENDIF

*.....Drag stress YYM (at mid point in time)
YMM = (YYO(EN,LH) + YY1(EN,LH))/2.0

*.....Incremental viscoplastic strain DEVP (at current time step,
* note that YYM and TE2 are at midpoint in time)
IF (TE2 .EQ. 0.0) THEN
    BK1(EN,LH) = 0.0
    DO K=1,4
        DEVP(K) = 0.0
    END
ELSE
    BK1(EN,LH) = DT*GM*(TE2/YYM)**MP ! at current time
    DO K=1,4
        DEVP(K) = BK1(EN,LH)*1.5*SD(K)/TE2
END IF
END DO
ENDIF
BKM = DABS((BK0(EN,LH) + BK1(EN,LH))/2.0) ! at time midp

*......Incremental viscous stress DSV (at current time step)
DO K=1,4
DSV(EN,LH,K) = 2.0*G1*DEVP(K)
END DO

*......Drag stress YY1 (at current time step)
YY1(EN,LH) = (YY0(EN,LH)*((1.0 - BKM*0.5*A1) + BKM*Bl)/
          (1.0 + BKM*0.5*A1))

*......Engineering stress STRESS

* Find total force across the cylinder, SUMM and then divide
  by the total area (note that PIE is left out in both
  area calculations
IF(EN .LE. TEX) THEN ! going across 1 row of elements
  IF(LG .EQ. 1) THEN
    SUMM = SUMM + (RE4**2 - REL1**2)*SIG(EN,1,1)
  ELSEIF(LG .EQ. 2) THEN
    IF(LH .EQ. 3) SUMM = SUMM +
      (RE4**2 - REL2**2)*SIG(EN,3,1)
    ELSE
      IF(LH .EQ. 4) SUMM = SUMM +
        (REL2**2 - REL1**2)*SIG(EN,4,1)
      ELSE
        SUMM = SUMM +
          (RE4**2 - REL1**2)*SIG(EN,7,1)
  ELSE
    IF(LH .EQ. 3) SUMM = SUMM +
      (RE4**2 - REM1**2)*SIG(EN,3,1)
    ELSE IF(LH .EQ. 4) SUMM = SUMM +
      (REM2**2 - REL1**2)*SIG(EN,4,1)
    ELSE IF(LH .EQ. 3) SUMM = SUMM +
      (RE3**2 - REL1**2)*SIG(EN,7,1)
    ELSE IF(LH .EQ. 4) SUMM = SUMM +
      (RE2**2 - REL1**2)*SIG(EN,7,1)
    ELSE
      IF(LH .EQ. 3) SUMM = SUMM +
        (RE3**2 - REL1**2)*SIG(EN,7,1)
    ENDIF
  ENDIF
  ELSE IF(EN .LE. TEX .AND. LH .EQ. LINT) THEN
    STRESS = SUMM/RRT**2
ENDIF

*......Maximum error
ER1 = DMAX1(ERV,ERT)
ERREL1 = DMAX1(ERRELV,ERRELT)
ERO = ERG(IT)
ERRELO = ERRELG(IT)
ERG(IT) = DMAX1(ERO,ER1)
ERRELG(IT) = DMAX1(ERRELO,ERREL1)

1000 END DO
1100 END DO

*......Check accuracy
*   terminate when ok or # of IT = JTERM
ERL = ERG(IT)
ERRELL = ERRELG(IT)
IF(IT .EQ. JTERM) THEN
  WRITE(6,*) 'TOO MANY ITERATIONS, IT=' ,IT
  WRITE(6,*) 'AT LOAD STEP, IL = ',IL
  WRITE(16,*) 'TOO MANY ITERATIONS, IT=' ,IT
  WRITE(16,*) 'AT LOAD STEP, IL = ',IL
  IF(IQ .EQ. IP .OR. IL .EQ. 1) THEN
    WRITE(36,*) 'TOO MANY ITERATIONS, IT=' ,IT
    WRITE(36,*) 'AT LOAD STEP, IL = ',IL
  ENDIF
ENDIF

ENDIF
IF(ERL .LE. ERC .OR. ERRELL .LE. ERRELC) THEN
  WRITE(16,*) 'CONVERGENCE, IT=' ,IT
  IF(IQ .EQ. IP .OR. IL .EQ. 1) THEN
    WRITE(36,*) 'CONVERGENCE, IT=' ,IT
  ENDIF
ENDIF
END

*....Print out results
IF(IT.EQ.JTERM) THEN
  WRITE(26,*),'ENGINEERING STRESS & STRAIN
  DO I=1,ETOTAL
    IF(TYPE(I).EQ.0) THEN
      OUT = SEF(ENUM(I),GP(I))/NORM(IMAT(ENUM(I)))
    ELSEIF(TYPE(I).GT.0 .AND. TYPE(I).LT.5) THEN
      OUT = SIG(ENUM(I),GP(I),TYPE(I))/NORM(IMAT(ENUM(I)))
    ELSEIF(TYPE(I).EQ.5) THEN
      OUT = YY1(ENUM(I),GP(I))/NORM(IMAT(ENUM(I)))
    ELSE
      WRITE(46+I,*),STRAIN,BK1(ENUM(I),GP(I))
    ENDIF
  END
END

IF(IP.EQ.IP.OR.IL.EQ.1) THEN
  CALL PLOT
  WRITE(16,620)
  WRITE(16,630)(ERG(K),K=1,JTERM)
  WRITE(16,632)
  WRITE(16,635)(ERRELG(K),K=1,JTERM)
  WRITE(16,640)
  WRITE(16,650) (M(DIS(I,M),I=1,NDF),M=1,NP)
  WRITE(16,660)
  DO EN=1,NE
    WRITE(16,670) EN
    DO LH=1,LINT
      WRITE(16,680) LH,(SIG(EN,LH,K),K=1,4)
      WRITE(16,660) LH,(SEF(EN,LH),YY)
    END
  END
END

DO EN=1,NE
  WRITE(16,685)
  DO EN=1,NE
    WRITE(16,670) EN
    DO LH=1,LINT
      WRITE(16,687) LH,(EPST(EN,LH,K),K=1,4)
      WRITE(16,667) LH,(EVF(EN,LH))
    END
  END
END

ENGINEERING STRESS in Z DIR = ',STRESS
ENGINEERING STRAIN in Z DIR = ',STRAIN

IQ = 0

*....Update the values of the stresses and strains
DO EN=1,NE
  DO I=1,LINT
    YYO(EN,I) = YY1(EN,I)
    BKO(EN,I) = BK1(EN,I)
    EPSO(EN,I,K) = EPST(EN,I,K)
    SIGO(EN,I,K) = SIG(EN,I,K)
  END
END

END
ENDIF

*........Format

FORMAT(//4X,2HEN,3X,2HPT,6X,7HRR-CORD,8X,7HZZ-CORD)

FORMAT(2I5,2E15.5)

FORMAT(//5X,'ABSOLUTE ERROR PROPAGATION HISTORY')

FORMAT(5X,10E12.5)

FORMAT(//5X,'RELATIVE ERROR PROPAGATION HISTORY')

FORMAT(5X,10E12.5)

FORMAT(//5X,'INCREMENTAL DISPLACEMENT')

FORMAT(I5,2E15.6,1X,I5,2E15.6,1X,I5,2E15.6)

FORMAT(//2X,2HPT,2X,9HZZ-STRESS

+ 4X,9HRR-STRESS,4X,9HTH-STRESS,4X,9HRZ-STRESS

+ 4X,9HEF-STRESS,4X,9HYD-STRESS,4X,9HEF-STNRAT)

FORMAT(10X,'ELEMENT #',I3)

FORMAT(2X,I2,7E13.5)

FORMAT(//2X,2HPT,2X,9HZZ-STRAIN

+ 4X,9HRR-STRAIN,4X,9HTH-STRAIN,4X,9HRZ-STRAIN

+ 4X,14HEF-STRAIN RATE)

FORMAT(2X,I2,5E13.5)

FORMAT(/10X,'ENGINEERING STRESS=’,E12.5,10X,'PRESSURE=’,F6.1)

RETURN

END
Subroutine PLOT

0001  *
0002 SUBROUTINE PLOT
0003 IMPLICIT REAL*8(A-H,O-Z)
0004 INCLUDE 'COMMON.FOR/LIST'
0005 1 * MAX - Maximum value for variable
0006 1 INTEGER JTMAX,MATMAX,NEMAX,NMAX,NBMAX,NLDMAX
0007 1 INTEGER EN,TEX,BCFLAG,KFLAG,ETMAX
0008 1 PARAMETER (JTMAX=20,MATMAX=5,NEMAX=200,NMAX=400, NBMAX=100)
0009 1 PARAMETER (NLDMAX=600,ETMAX=7)
0010 1 INTEGER TIMES(NLDMAX)
0011 1 INTEGER TYPE(ETMAX),GP(ETMAX),ENUM(ETMAX),ETOTAL
0012 1 REAL*8 LENGTH,DDELTA,NORM(MATMAX)
0013 1 REAL*8 DIS(2,NMAX),DELTAT(NLDMAX),DELTAU(NLDMAX)
0014 1 REAL*8 NSTRESS(NNMAX),SKT(2*NMAX,NMAX)
0015 1 COMMON TYPE,GP,ENUM,ETOTAL
0016 1 COMMON NP,NE,NB,NLD,NDF,NMAT,NSZF,NBAND,KFLAG
0017 1 COMMON IL,IT,IP,IQ,COUNT(NNMAX),IS,TEX,BCFLAG
0018 1 COMMON PIE,ERC,ERREL,CRT,DT,JTERM,LD
0019 1 COMMON ORT(MATMAX,10),CORD(NNMAX,2),NOP(NEMAX,8)
0020 1 COMMON IMAT(NEMAX),ISNE(NMAX),MBF(MATMAX)
0021 1 COMMON NBC(NMAX),NFIX(NNMAX),UBC(NMAX,2)
0022 1 COMMON SK(2*NMAX,NMAX),SKS(2*NMAX,NMAX)
0023 1 COMMON RL(2*NMAX),RL(2*NMAX),UL(2*NMAX)
0024 1 COMMON LENGTH,DDELTA,STRESS,STRAIN,NORM
0025 1 COMMON YYO(NMAX,9),YY1(NMAX,9),BK1(NMAX,9),BK1(NMAX,9)
0026 1 COMMON SIG(NMAX,9,4),SIGO(NMAX,9,4)
0027 1 COMMON EVF(NMAX,9),SEF(NMAX,9),SEFM(NMAX,9)
0028 1 COMMON EPST(NMAX,9,4),EPSO(NMAX,9,4)
0029 1 COMMON DSV(NMAX,9,4),ESTIM(16,16)
0030 1 COMMON ERC(JTMAX),ERRELG(JTMAX)
0031 1 * This routine assumes there are four noded elements.
0032 1 .......... Variables
0033 1 COUNT(NN) - Number of stress placed at a node
0034 1 EN - element number
0035 1 I - Counter for output file number
0036 1 IL - Load step
0037 1 IS - Flag for type of nodal stress to be plotted
0038 1 IS=0 -> SEF => IS=1 -> SIG(Z), IS=2 -> SIG(R)
0039 1 IS=3 -> SIG(SIG(T), IS=4 -> SIG(Z)
0040 1 ISNE(EN) - Number of nodes in element EN
0041 1 J - Counter of local node number
0042 1 K - Counter for Gauss points
0043 1 LG - Number of Gauss points in 1 dir
0044 1 NE - Total number of elements
0045 1 NEL - Number of nodes/element
0046 1 NN - Global node number
0047 1 NPF(NN,J) - Global node number of element EN and local node J
0048 1 NORM(N) - Normalization factor (Initial Drag value)
0049 1 NP - Total number of nodal points
0050 1 NSTRESS(NN) - Nodal stress
0051 1 SEF(EN,K) - Effective stress for element EN at Gauss point K
0052 1 SIG(EN,K,IS) - Stress for element EN at Gauss point K (IS is
0053 1 defined above
0054 1 STRAIN - Engineering strain in Z direction (%)
DO NN=1,NP
  COUNT(NN) = 0
  NSTRESS(NN) = 0.0
END DO
DO EN=1,NE
  NEL = ISNE(EN)
  IF(NEL .NE. 4) THEN
    WRITE(6,*), 'NODAL STRESS PLOTTER CAN ONLY WORK'
    WRITE(6,*), 'FOR 4 NODED ELEMENTS'
  ELSE
    NSTRESS(NN) = NSTRESS(NN) + SEF(EN,K)
  ENDIF
END
IF(LG .NE. 3) THEN
  DO J=1,NEL
    IF(LG .EQ. 1) K = 1 ! one gauss point
    IF(LG .EQ. 2) K = J ! total of 4 gauss points
    NN = NOP(EN,J)
    IF(IS .EQ. 0) THEN
      NSTRESS(NN) = NSTRESS(NN) +
      (SEF(EN,1)+SEF(EN,5)+SEF(EN,8)+SEF(EN,9))/4.0
    ELSE
      NSTRESS(NN) = NSTRESS(NN) + SIG(EN,K,IS)
    ENDIF
    COUNT(NN) = COUNT(NN) + 1
  END DO
ELSE
  DO J=1,NEL
    NN = NOP(EN,J)
    IF(IS .EQ. 0) THEN
      NSTRESS(NN) = NSTRESS(NN) +
      (SIG(EN,1,IS)+SIG(EN,5,IS)+SIG(EN,8,IS)+SIG(EN,9,IS))/4.0
    ELSE
      NSTRESS(NN) = NSTRESS(NN) +
      (SIG(EN,2,IS)+SIG(EN,5,IS)+SIG(EN,6,IS)+SIG(EN,9,IS))/4.0
    ENDIF
    COUNT(NN) = COUNT(NN) + 1
  END DO
ENDIF
WRITE(36,*), NP,' - NP'
WRITE(36,*), IL,' - LOAD STEP  STRESS TYPE = ',IS
WRITE(36,*), STRAIN,' = ENGINEERING STRAIN IN Z DIRECTION (%)'
DO NN=1,NP
  NSTRESS(NN) = NSTRESS(NN)/COUNT(NN)
END
0117    X = CORD(NN,1)
0118    Y = CORD(NN,2)
0119    OUT = NSTRESS(NN)/NORM(1)
0120    WRITE(36,*) NN,X,Y,OUT
0121    END DO
0122    *
0123    RETURN
0124    END
Example Input File

READ IP, IS, LG, NBAND, JTERM, ERC, ERREC
READ NP, NE, NB, KFLAG, BCFLAG, NLD, NDF, NMAT, I1
READ (N, (ORT(N,I), I=1,5), N=1, NMAT)
READ (N, (ORT(N,I), I=6,10), MPB(N), N=1, NMAT)
READ RRT, LENGTH
*nodal input
READ FLAG
DO WHILE (FLAG .NE. -1) -> end of nodal input
  IF(FLAG .EQ. 1) THEN rectangular linear nodal input
    READ N1, X1, Y1, X2, Y2, NXT, NYT
  ELSEIF(FLAG .EQ. 2) THEN rectangular quadratic nodal input
    READ N1, X1, Y1, X2, Y2, NXT, NYT
  ELSEIF(FLAG .EQ. 3) THEN input for 1 row or column
    READ N1, X1, Y1, X2, Y2, NT, NINC
  ELSEIF(FLAG .EQ. 4) THEN individual node input
    READ NN, X1, Y1
ENDIF
READ FLAG
END DO
*element input
READ FLAG
DO WHILE (FLAG .NE. -1) -> end of element input
  IF(FLAG .EQ. 1) THEN simple 4 node element input
    READ El, NEL, MAT, (GNN(I), I=1, NEL), XINC, YINC, TEX, TEY
  ENDIF
READ FLAG
END DO
*force b.c.
IF(II .EQ. 1 .OR. BCFLAG .NE. 1) THEN
  READ DT
  IF(II .EQ. 1 .OR. BCFLAG .NE. 1) THEN
    READ FLAG
    DO WHILE(FLAG .NE. -1) -> end of force input
      READ NQ, (R(K), K=1, NDF)
    END DO
    READ FLAG
  ENDIF
  READ DELTAL
  READ (NBC(I), NFIX(I), (UBC(I,K), K=1,2), I=1, NB)
  ELSEIF(II .EQ. 2 .AND. BCFLAG .EQ. 1) THEN
    READ BINC
    DO K=1, BINC
      READ TIMES(K), DELTAT(K), DELTAU(K)
    END DO
  ENDIF
END DO

130
Appendix D

PLOTTING PROGRAMS

The three programs listed here are used for plotting the numerical data. The first program, LPlot, calls the NCAR plotting routines to produce line plots. The program is capable of plotting several y variables as a function of one x variable. The second program, CPlot, calls the NCAR plotting routines to create nodal contours. Unfortunately, the NCAR contour plotting routines called by CPlot do not display enough decimal digits for clear representation of the normalized stress contours. The third program, PATPlot, creates an imitation PATRAN neutral file to plot nodal contours in PATRAN. Fortunately, PATPlot does display enough decimal digits for clear representation of the normalized stress contours.
Program LPLOT

* LPLOT
* plots stresses versus strains using NCAR
* plotting routine.

PROGRAM LPLOT
INTEGER MAX, RES
PARAMETER (MAX=800)
REAL X(MAX,9), Y(MAX,9), YMAX, YMIN, XMAX, XMIN
INTEGER NLD, MANY, I, FLAG, POSX, POSY
CHARACTER*11 FILE(9)
CHARACTER*4
XLAB, YLAB, TITLE

* For execution and linking first type NCAR
then to link type GKS LINK CPLOT

CALL GOPKS(6)
CALL GOPWKd, 2, 1)
CALL GACWK(1)

* WRITE(6,*) 'HOW MANY CURVES, 9 IS MAX ?'
READ(*,*) MANY
IF(MANY .GT. 9) THEN
WRITE(6,*) 'TOO MANY CURVES, 9 IS MAX'
CALL GDAWKd)
CALL GCLWKd)
CALL GCLKS
STOP

END

DO I=1, MANY
WRITE(6,600) I
READ(*,*) FILE(I)
END DO

* Default axis labels
XLAB = 'COMPRESSIVE ENGINEERING STRAIN (%)'
YLAB = 'COMPRESSIVE ENGINEERING STRESS'
TITLE =

* Check for default plotting
WRITE(6,*) 'DO YOU WANT DEFAULT PLOTTING (YES=1, NO=2) ?'
READ(*,*) FLAG
IF(FLAG .NE. 1) THEN
WRITE(6,*).

* Check for title
WRITE(6,*) 'DO YOU WANT A TITLE',
+(YES=1, NO=2)'
READ(*,*) FLAG
IF(FLAG .EQ. 1) THEN
WRITE(6,*) 'ENTER TITLE IN QUOTES (39 CHARACTERS MAX)'
READ(*,*) TITLE
ENDIF

* Check for label names
WRITE(6,*) 'DO YOU WANT TO CHANGE X AXIS LABEL? ',
+(YES=1, NO=2)'
WRITE(6,*) 'DEFAULT IS ', XLAB
READ(*,*) FLAG
IF(FLAG .EQ. 1) THEN
WRITE(6,*) 'X LABEL IN QUOTES IS (39 CHARACTERS MAX) ?'
READ(*,*) XLAB
ENDIF

* WRITE(6,*) 'DO YOU WANT POSITIVE (ABSOLUTE) X QUANTITIES?',
+(YES=1, NO=2)'

132
READ(5,*) POSX
0062 * WRITE(6,*) 'DO YOU WANT TO CHANGE Y AXIS LABEL? ',
0063   '(YES=1, NO=2)'
0064 + WRITE(6,*) 'DEFAULT IS ' , YLAB
0065 READ(5,*) FLAG
0066 IF(FLAG .EQ. 1) THEN
0067   WRITE(6,*) 'Y LABEL IN QUOTES IS (39 CHARACTERS MAX) ?'
0068   READ(5,*) YLAB
0069 ENDIF
0070
0071 * WRITE(6,*) 'DO YOU WANT POSITIVE (ABSOLUTE) Y QUANTITIES?',
0072   '(YES=1, NO=2)'
0073 + READ(5,*) POSY
0074
0075 * * * Check for high & low Y values
0076 WRITE(6,*) 'DO YOU WANT TO INPUT MIN & MAX Y VALUES?',
0077 +   '(YES=1, NO=2)'
0078 READ(5,*) FLAG
0079 IF(FLAG .EQ. 1) THEN
0080   WRITE(6,*) 'YMAX = ?'
0081   READ(5,*) YMAX
0082   WRITE(6,*) 'YMIN = ?'
0083   READ(5,*) YMIN
0084   CALL AGSETF('Y/MAXIMUM.',YMAX)
0085   CALL AGSETF('Y/MINIMUM.',YMIN)
0086 ENDIF
0087 * * * Check for high & low X values
0088 WRITE(6,*) 'DO YOU WANT TO INPUT MIN & MAX X VALUES?',
0089 +   '(YES=1, NO=2)'
0090 READ(5,*) FLAG
0091 IF(FLAG .EQ. 1) THEN
0092   WRITE(6,*) 'XMAX = ?'
0093   READ(5,*) XMAX
0094   WRITE(6,*) 'XMIN = ?'
0095   READ(5,*) XMIN
0096   CALL AGSETF('X/MAXIMUM.',XMAX)
0097   CALL AGSETF('X/MINIMUM.',XMIN)
0098 ENDIF
0099 * * * Read nodal data
0100 DO I=1,MANY
0101 OPEN (UNIT=15, FILE=FILE(I), STATUS = 'UNKNOWN')
0102 READ(15,*) ! reads title
0103 READ(15,*) NLD
0104 WRITE(6,*) 'NLD = ', NLD
0105 IF(NLD .GT. 0) THEN
0106   WRITE(6,*) 'TOO MANY POINTS, > ', MAX
0107   STOP
0108 CALL GDAXK(1)
0109 CALL GCLKS
0110 CALL GCLKS
0111 ENDIF
0112 DO K=1,NLD
0113 READ(15,*) X(K,I), Y(K,I)
0114 IF(POSX .EQ. 1) X(K,I) = ABS(X(K,I))
0115 IF(POSY .EQ. 1) Y(K,I) = ABS(Y(K,I))
0116 WRITE(6,*) X(K,I), Y(K,I)
0117 END DO
0118 END DO
0119 CLOSE (UNIT=15)
0120 END DO
0121   *         XLAB = XLAB//'$'
0122   *         YLAB = YLAB//'$'
0123   *         TITLE = TITLE//'$'
0124   CALL ANOTAT(XLAB,YLAB,0.,0.,0.,0.)
0125   CALL AGSETI('DASH/SELECTOR.',-1)
0126   CALL EZMXY(X,Y,MAX,MANY,NLD,TITLE)
0127   CALL FRAME
0128   *
0129   CALL GDAWK(1)
0130   CALL GCLWK(1)
0131   CALL GCLKS
0132   *
0133   600   FORMAT('0','ENTER ENTIRE FILE NAME IN QUOTES (11 LETTERS MAX) ','
0134   OF INPUT FILE ','11,/,' EXAMPLE ABCDEFG.ENG ')
0135   *
0136   STOP
0137   END
Program CPLOT

0001 *  CPLOT
0002 *  plots contours of nodal stresses using the
0003 *  NCAR plotting package.
0004 *
0005 PROGRAM CPLOT
0006 INTEGER MAX,RES
0007 PARAMETER (MAX=400, RES=40)
0008 REAL X(MAX),Y(MAX),VALUE(MAX),WK(13*MAX),SCALE
0009 REAL SCRARR(RES**2),ARRAY(2),CINC,CARRAY(30),BP
0010 INTEGER IWK(31*MAX),FLAG,FLAG1,FLAG2,NCL,NP,IL
0011 CHARACTER*7 FILE
0012 CHARACTER*39 TITLE
0013 *
0014 For execution and linking first type NCAR
0015 then to link type  GKS LINK CPLOT
0016 *
0017 CALL GOPKS(6)
0018 CALL GOPWK(1,2,1)
0019 CALL GACWK(1)
0020 *Read name of input file
0021 WRITE(6,"(A7)*)& input file. extension is .cnt by default'
0022 READ(5,*),FILE
0023 *Check if default plotting is wanted
0024 WRITE(6,*),'DO YOU WANT DEFAULT PLOTTING (YES=1, NO=2)' 
0025 READ(5,*),FLAG
0026 IF(FLAG .NE. 1) THEN
0027 *Check for title
0028 WRITE(6,*),'DO YOU WANT TITLE? (YES=1, NO=2)'
0029 READ(5,*),FLAG
0030 IF(FLAG .EQ. 1) THEN
0031 WRITE(6,*),'ENTER TITLE IN QUOTES (39 CHARACTERS MAX)'
0032 READ(5,*),TITLE
0033 CALL CONOP4('TLE-ON',TITLE,39,0)
0034 ENDIF
0035 *Check for scaling
0036 WRITE(6,*),'DO YOU WANT DATA SCALED ? (YES=1, NO=2) ' 
0037 READ(5,*),FLAG
0038 IF(FLAG .EQ. 1) THEN
0039 WRITE(6,*),'ENTER SCALING FACTOR'
0040 READ(5,*),SCALE
0041 CALL CONOP3('SCA-ON',SCALE,1)
0042 ENDIF
0043 *Check if high & low contour is to be specified
0044 WRITE(6,*),'DO YOU WANT TO SPECIFY HIGH & LOW CONTOURS ? (YES=1, NO=2) ' 
0047 READ(5,*),FLAG1
0048 IF(FLAG1 .EQ. 1) THEN
0049 WRITE(6,*),'ENTER HIGH AND LOW VALUE (SEPERATED BY COMMA)'
0050 READ(5,*),ARRAY(1),ARRAY(2)
0051 CALL CONOP3('CHL-ON',ARRAY,2)
0052 ENDIF
0053 *Check if contour increment is to be specified
0054 WRITE(6,*),'DO YOU WANT TO SPECIFY THE CONTOUR INCREMENT ? (YES=1, NO=2) ' 
0057 READ(5,*),FLAG2
0058 IF(FLAG2 .EQ. 1) THEN
0059 WRITE(6,*),'ENTER THE INCREMENT'
0060 READ(5,*),CINC
0061 CALL CONOP3('CIL-ON',CINC,1)
*........Check if contour values are to be specified  
*  (only if above were not)

IF(FLAG1 .NE. 1 .AND. FLAG2 .NE. 1) THEN

WRITE(6,*) 'DO YOU WANT TO SPECIFY THE CONTOUR VALUES  
+ (YES=1, NO=2)'
READ(5,*) FLAG

IF(FLAG .EQ. 1) THEN
WRITE(6,*) 'ENTER NUMBER OF CONTOURS < 30'
READ(5,*) NCL

IF(NCL .GE. 30) THEN
WRITE(6,*) 'TOO MANY CONTOUR LEVELS'
CALL GDAWK(1)
CALL GCLWK(1)
CALL GCLKS
STOP
ENDIF
DO I=1,NCL
WRITE(6,650) I
READ(5,*) CARRAY(I)
END DO
CALL CONOP3('CON=ON',CARRAY,NCL)
ENDIF
ENDIF

*........Check if a certain range should be dotted
WRITE(6,*) 'DO YOU WANT TO MAKE A CERTAIN RANGE DOTTED ?  
+ (YES=1, NO=2)'
READ(5,*) FLAG

IF(FLAG .EQ. 1) THEN
WRITE(6,*) 'ENTER THE VALUE ABOVE WHICH CONTOURS',  
+ ' WILL BE DOTTED'
READ(5,*) BP
CALL CONOP3('DBP=ON',BP,1)
CALL CONOP4('DAS=GTR','$$$',0,0)
ENDIF
ENDIF

*........Read nodal data
OPEN (UNIT=15,FILE=FILE//'CNR',STATUS='UNKNOWN')
READ(15,*) ! reads error message if any
READ(15,*) NP
WRITE(6,*) 'NP =',NP
IF(NP .GT. MAX) THEN
WRITE(6,*) 'TOO MANY POINTS, > ',MAX
CALL GDAWK(1)
CALL GCLWK(1)
CALL GCLKS
STOP
ENDIF
READ(15,*) IL ! load number
WRITE(6,*) 'IL =',IL
READ(15,*) ZSTRAIN ! strain in Z direction
WRITE(6,*) 'Z-STRAIN = ',ZSTRAIN
DO K=1,NP
READ(15,*) NN,X(K),Y(K),VALUE(K)
WRITE(6,*) NN,X(K),Y(K),VALUE(K)
END DO
CLOSE (UNIT=15)
CALL CONRAN(X,Y,VALUE,NP,WK,IKW,SCRARR)
CALL FRAME
0121  *        CALL GDANK(1)
0122  *        CALL GCLWK(1)
0123  *        CALL GCLKS
0125  *      FORMAT('0',',CONTOUR NUMBER ',I2,')
0126  650  *        STOP
0128  *        END
Program PATPLOT

0001  *  PATPLOT
0002  *  Used for plotting contours of nodal stresses
0003  *  using PATRAN. This program prepares the data
0004  *  for PATRAN. Data is plotted in PATRAN using the
0005  *  INTERFACE, NEUTRAL and RESULTS, EXTERNAL, NODAL
0006  *  options in PATRAN.
0007  *
0008  PROGRAM PATPLOT
0009  INTEGER MAXNODEMAX,RES
0010  PARAMETER (MAXNODE=1000, NCOLMAX=5)
0011  REAL X,Y,DEFMAX,VALUE(MAXNODE,NCOLMAX),Z
0012  INTEGER NP,NDMAX,NCOLMAX,NN(MAXNODE),NE,EN
0013  INTEGER G1,G2,G3,G4,JUNK,I
0014  CHARACTER*7 FILE1,FILE2(NCOLMAX),FILE3
0015  CHARACTER*80 TITLE1,TITLE2,TITLE3
0016  *
0017  TITLE1="'
0018  TITLE2="'
0019  TITLE3="'
0020  DEFMAX=1
0021  NDMIN=1
0022  Y=0.0 !plotting is assumed to be 2-d in xz plane
0023  *
0024  *  Read name of input file containing connectivity data
0025  *  This program assumes that QUAD elements are used.
0026  WRITE(6,*), 'ENTER NAME IN QUOTES (7 LETTERS MAX) OF ',
0027  + 'CONNECTIVITY FILE.'
0028  WRITE(6,*), 'EXTENSION IS .CON BY DEFAULT'
0029  WRITE(6,*), 'OUTPUT FILE FOR NASPAT IS *.NCN'
0030  *
0031  READ(5,*), FILE1
0032  *
0033  WRITE(6,*), 'HOW MANY FILES WITH CONTOURS?'
0034  READ(5,*), NCOL
0035  DO I=1,NCOL
0036      WRITE(6,*), 'ENTER NAME IN QUOTES (7 LETTERS MAX) OF ',
0037      + 'CONTOUR FILE.'
0038      WRITE(6,*), 'EXTENSION IS .CNT BY DEFAULT'
0039  READ(5,*), FILE2(I)
0040  READ(5,*), FILE3(I)
0041  END DO
0042  *
0043  *  The output file contains all the contour info in 1 file
0044  *  one collumn for each contour data set
0045  WRITE(6,*), 'OUTPUT FILE FOR NASPAT IS *.NVA'
0046  READ(5,*), FILE3
0047  WRITE(6,*), 'ENTER MAIN TITLE (IN QUOTES)'
0048  READ(5,*), TITLE1
0049  WRITE(6,*), 'ENTER SECOND TITLE (IN QUOTES)'
0050  READ(5,*), TITLE2
0051  WRITE(6,*), 'ENTER THIRD TITLE (IN QUOTES)'
0052  READ(5,*), TITLE3
0053  *
0054  *  Read connectivity data
0055  OPEN (UNIT=15,FILE=FILE1//'.CON',STATUS='UNKNOWN')
0056  OPEN (UNIT=36,FILE=FILE3//'.NCN',STATUS='UNKNOWN',
0057     + CARRIAGECONTROL='LIST')
0058  READ(15,*), NP !number of grid points
0059  WRITE(6,*), 'NP =',NP
DO K=1,NP
READ(15,*), NN(1),X,Z
WRITE(6,*), NN(1),X,Z
WRITE(36,600) NN(1),0,X,Y,Z ! y is zero from above
END DO
READ(15,*), ! reads header
READ(15,*), NE ! number of elements
DO K=1,NE
READ(15,*), EN,G1,G2,G3,G4, JUNK, JUNK
* junk is data not needed by PATRAN
READ(15,*), ! reads blank line
WRITE(6,*), EN,G1,G2,G3,G4
WRITE(36,610) EN,1,G1,G2,G3,G4 ! write data for PATRAN
END DO
CLOSE (UNIT=15)
CLOSE (UNIT=36)

* read nodal data
DO I=1,NCOL ! for each contour set
OPEN (UNIT=15,FILE=FILE2(I)///'CNT',STATUS='UNKNOWN')
READ(15,*), N
IF(N .NE. NP) THEN
WRITE(6,*), 'INPUT ERROR, .CON FILE AND .CNT FILE'
WRITE(6,*), 'HAVE DIFFERENT NUMBER OF NODES'
STOP
ENDIF
READ(15,*), ! read header info
READ(15,*), ! read header info
DO J=1,NP
READ(15,*), NN(J),X,Y,VALUE(J,I)
END DO
CLOSE (UNIT=15)

OPEN (UNIT=36,FILE=FILE3///'NVA',STATUS='UNKNOWN',
+ CARRIAGECONTROL='LIST')
WRITE(36,650) TITLE1
WRITE(36,660) NP,MAXNODE,DEFMAX,NDMAX,NCOL
WRITE(36,650) TITLE2
WRITE(36,650) TITLE3
DO J=1,NP
WRITE(36,670) J,(VALUE(J,I),I=1,NCOL)
END DO

STOP
END