SANDIA REPORT

SAND2018-3236
Unlimited Release
Printed March 28, 2018

Sierra/SolidMechanics 4.48 User's Guide: Addendum for Shock Capabilities

SIERRA Solid Mechanics Team
Computational Solid Mechanics and Structural Dynamics Department
Engineering Sciences Center

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology and Engineering Solutions of Sandia, LLC.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone:	(865) 576-8401
Facsimile:	(865) 576-5728
E-Mail:	reports@adonis.osti.gov
Online ordering:	http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone:	(800) 553-6847
Facsimile:	(703) 605-6900
E-Mail:	orders@ntis.fedworld.gov
Online ordering:	http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2018-3236
Unlimited Release
Printed: March 28, 2018

Sierra/SolidMechanics 4.48 User's Guide: Addendum for Shock Capabilities

SIERRA Solid Mechanics Team
Computational Solid Mechanics and Structural Dynamics Department
Engineering Sciences Center
Sandia National Laboratories
Box 5800
Albuquerque, NM 87185-0380

Abstract

This is an addendum to the Sierra/SolidMechanics 4.48 User's Guide that documents additional capabilities available only in alternate versions of the Sierra/SolidMechanics (Sierra/SM) code. These alternate versions are enhanced to provide capabilities that are regulated under the U.S. Department of State's International Traffic in Arms Regulations (ITAR) export-control rules. The ITAR regulated codes are only distributed to entities that comply with the ITAR export-control requirements. The ITAR enhancements to Sierra/SM include material models with an energy-dependent pressure response (appropriate for very large deformations and strain rates) and capabilities for blast modeling. Since this is an addendum to the standard Sierra/SM user's guide, please refer to that document first for general descriptions of code capability and use.

Acknowledgments

This document is the result of the collective effort of a number of individuals. This document was originally written primarily by Arne Gullerud, John Carpenter, and Bill Scherzinger. The current development team responsible for the Sierra/SolidMechanics codes includes: Nathan K. Crane, Gabriel J. de Frias, San Le, David J. Littlewood, Mark T. Merewether, Matthew D. Mosby, Kendall H. Pierson, Julia A. Plews, Vicki L. Porter, Timothy R. Shelton, Jesse D. Thomas, Michael R. Tupek, Michael G. Veilleux, and Patrick G. Xavier. This document is maintained by this team.

Presto_ITAR 4.48 Release Notes

Following is a list of new features and syntax changes made since the 4.40 release.

The Zapotec documentation was updated for the 4.42 release, see Section 7.

The legacy Zapotec documentation has been updated for Sierra/SM and is now available as part of the Sierra documentation package for ITAR codes. This updated reference contains most of the information about Zapotec and how to use it. Also, an examples manual has been made for Zapotec that can provide a great starting point for a new Zapotec analysis. This is distributed as part of the Sierra documentation package for ITAR codes. See Section 7 for more details.

Extensive fixes have been made for Zapotec for this release. Fixes include:

- Improved input error checking and error messages
- Fixes to time stepping mechanism to prevent excessively small and excessively large time steps
- Improvements to the force mapping algorithm that substantially increase force accuracy at mesh corners
- Inclusion of CTH as an externally built library linked into Sierra, which ensures that CTH is correctly built
- Fixes for errors in the Sierra/SM linkage routines
- Fixes to support parallel runs up to 5000 processors
- Fixes for AMR support in Zapotec, especially for explosives

Following is a list of new features and syntax changes made since the 4.42 release.

There have been no new major developments since the 4.42 release.

Following is a list of new features and syntax changes made since the 4.44 release.

There have been no new major developments since the 4.44 release.

Chapter 1

Introduction

This document is an addendum to the Sierra/SolidMechanics 4.48 User's Guide. The standard Sierra/SolidMechanics (Sierra/SM) user's guide describes the general input structure and most of the commands that are permissible in Sierra/SM. This addendum describes additional capabilities that are available only in ITAR versions of Sierra/SM, i.e. enhanced versions of Sierra/SM that include additional capabilities that make them regulated under the U.S. Department of State's International Traffic in Arms Regulations (ITAR) rules. The enhanced codes are only distributed to entities that comply with the ITAR export-control requirements.

The capabilities in the enhanced Sierra/SM codes that have been indicated as being ITAR restricted are, in general, only applicable to explicit transient dynamics. These capabilities deal with material response under very high rates of loading and/or deformation or with blast modeling. Most of the material response capabilities have been adopted from other export-controlled codes, such as EPIC and CTH. Some material capabilities, such as the ideal gas material model, are not explicitly export controlled but are similar in structure to the export-controlled capabilities. These capabilities are only available in the ITAR-controlled version of Sierra/SM and are thus documented here.

Most of the documentation of how to use the Sierra solid mechanics codes is not included in this document. For that information, refer to the standard user's guides for Sierra/SolidMechanics [1].

1.1 Document Overview

This document describes the ITAR restricted capabilities within the Sierra Solid Mechanics codes. Highlights of the document contents are as follows:

- Chapter 2 presents material models that are included in the Presto_ITAR version of Sierra/SM. These include materials from the Modular Material Models (MMM) interface (from EPIC) and CTH, as well as native implementations. These materials models have a pressure response that is dependent on the energy within the element. This chapter also describes how energy deposition is enabled within the code.

- Chapter 3 describes element features that support the energy dependent material models, such as internal iterations to resolve nonlinear energy-pressure relations.

- Chapter 4 describes a two-way code coupling ability known as Fortissimo.

- Chapter 5 describes a specialized boundary condition based on the ConWep code to simulate the blast pressure from an explosive.

- Chapter 6 presents the variables available for output from the Sierra/SM ITAR material models.

- Chapter 7 describes a two-way code coupling ability known as Zapotec.

1.2 Running The Code

There are three Sierra/SM ITAR codes: "Presto_ITAR," "Fortissimo," and "Zapotec." The command to run any of these executables is essentially the same. For example, the command to run a basic Presto_ITAR analysis is:

```
sierra presto_itar -i sierra_input.i
```

Note that the capabilities defined in this addendum are only available when running the relevant executable (`presto_itar`, `fortissimo`, or `zapotec`) and are not available when running the basic `adagio` executable. However, generally all analyses that run with the `adagio` executable will also run with the `presto_itar` executable.

The sierra command also optionally takes many more options to specify number of processors, queues to use, output log file names, etc. See the sierra command documentation for a full description of capabilities.

1.3 Obtaining Support

Support for all SIERRA Mechanics codes, including Sierra/SM ITAR, can be obtained by contacting the SIERRA Mechanics user support hotline by email at sierra-help@sandia.gov.

References

[1] Sierra/SolidMechanics Team. Sierra/SolidMechanics 4.46 User's Guide. Technical Report SAND2017-9759, Sandia National Laboratories, Albuquerque, NM, 2017.

Chapter 2

Materials

This chapter describes material models that exist in Presto_ITAR but not in standard Presto_ITAR. In general, all material models that have an explicit pressure dependence on energy are available only in the ITAR export-controlled version of the code. The material models documented in this manual are broken into three groups:

- Modular Material Models (MMM): The MMM models are a select set of models extracted from the EPIC code and put into a common interface. They include a range of models that are widely used in modeling materials in the mild shock regime in a Lagrangian framework. See Reference [1] for more information.
- CTH models: These are material models that exist within the CTH code base. This does not include all of the models in CTH; only those that directly compute a stress are included. These models include the ability to reference SESAME equation-of-state models to handle some level of phase change under very large deformations.
- Standard Equation-of-State (EOS) Models: These are implementations of standard EOS models within the LAME material model package [2].

All material models documented here are only available in `presto_itar` and not in the standard `adagio` executable. Only the commands specific to these models are provided here. General information about conventions and commands for usage of material models is provided in the Sierra/SolidMechanics 4.48 User's Guide.

Additional information in this section describes how to deposit energy into the elements. Only energy-dependent materials such as those described in this document have the capability to respond to deposited energy.

2.1 Modular Material Model (MMM) Specifications

A set of material models known as Modular Material Model (MMM) subroutines has been developed to be portable across a variety of codes, as described in [1]. These models have been made available in Presto_ITAR.

The following MMM models are provided in Presto_ITAR:

- Bodner-Partom strength model with Mie-Gruneisen EOS
- Holmquist-Johnson-Cook concrete model
- Hull concrete model
- Johnson-Cook strength model with Mie-Gruneisen EOS and Johnson-Cook failure model
- Johnson-Holmquist ceramic model
- Johnson-Holmquist-Beissel ceramic model
- Mechanical Threshold Stress (MTS) strength model with Mie-Gruneisen EOS
- Mechanical Threshold Stress (MTS) strength model with Mie-Gruneisen EOS and the TEPLA continuum level damage model
- Zerilli-Armstrong strength model for BCC metals with Mie-Gruneisen EOS
- Zerilli-Armstrong strength model for FCC metals with Mie-Gruneisen EOS

The inputs for these models are documented in the subsections below. A full description of the theory and implementation of the models is available in Reference [1].

2.1.1 Bodner-Partom strength model with Mie-Gruneisen EOS

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL BPSTRESS_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    ABS ZERO TEMP = <real>ABS_ZERO_TEMP
    INIT TEMPERATURE = <real>INIT_TEMPERATURE
    SPECIFIC HEAT = <real>SPECIFIC_HEAT
    INIT STATE VAR Z0 = <real>INIT_STATE_VAR_Z0
    MAX RATE D0 = <real>MAX_RATE_D0
    MAX STATE VAR Z1 = <real>MAX_STATE_VAR_Z1
    STRAIN HARD PAR ALPHA = <real>STRAIN_HARD_PAR_ALPHA
    STRAIN HARD PAR M0 = <real>STRAIN_HARD_PAR_M0
    STRAIN HARD PAR M1 = <real>STRAIN_HARD_PAR_M1
    STRAIN RATE EXP N0 = <real>STRAIN_RATE_EXP_N0
    THERM SOFT PAR N1 = <real>THERM_SOFT_PAR_N1
    GRUN COEF = <real>GRUN_COEF
    MIEGRU COEF K2 = <real>MIEGRU_COEF_K2
    MIEGRU COEF K3 = <real>MIEGRU_COEF_K3
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [PARAMETERS FOR MODEL BPSTRESS_MMM]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Bodner-Partom Stress model with a Mie-Gruneisen EOS. The expression for the yield function of this model is:

$$\sigma = Z\left(-\left(\frac{2n}{n+1}\right) ln\left(\frac{\sqrt{3}\dot{\epsilon}_p}{2D_0}\right)\right)^{\frac{1}{2n}} \tag{2.1}$$

where n, Z, and m are defined by

$$n = n_0 + n_1/T \tag{2.2}$$

$$Z = Z_1 - (Z_1 - Z_0)exp(m - m_0 - m_1)/\alpha - m_0 W_p \tag{2.3}$$

$$m = m_0 + m_1 exp(-\alpha W_p) \tag{2.4}$$

and where $\dot{\epsilon}_p$ is the equivalent plastic strain rate, D_0 is the maximum allowable equivalent plastic strain rate, T is the absolute temperature, and W_p is the plastic work per initial volume. Z_0, Z_1, n_0, n_1, m_0, m_1, α, and D_0 are all material constants.

The pressure response is described by a cubic Mie-Gruneisen model:

$$P = (K_1\mu + K_2\mu^2 + K_3\mu^3)\left(1 - \frac{\Gamma\mu}{2}\right) + \Gamma E_s(1 + \mu) \tag{2.5}$$

where $\mu = V_0/V - 1$, Γ is the Gruneisen coefficient, V_0 and V are the initial and current volumes, respectively, K_1 is the elastic bulk modulus and K_2, and K_3 are material constants.

The Bodner-Partom command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL BPSTRESS_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL BPSTRESS_MMM]
```

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material is defined with the `YIELD STRESS` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - The temperature at absolute zero is defined with the `ABS ZERO TEMP` command line.
  - The initial temperature is defined with the `INIT TEMPERATURE` command line.
  - The specific heat is defined with the `SPECIFIC HEAT` command line.
  - The material parameters Z0, D0, Z1, ALPHA, M0, M1, N0, and N1 are defined with the corresponding command lines listed above.
  - The Gruneisen parameter Gamma is defined with the `GRUN COEF` command line.

- The K2 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K2` command line.
- The K3 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K3` command line.
- The maximum permitted tensile pressure is defined with the `MAX TENS PRESS` command line.

Output variables available for this model are listed in Table 6.1.

2.1.2 Holmquist-Johnson-Cook concrete model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL HJCCONCRETE_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    COMP STREN FC = <real>COMP_STREN_FC
    DAMAGE COEF D1 = <real>DAMAGE_COEF_D1
    DAMAGE EXP D2 = <real>DAMAGE_EXP_D2
    INIT SHEAR MODULUS = <real>INIT_SHEAR_MODULUS
    MAX STRESS = <real>MAX_STRESS
    MAX TENS PRESS T = <real>MAX_TENS_PRESS_T
    MIN FAIL STRAIN = <real>MIN_FAIL_STRAIN
    PCRUSH = <real>PCRUSH
    PLOCKI = <real>PLOCKI
    PRESS COEF K1 = <real>PRESS_COEF_K1
    PRESS COEF K2 = <real>PRESS_COEF_K2
    PRESS COEF K3 = <real>PRESS_COEF_K3
    PRESS HARD COEF B = <real>PRESS_HARD_COEF_B
    PRESS HARD EXP N = <real>PRESS_HARD_EXP_N
    STRAIN RATE COEF C = <real>STRAIN_RATE_COEF_C
    UCRUSH = <real>UCRUSH
    ULOCK = <real>ULOCK
    YIELD STRESS A = <real>YIELD_STRESS_A
   END [PARAMETERS FOR MODEL HJCCONCRETE_MMM]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Holmquist-Johnson-Cook concrete model. This model has a yield surface defined by:

$$\sigma = [A(1 - D) + BP^{*n}][1 + C\, ln\, \dot{\epsilon}^*] \tag{2.6}$$

where A,B, n, and C are material constants. Additionally, $\dot{\epsilon}^* = \dot{\epsilon}/\dot{\epsilon}_0$, where $\dot{\epsilon}$ is the total equivalent strain rate and $\dot{\epsilon}_0 = 1.0s^{-1}$. P^* is the pressure normalized by f_c' (the uniaxial compressive strength at $\dot{\epsilon}^* = 1.0$). The value of σ can be limited to σ_{max} if specified in the input file. D is the damage term, which is computed through the equation

$$D = \sum \frac{\Delta\epsilon_p + \Delta\mu_p}{\epsilon_p^f + \mu_p^f} \tag{2.7}$$

where ϵ denotes equivalent plastic strain, μ denotes plastic volumetric strain, ^{f} indicates values at failure, and Δ indicates change over a step. The combined failure strain $\epsilon_p^f + \mu_p^f$ is set to $D_1(P^* + T^*)^{D_2}$, where D_1 and D_2 are material constants, P^* was defined previously, and $T^* = T/f_c'$, where T is the maximum permitted tensile pressure.

The compressive pressure response is dependent upon the values of volumetric crush (μ_{crush}) and lock (μ_{lock}), where $\mu = V_0/V - 1$, and V and V_0 are the current and initial volumes. At strains below μ_{crush}, the bulk modulus is constant and equal to P_{crush}/μ_{crush}. At volume strains above μ_{lock}, the material is considered to be fully compressed with no voids, and is described as $P = K_1\bar{\mu} + K_2\bar{\mu}^2 + K_3\bar{\mu}^3$ where $\bar{\mu} = (\mu - \mu_{lock})/(1 + \mu_{lock})$. Between μ_{crush} and μ_{lock}, voids are crushed out of the material, and a linear fit is made between the states at μ_{crush} and μ_{lock}.

The tensile pressure response is defined as $P = K\mu$ before μ_{crush}, $P = K_1\bar{\mu}$ after μ_{plock} (note this is different than μ_{lock}), and is linearly interpolated between the states at μ_{crush} and μ_{plock} when between these values. A limit is placed on the tensile pressure by the expression $P_{max} = T(1 - D)$, using the T described previously.

The command block for this model starts with the input line:

```
BEGIN PARAMETERS FOR MODEL HJCCONCRETE_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL HJCCONRETE_MMM]
```

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material is defined with the `YIELD STRESS A` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - The line `COMP STREN FC` sets the value of f_c'.
  - The material constants D_1 and D_2 are used in the damage evolution equation.

- The initial shear modulus is set through the command `INIT SHEAR MODULUS`.
- The maximum permitted equivalent compressive stress is set by the command `MAX STRESS`.
- The maximum permitted tensile stress (T) is set by the command `MAX TEN PRESS T`.
- The minimum failure strain is set with the command `MIN FAIL STRAIN`.
- The pressure and volumetric strain at crush are set with the commands `PCRUSH` and `UCRUSH`, respectively.
- The pressure and volumetric strain at volumetric locking (fully dense material) are set with the commands `PLOCKI` and `ULOCK`, respectively.
- The fully dense compressive pressure constants K_1, K_2, and K_3 are specified through the related command lines.
- The yield function material constants B, n, and C are specified through the related command lines.

Output variables available for this model are listed in Table 6.4. More information about this model is available in Reference [1].

2.1.3 Hull Concrete Model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL HULLCONCRETE_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    KLOCK = <real>KLOCK
    MAX STRESS = <real>MAX_STRESS
    MAX TENS PRESS T = <real>MAX_TENS_PRESS_T
    PCRUSH = <real>PCRUSH
    PRESS COEF K1 = <real>PRESS_COEF_K1
    PRESS COEF K2 = <real>PRESS_COEF_K2
    PRESS COEF K3 = <real>PRESS_COEF_K3
    PRESS HARD COEF B = <real>PRESS_HARD_COEF_B
    STRAIN RATE COEF C = <real>STRAIN_RATE_COEF_C
    UCRUSH = <real>UCRUSH
    ULOCK = <real>ULOCK
    YIELD STRESS A = <real>YIELD_STRESS_A
  END [PARAMETERS FOR MODEL HULLCONCRETE_MMM ]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Hull concrete model. This model has a yield surface defined by:

$$\sigma = [A + BP][1 + C \ln \dot{\epsilon}^*] \tag{2.8}$$

where A,B, and C are material constants. Additionally, $\dot{\epsilon}^* = \dot{\epsilon}/\dot{\epsilon}_0$, where $\dot{\epsilon}$ is the total equivalent strain rate and $\dot{\epsilon}_0 = 1.0s^{-1}$. The value of σ can be limited to σ_{max} if specified in the input file.

The compressive pressure response is dependent upon the values of volumetric crush (μ_{crush}) and lock (μ_{lock}), where $\mu = V_0/V - 1$, and V and V_0 are the current and initial volumes. At strains below μ_{crush}, the bulk modulus is constant and equal to P_{crush}/μ_{crush}. Between μ_{crush} and μ_{lock}, $P = P_{crush} + K_1\bar{\mu} + K_2\bar{\mu}^2 + K_3\bar{\mu}^3$ where $\bar{\mu} = \mu - \mu_{crush}$. At volume strains above μ_{lock}, $P = K_{lock}(\mu - \mu_0)$ where μ_0 is the volumetric strain after unloading down to $P = 0$ from $\mu = \mu_{lock}$.

The command block for this model starts with the input line:

```
BEGIN PARAMETERS FOR MODEL HULLCONCRETE_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL HULLCONCRETE_MMM]
```

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material is defined with the `YIELD STRESS A` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - The maximum permitted equivalent compressive stress is set by the command `MAX STRESS`.
  - The maximum permitted tensile stress (T) is set by the command `MAX TEN PRESS T`.
  - The pressure and volumetric strain at crush are set with the commands `PCRUSH` and `UCRUSH`, respectively.
  - The volumetric strain at volumetric locking (fully dense material) is set with the command `ULOCK`.
  - The pressure constants K_1, K_2, and K_3 are specified through the related command lines.
  - The yield function material constants B and C are specified through the related command lines.

Output variables available for this model are listed in Table 6.5. More information about this model is available in Reference [1].

2.1.4 Johnson-Cook strength model with Mie-Gruneisen EOS and Johnson-Cook failure model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL JCSTRESS_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    ART VIS CL = <real>Linear_Artificial_Bulk_Viscosity
    ART VIS CQ = <real>Quadratic_Artificial_Bulk_Viscosity
    INIT TEMPERATURE = <real>INIT_TEMPERATURE
    MELT TEMPERATURE = <real>MELT_TEMPERATURE
    ROOM TEMPERATURE = <real>ROOM_TEMPERATURE
    SPECIFIC HEAT = <real>SPECIFIC_HEAT
    JCF MODEL =  NONE | ORIGINAL | MODIFIED
    JCF D1 = <real>JCF_D1
    JCF D2 = <real>JCF_D2
    JCF D3 = <real>JCF_D3
    JCF D4 = <real>JCF_D4
    JCF D5 = <real>JCF_D5
    JCF EFMIN = <real>JCF_EFMIN
    JCF KSTAR = <real>KSTAR
    JCF LAMBDA = <real>LAMBDA
    JCF LFAIL = <real>JCF_LFAIL
    JCF PFAIL = <real>JCF_PFAIL
    JCF WM = <real>JCF_WM
    JCF REFVOL = <real>JCF_REFVOL
    JCF ICSEED = <integer> JCF_ICSEED
    JCF ITSEED = <integer> JCF_ITSEED
    MAX STRESS = <real>MAX_STRESS
    PRESS HARD COEF = <real>PRESS_HARD_COEF
    STRAIN HARD COEF = <real>STRAIN_HARD_COEF
    STRAIN HARD EXP = <real>STRAIN_HARD_EXP
    STRAIN RATE COEF = <real>STRAIN_RATE_COEF
    STRAIN RATE MODEL =  LOG | POWER
    THERM SOFT EXP = <real>THERM_SOFT_EXP
    MIEGRU FORM =  CUBIC | USUP
    GRUN COEF = <real>GRUN_COEF
    MIEGRU COEF K2 = <real>MIEGRU_COEF_K2
    MIEGRU COEF K3 = <real>MIEGRU_COEF_K3
    MIEGRU CSBULK = <real>CSBULK
```

```
      MIEGRU SLOPE = <real>SLOPE
      MAX TENS PRESS = <real>MAX_TENS_PRESS
    END [PARAMETERS FOR MODEL JCSTRESS_MMM ]
  END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Johnson-Cook Stress model with a Mie-Gruneisen EOS and the Johnson-Cook failure model. This is a widely used material model, and fits for a range of materials can be found in the literature. Several options turn on and off slight modifications to the model, and the failure portion can be used or turned off. The failure model also includes an option to randomly perturb the failure strains for the model, permitting the adding of material non-heterogeneity into analyses.

The Johnson-Cook Stress model has a yield function described by

$$\sigma = [A + B\epsilon_p^n][1 + C\ ln\ \dot{\epsilon}^*][1 - T^{*m}] + \alpha P \tag{2.9}$$

Where ϵ_p is the equivalent plastic strain, $\dot{\epsilon}^* = \dot{\epsilon}/1.0\ sec^{-1}$, P is the hydrostatic pressure, $T^* = (T - T_{room})/(T_{melt} - T_{room})$ where T refers to temperature, and A, B, C, n, m, and α are material constants. The stress can be capped to user-specified maximum.

The strain rate dependence can also take on a power-law form, where the expression $[1 + C\ ln\ \dot{\epsilon}^*]$ is replaced with $[\dot{\epsilon}^{*C}]$.

The Johnson-Cook Stress model also has the capability to compute material failure. Once failed, the model provides resistance only to hydrostatic pressure. Material failure occurs when the damage D is greater than 1.0. Note that a value of D less than 1.0 has no effect on the computed stresses in the model. D accumulates according to the equation

$$D = \sum(\Delta\epsilon_p/\epsilon_p^f) \tag{2.10}$$

where $\Delta\epsilon_p$ is the increment of plastic strain over a time step, and ϵ_p^f is the failure strain. The failure strain is described by the expression

$$\epsilon_p^f = [D_1 + D_2 exp(D_3\sigma^*)][1 + D_4\ ln\ \dot{\epsilon}_p][1 + D_5 T^*] \tag{2.11}$$

where σ^* is the mean pressure divided by the von Mises equivalent stress, T^* is the normalized temperature described earlier, $\dot{\epsilon}_p$ is the plastic strain rate, and D_1 through D_5 are material constants. Note that the failure strain for a material point changes if the loading or temperature changes.

For high tensile stresses, the failure strain is handled differently. In the original J-C failure model, the failure strain is capped at ϵ_{min}^f once the stress reaches σ_{spall}^*, which is defined as the user-specified σ_{spall} normalized by the von Mises stress. The transition to this cap starts at a normalized tensile stress of $\sigma^* > 1.5$, at which point it varies linearly to the cap values. Alternatively, a modified version accumulates damage for tensile pressures as

$$D = \frac{\sum(\sigma^* - 1)^\lambda \Delta t}{K^*} \tag{2.12}$$

where λ and K^* are material constants. This is activated once the mean tensile pressure exceeds the threshold σ_{m0}.

Statistical variation of the failure parameters can also be added through this model. See below for the commands which activate this.

The command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL JCSTRESS_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL JCSTRESS_MMM]
```

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material, shown as A in the equations above, is defined with the `YIELD STRESS` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - Extra linear artificial bulk viscosity can be defined with the `ART VIS CL` this value should generally be set to zero.
  - Extra quadratic artificial bulk viscosity can be defined with the `ART VIS CQ` this value should generally be set to zero.
  - The initial temperature is defined with the `INIT TEMPERATURE` command line.
  - The room temperature is defined with the `ROOM TEMPERATURE` command line.
  - The melt temperature is defined with the `MELT TEMPERATURE` command line.
  - The specific heat is defined with the `SPECIFIC HEAT` command line.
  - The hardening constant B is specified with the command `STRAIN HARD COEF`
  - The hardening exponent n is specified with the command `STRAIN HARD EXP`

- The exponent on the temperature m is specified with the command `THERM SOFT EXP`
- The term α is specified with the command `PRESS HARD COEF`
- A limit on the yield stress can be specified using the `MAX STRESS` command line.
- The form of the rate dependence is chosen with the command `STRAIN RATE MODEL` – choose LOG for the traditional form, and POWER for the power law version. In both cases, the material parameter C which controls the rate effect is specified with the command line `STRAIN RATE COEF`.
- The type of failure model is defined with the `JCF MODEL` command line. If the value is NONE, then no failure model is used. The original version of the Johnson-Cook Failure model with it's original treatment of spall is chosen with the `ORIGINAL` keyword. The MODIFIED value chooses the modified version of the spall model.
- The Johnson-Cook failure model parameters D_1, D_2, D_3, D_4, and D_5 are defined with their corresponding commands, each of which begin with the `JCF` command word.
- The spall cap for the original Johnson-Cook Failure model is specified with the commands `JCF PFAIL` and `JCF EFMIN` for the spall stress (σ_{spall}) and minimum failure strain (ϵ^f_{min}), respectively.
- The spall behavior for the modified Johnson-Cook Failure model is specified with the commands `JCF KSTAR` and `JCF LAMBDA` for K^* and λ, respectively. The command `JCF PFAIL` specifies the threshold mean tensile pressure (σ_{m0}) after which the spall model is used for failure.
- The command `JCF LFAIL` controls whether the stress will be decayed if a damage > 1.0 is reached. If this value is set to zero, no failure will occur, though damage will still be computed. A value of 1 will cause the stress to go to zero once damage > 1.0.
- A Weibull modulus based variability is available through the JCSTRESS_MMM model. This capability is activated if the value given for the Weibull modulus using the command `JCF WM` is a value greater than zero. `JCF REFVOL` defines a representative element size, such as the average size of elements where failure is expected. The commands `JCF ICSEED` and `JCF ITSEED` serve as seeds for the random number generator.
- JCSTRESS_MMM permits the choice of two different implementations of the Mie-Gruneisen model. The command `MIEGRU FORM` chooses the version.
  - The Gruneisen parameter Gamma is defined with the `GRUN COEF` command line.
  - If `MIEGRU FORM` is chosen as `CUBIC`, then the cubic version of Mie-Gruneisen is chosen. The following commands are active:
    - The K_2 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K2` command line.
    - The K_3 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K3` command line.
  - If `MIEGRU FORM` is chosen as `USUP`, then the linear $U_s - U_p$ version of Mie-Gruneisen is chosen. The following commands are active:

  - The initial bulk sound speed is defined with the `MIEGRUN CSBULK` command line.
  - The slope of the $U_s - U_p$ relation (S) is defined with the `MIEGRUN SLOPE` command line.

- The maximum permitted tensile pressure is defined with the `MAX TENS PRESS` command line.

Output variables available for this model are listed in Table 6.8. More information about this model is available in Reference [1].

2.1.5 Johnson-Holmquist Ceramic Models

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL JH1CERAMIC_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    BULKING CNST = <real>BULKING_CNST
    DAMAGE CNST DP1 = <real>DAMAGE_CNST_DP1
    FSTRENGTH MAX = <real>FSTRENGTH_MAX
    FSTRENGTH SLOPE = <real>FSTRENGTH_SLOPE
    MAX FAIL STRAIN = <real>MAX_FAIL_STRAIN
    PRESS COEF K2 = <real>PRESS_COEF_K2
    PRESS COEF K3 = <real>PRESS_COEF_K3
    STRAIN RATE COEF = <real>STRAIN_RATE_COEF
    STRENGTH CNST P1 = <real>STRENGTH_CNST_P1
    STRENGTH CNST P2 = <real>STRENGTH_CNST_P2
    STRENGTH CNST S1 = <real>STRENGTH_CNST_S1
    STRENGTH CNST S2 = <real>STRENGTH_CNST_S2
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [PARAMETERS FOR MODEL JH1CERAMIC_MMM]

  BEGIN PARAMETERS FOR MODEL JH2CERAMIC_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    BULKING CNST = <real>BULKING_CNST
    DAMAGE COEF D1 = <real>DAMAGE_COEF_D1
    DAMAGE EXP D2 = <real>DAMAGE_EXP_D2
    FSTRENGTH COEF B = <real>FSTRENGTH_COEF_B
    FSTRENGTH EXP M = <real>FSTRENGTH_EXP_M
    FSTRENGTH MAX NORM = <real>FSTRENGTH_MAX_NORM
    HEL = <real>HEL
    MIN FAIL STRAIN = <real>MIN_FAIL_STRAIN
    PRESS COEF K2 = <real>PRESS_COEF_K2
    PRESS COEF K3 = <real>PRESS_COEF_K3
```

```
    STRAIN RATE COEF = <real>STRAIN_RATE_COEF
    STRENGTH COEF A = <real>STRENGTH_COEF_A
    STRENGTH EXP N = <real>STRENGTH_EXP_N
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [PARAMETERS FOR MODEL JH2CERAMIC_MMM]

  BEGIN PARAMETERS FOR MODEL JH3CERAMIC_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    BULKING CNST = <real>BULKING_CNST
    DAMAGE COEF D1 = <real>DAMAGE_COEF_D1
    DAMAGE EXP D2 = <real>DAMAGE_EXP_D2
    FSTRENGTH COEF B = <real>FSTRENGTH_COEF_B
    FSTRENGTH EXP M = <real>FSTRENGTH_EXP_M
    FSTRENGTH MAX NORM = <real>FSTRENGTH_MAX_NORM
    HEL = <real>HEL
    MIN FAIL STRAIN = <real>MIN_FAIL_STRAIN
    PRESS COEF K2 = <real>PRESS_COEF_K2
    PRESS COEF K3 = <real>PRESS_COEF_K3
    STRAIN RATE COEF = <real>STRAIN_RATE_COEF
    STRENGTH COEF A = <real>STRENGTH_COEF_A
    STRENGTH EXP N = <real>STRENGTH_EXP_N
    MAX TENS PRESS = <real>MAX_TENS_PRESS
   END [PARAMETERS FOR MODEL JH3CERAMIC_MMM ]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Johnson-Holmquist Ceramic models 1 through 3. The three models differ slightly in how they handle failure. More information is available in Reference [1]

The command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL JH#CERAMIC_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL JH#CERAMIC_MMM]
```

where # is 1, 2, or 3.

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material is defined with the `YIELD STRESS` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - The maximum permitted tensile pressure is defined with the `MAX TENS PRESS` command line.
  - The remaining command lines are described in Reference [1].

Output variables available for these models are listed in Table 6.6. More information about these models is available in Reference [1] and [3].

2.1.6 Johnson-Holmquist-Beissel Ceramic Models

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL JHB1CERAMIC_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    BULKING CNST = <real>BULKING_CNST
    DAMAGE COEF D1 = <real>DAMAGE_COEF_D1
    DAMAGE EXP N = <real>DAMAGE_EXP_N
    FSTRENGTH CNST PF = <real>FSTRENGTH_CNST_PF
    FSTRENGTH CNST SF = <real>FSTRENGTH_CNST_SF
    FSTRENGTH MAX = <real>FSTRENGTH_MAX
    MAX FAIL STRAIN = <real>MAX_FAIL_STRAIN
    PRESS COEF K2 = <real>PRESS_COEF_K2
    PRESS COEF K3 = <real>PRESS_COEF_K3
    STRAIN RATE COEF = <real>STRAIN_RATE_COEF
    STRENGTH CNST PI = <real>STRENGTH_CNST_PI
    STRENGTH CNST SI = <real>STRENGTH_CNST_SI
    STRENGTH MAX = <real>STRENGTH_MAX
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [PARAMETERS FOR MODEL JHB1CERAMIC_MMM ]

  BEGIN PARAMETERS FOR MODEL JHB2CERAMIC_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    BULKING CNST = <real>BULKING_CNST
    DAMAGE COEF D1 = <real>DAMAGE_COEF_D1
    DAMAGE EXP N = <real>DAMAGE_EXP_N
    FSTRENGTH CNST PF = <real>FSTRENGTH_CNST_PF
    FSTRENGTH CNST SF = <real>FSTRENGTH_CNST_SF
    FSTRENGTH MAX = <real>FSTRENGTH_MAX
    HYSTERESIS CNST = <real>HYSTERESIS_CNST
    MAX FAIL STRAIN = <real>MAX_FAIL_STRAIN
    PHASE TRAN P1 = <real>PHASE_TRAN_P1
```

```
    PHASE TRAN P2 = <real>PHASE_TRAN_P2
    PHASE2 KP1 = <real>PHASE2_KP1
    PHASE2 KP2 = <real>PHASE2_KP2
    PHASE2 KP3 = <real>PHASE2_KP3
    PHASE2 UPZERO = <real>PHASE2_UPZERO
    PRESS COEF K2 = <real>PRESS_COEF_K2
    PRESS COEF K3 = <real>PRESS_COEF_K3
    STRAIN RATE COEF = <real>STRAIN_RATE_COEF
    STRENGTH CNST PI = <real>STRENGTH_CNST_PI
    STRENGTH CNST SI = <real>STRENGTH_CNST_SI
    STRENGTH MAX = <real>STRENGTH_MAX
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [PARAMETERS FOR MODEL JHB2CERAMIC_MMM]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Johnson-Holmquist-Beissel Ceramic models 1 and 2. The two models differ slightly in how they handle failure. More information is available in Reference [1]

The command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL JHB#CERAMIC_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL JHB#CERAMIC_MMM]
```

where # is either 1 or 2.

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.

- The maximum permitted tensile pressure is defined with the `MAX TENS PRESS` command line.
- The remaining command lines are described in Reference [1].

Output variables available for these models are listed in Table 6.7. More information about these models is available in Reference [1] and [3].

2.1.7 Mechanical Threshold Stress (MTS) strength model with Mie-Gruneisen EOS

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL MTSSTRESS_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    ABS ZERO TEMP = <real>ABS_ZERO_TEMP
    INIT TEMPERATURE = <real>INIT_TEMPERATURE
    MELT TEMPERATURE = <real>MELT_TEMPERATURE
    SPECIFIC HEAT = <real>SPECIFIC_HEAT
    ABS ZERO SHRMOD SM0 = <real>ABS_ZERO_SHRMOD_SM0
    BURGVEC MAG = <real>BURGVEC_MAG
    CNST ALPHA = <real>CNST_ALPHA
    CNST BOLTZ = <real>CNST_BOLTZ
    CNST CAPA = <real>CNST_CAPA
    CNST PINV = <real>CNST_PINV
    CNST PINVI = <real>CNST_PINVI
    CNST PINVS = <real>CNST_PINVS
    CNST QINV = <real>CNST_QINV
    CNST QINVI = <real>CNST_QINVI
    CNST QINVS = <real>CNST_QINVS
    DISLOC CNST HF0 = <real>DISLOC_CNST_HF0
    DISLOC CNST HF1 = <real>DISLOC_CNST_HF1
    DISLOC CNST HF2 = <real>DISLOC_CNST_HF2
    DISLOC CNST SIGA = <real>DISLOC_CNST_SIGA
    DISLOC CNST SIGI = <real>DISLOC_CNST_SIGI
    DISLOC CNST SIGS = <real>DISLOC_CNST_SIGS
    INIT STATE VAR SIG0 = <real>INIT_STATE_VAR_SIG0
    NORM ACT ENRGY G0 = <real>NORM_ACT_ENRGY_G0
    NORM ACT ENRGY G0I = <real>NORM_ACT_ENRGY_G0I
    NORM ACT ENRGY G0S = <real>NORM_ACT_ENRGY_G0S
    REF STN RAT EDOT0 = <real>REF_STN_RAT_EDOT0
    REF STN RAT EDOTI = <real>REF_STN_RAT_EDOTI
    REF STN RAT EDOTS = <real>REF_STN_RAT_EDOTS
    REF STN RAT EDOTS0 = <real>REF_STN_RAT_EDOTS0
    SAT TH STS SIGS0 = <real>SAT_TH_STS_SIGS0
    SHRMOD CNST SM1 = <real>SHRMOD_CNST_SM1
    SHRMOD CNST SM2 = <real>SHRMOD_CNST_SM2
    GRUN COEF = <real>GRUN_COEF
    MIEGRU COEF K2 = <real>MIEGRU_COEF_K2
```

```
    MIEGRU COEF K3 = <real>MIEGRU_COEF_K3
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [PARAMETERS FOR MODEL MTSSTRESS_MMM]

END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Mechanical Threshold Stress (MTS) model with a cubic Mie-Gruneisen EOS. The MTS model has a yield function defined by

$$\sigma = \hat{\sigma}_a + \frac{G}{G_0}(s_{th}\hat{\sigma} + s_{th,i}\hat{\sigma}_i + s_{th,s}\hat{\sigma}_s) \tag{2.13}$$

where $\hat{\sigma}$ is the mechanical threshold stress (defined below), $\hat{\sigma}_a$, $\hat{\sigma}_i$, and $\hat{\sigma}_s$ are constants representing dislocation interactions corresponding to long-range barriers, interstitial atoms, and solute atoms, and G_0 is the shear modulus at absolute zero. The shear modulus at other temperatures are defined as $G = G_0 - b_1/(exp(b_2/T) - 1)$, where b_1 and b_2 are material constants and T is the absolute temperature.

The s_{th} terms have the general form

$$s_{th} = \left[1 - \left(\frac{kT\ ln(\dot{\epsilon}_0/\dot{\epsilon})}{Gb^3 g_0}\right)^{\frac{1}{q}}\right]^{\frac{1}{p}} \tag{2.14}$$

where k is the Boltzmann constant, b is the magnitude of the Burger's vector, g_0 is a normalized activation energy, $\dot{\epsilon}_0$ is a reference strain rate, and p and q are exponential constants. For $s_{th,i}$ and $s_{th,s}$, the equation is identical but with different constants.

The update of the mechanical threshold stress $\hat{\sigma}$ is governed by

$$\hat{\sigma}_{t+\Delta t} = \hat{\sigma}_t + \frac{\delta\hat{\sigma}}{\delta\epsilon_p}(\dot{\epsilon}_p \Delta t) \tag{2.15}$$

where

$$\frac{\delta\hat{\sigma}}{\delta\epsilon_p} = \Theta_0 \left[1 - \frac{tanh\left(\alpha\frac{\hat{\sigma}}{\hat{\sigma}_s}\right)}{tanh(\alpha)}\right] \tag{2.16}$$

$$\Theta_0 = a_0 + a_1\ ln(\dot{\epsilon}) + a_2\sqrt{\dot{\epsilon}} \tag{2.17}$$

$$\hat{\sigma}_s = \hat{\sigma}_{so}\left(\frac{\dot{\epsilon}}{\dot{\epsilon}_{so}}\right)^{kT/Gb^3 A} \tag{2.18}$$

where A, α, a_0, a_1, and a_2 are material constants, $\hat{\sigma}_{so}$ is the saturation threshold stress, and $\dot{\epsilon}_{so}$ is a reference strain rate.

The command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL MTSSTRESS_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL MTSSTRESS_MMM]
```

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material is defined with the `YIELD STRESS` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - The temperature at absolute zero is defined with the `ABS ZERO TEMP` command line.
  - The initial temperature is defined with the `INIT TEMPERATURE` command line.
  - The melt temperature is defined with the `MELT TEMPERATURE` command line.
  - The specific heat is defined with the `SPECIFIC HEAT` command line.
  - The shear modulus at absolute zero (G_0) is defined with the `ABS ZERO SHRMOD SM0` command line.
  - The magnitude of the Burgers vector (b) is defined with the `BURGVEC MAG` command line.
  - The material constant α is defined with the `CNST ALPHA` command line.
  - The material constant A is defined with the `CNST CAPA` command line.
  - The material constant a_0 is defined with the `DISLOC CNST HF0` command line.
  - The material constant a_1 is defined with the `DISLOC CNST HF1` command line.
  - The material constant a_2 is defined with the `DISLOC CNST HF2` command line.
  - The Boltzmann constant k is defined with the `CNST BOLTZ` command line.

- The dislocation interaction constant $\hat{\sigma}_a$ is defined with the `DISLOC CNST SIGA` command line.
- The dislocation interaction constant $\hat{\sigma}_i$ is defined with the `DISLOC CNST SIGI` command line.
- The dislocation interaction constant $\hat{\sigma}_s$ is defined with the `DISLOC CNST SIGS` command line.
- The $1/p$ exponent in the equation for s_{th} is defined with the `CNST PINV` command line.
- The $1/p$ exponent in the equation for $s_{th,i}$ is defined with the `CNST PINVI` command line.
- The $1/p$ exponent in the equation for $s_{th,s}$ is defined with the `CNST PINVS` command line.
- The $1/q$ exponent in the equation for s_{th} is defined with the `CNST QINV` command line.
- The $1/q$ exponent in the equation for $s_{th,i}$ is defined with the `CNST QINVI` command line.
- The $1/q$ exponent in the equation for $s_{th,s}$ is defined with the `CNST QINVS` command line.
- The g_0 value in the equation for s_{th} is defined with the `NORM ACT ENRGY G0` command line.
- The g_0 value in the equation for $s_{th,i}$ is defined with the `NORM ACT ENRGY G0I` command line.
- The g_0 value in the equation for $s_{th,s}$ is defined with the `NORM ACT ENRGY G0S` command line.
- The $\dot{\epsilon}_0$ value in the equation for s_{th} is defined with the `REF STN RAT EDOT0` command line.
- The $\dot{\epsilon}_0$ value in the equation for $s_{th,i}$ is defined with the `REF STN RAT EDOTI` command line.
- The $\dot{\epsilon}_0$ value in the equation for $s_{th,s}$ is defined with the `REF STN RAT EDOTS` command line.
- The initial value for the mechanical threshold stress $\hat{\sigma}$ is defined with the `INIT STATE VAR SIG0` command line.
- The value for $\hat{\sigma}_{so}$ in the equation for the saturation stress $\hat{\sigma}_s$ is defined with the `SAT TH STS SIGS0` command line.
- The value for $\dot{\epsilon}_{so}$ in the equation for the saturation stress $\hat{\sigma}_s$ is defined with the `REF STN RAT EDOTS0` command line.
- The material constant b_1 in the temperature shear modulus equation is defined with the `SHRMOD CNST SM1` command line.
- The material constant b_2 in the temperature shear modulus equation is defined with the `SHRMOD CNST SM2` command line.

- The Gruneisen parameter Gamma is defined with the `GRUN COEF` command line.
- The K2 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K2` command line.
- The K3 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K3` command line.
- The maximum permitted tensile pressure is defined with the `MAX TENS PRESS` command line.

More information about this model is available in Reference [1].

2.1.8 Mechanical Threshold Stress strength model with Mie-Gruneisen EOS and the TEPLA continuum level damage model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL TEPLA_MTSSTRESS_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    ABS ZERO TEMP = <real>ABS_ZERO_TEMP
    INIT TEMPERATURE = <real>INIT_TEMPERATURE
    MELT TEMPERATURE = <real>MELT_TEMPERATURE
    SPECIFIC HEAT = <real>SPECIFIC_HEAT
    ABS ZERO SHRMOD SM0 = <real>ABS_ZERO_SHRMOD_SM0
    ALPHA11 = <real>ALPHA11
    ALPHA21 = <real>ALPHA21
    ALPHA22 = <real>ALPHA22
    ALPHA31 = <real>ALPHA31
    ALPHA32 = <real>ALPHA32
    ALPHA33 = <real>ALPHA33
    ALPHA41 = <real>ALPHA41
    ALPHA42 = <real>ALPHA42
    ALPHA43 = <real>ALPHA43
    ALPHA44 = <real>ALPHA44
    ALPHA51 = <real>ALPHA51
    ALPHA52 = <real>ALPHA52
    ALPHA53 = <real>ALPHA53
    ALPHA54 = <real>ALPHA54
    ALPHA55 = <real>ALPHA55
    BURGVEC MAG = <real>BURGVEC_MAG
    CNST ALPHA = <real>CNST_ALPHA
    CNST BOLTZ = <real>CNST_BOLTZ
    CNST CAPA = <real>CNST_CAPA
    CNST PINV = <real>CNST_PINV
    CNST PINVI = <real>CNST_PINVI
    CNST PINVS = <real>CNST_PINVS
    CNST QINV = <real>CNST_QINV
    CNST QINVI = <real>CNST_QINVI
    CNST QINVS = <real>CNST_QINVS
    DISLOC CNST HF0 = <real>DISLOC_CNST_HF0
    DISLOC CNST HF1 = <real>DISLOC_CNST_HF1
    DISLOC CNST HF2 = <real>DISLOC_CNST_HF2
```

```
DISLOC CNST SIGA = <real>DISLOC_CNST_SIGA
DISLOC CNST SIGI = <real>DISLOC_CNST_SIGI
DISLOC CNST SIGS = <real>DISLOC_CNST_SIGS
E11 = <real>E11
E21 = <real>E21
E22 = <real>E22
E31 = <real>E31
E32 = <real>E32
E33 = <real>E33
E41 = <real>E41
E42 = <real>E42
E43 = <real>E43
E44 = <real>E44
E51 = <real>E51
E52 = <real>E52
E53 = <real>E53
E54 = <real>E54
E55 = <real>E55
E61 = <real>E61
E62 = <real>E62
E63 = <real>E63
E64 = <real>E64
E65 = <real>E65
E66 = <real>E66
FAIL POR PHIF = <real>FAIL_POR_PHIF
FAIL SURF GAMA0 = <real>FAIL_SURF_GAMA0
FAIL SURF GAMA1 = <real>FAIL_SURF_GAMA1
FAIL SURF GAMA2 = <real>FAIL_SURF_GAMA2
ICOMP = <real>ICOMP
INIT POR PHI0 = <real>INIT_POR_PHI0
INIT STATE VAR SIG0 = <real>INIT_STATE_VAR_SIG0
LENGTH SCALE = <real>LENGTH_SCALE
NORM ACT ENRGY G0 = <real>NORM_ACT_ENRGY_G0
NORM ACT ENRGY G0I = <real>NORM_ACT_ENRGY_G0I
NORM ACT ENRGY G0S = <real>NORM_ACT_ENRGY_G0S
ORTHO = <real>ORTHO
REF STN RAT EDOT0 = <real>REF_STN_RAT_EDOT0
REF STN RAT EDOTI = <real>REF_STN_RAT_EDOTI
REF STN RAT EDOTS = <real>REF_STN_RAT_EDOTS
REF STN RAT EDOTS0 = <real>REF_STN_RAT_EDOTS0
RODRIGUES ANGLE = <real>RODRIGUES_ANGLE
RODRIGUES X = <real>RODRIGUES_X
RODRIGUES Y = <real>RODRIGUES_Y
RODRIGUES Z = <real>RODRIGUES_Z
SAT TH STS SIGS0 = <real>SAT_TH_STS_SIGS0
SHRMOD CNST SM1 = <real>SHRMOD_CNST_SM1
SHRMOD CNST SM2 = <real>SHRMOD_CNST_SM2
```

```
    VOID GROW PAR QG1 = <real>VOID_GROW_PAR_QG1
    VOID GROW PAR QG2 = <real>VOID_GROW_PAR_QG2
    VOID GROW PAR QG3 = <real>VOID_GROW_PAR_QG3
    GRUN COEF = <real>GRUN_COEF
    MIEGRU COEF K2 = <real>MIEGRU_COEF_K2
    MIEGRU COEF K3 = <real>MIEGRU_COEF_K3
  END [PARAMETERS FOR MODEL TEPLA_MTSSTRESS_MMM ]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

This specification activates the Mechanical Threshold Stress (MTS) strength model with a cubic Mie-Gruneisen EOS and the TEPLA continuum level damage model. This model is an extension of the standard MTS model, as described in section 2.1.7. The extensions provide an ability to initialize th porosity, pressure, failure porosity, flow stress, rotation, and stretch arrays and the specification of an orthotropic yield function. They also modify the MTS model to include the effect of evolving porosity (void growth) through an extended Gurson model. More information on this model is available in References [1] and [4].

The command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL TEPLA_MTSSTRESS_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL TEPLA_MTSSTRESS_MMM]
```

Most of the commands for this material are identical to those defined in section 2.1.7. In addition:

- The command `ORTHO` specifies that the material is orthotropic if set to 1, or isotropic if set to 0.
- The terms described by the commands `ALPHA11` through `ALPHA55` define the plastic shape tensor components.
- The terms described by the commands `E11` through `E66` represent the elastic stiffness tensor for an orthotropic material.
- The Rodrigues vector for the orthotropic yield surface is defined by the commands `RODRIGUES [X|Y|Z]`.
- The Rodrigues angle is the angle of rotation around the Rodrigues vector, and is defined by the command `RODRIGUES ANGLE`.
- The initial porosity is defined by the command `INIT POR PHI0`.
- The final porosity at failure is given by the command `FAIL POR PHIF`.

- The command ICOMP toggles pore growth; if it is 0, then pores can grow, whereas if it is 1, pores do not grow.
- The commands VOID GROW PAR QG[1|2|3] define the coefficients for the Tvergaard porosity evolution equation.
- The length scale for the over-stress formulation is specified by the command LENGTH SCALE.
- The commands FAIL SURF GAMA[0|1|2] define the material constants in the expression for the failure strain.

More information about this model is available in Reference [1].

2.1.9 Zerilli-Armstrong strength model for BCC metals with Mie-Gruneisen EOS

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
  BEGIN PARAMETERS FOR MODEL ZABCCSTRESS_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    ABS ZERO TEMP = <real>ABS_ZERO_TEMP
    INIT TEMPERATURE = <real>INIT_TEMPERATURE
    SPECIFIC HEAT = <real>SPECIFIC_HEAT
    STRAIN HARD COEF C5 = <real>STRAIN_HARD_COEF_C5
    STRAIN HARD EXP N = <real>STRAIN_HARD_EXP_N
    STRAIN RATE COEF C1 = <real>STRAIN_RATE_COEF_C1
    STRAIN RATE COEF C4 = <real>STRAIN_RATE_COEF_C4
    THERM SOFT COEF C3 = <real>THERM_SOFT_COEF_C3
    YIELD STRESS C0 = <real>YIELD_STRESS_C0
    GRUN COEF = <real>GRUN_COEF
    MIEGRU COEF K2 = <real>MIEGRU_COEF_K2
    MIEGRU COEF K3 = <real>MIEGRU_COEF_K3
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [PARAMETERS FOR MODEL ZABCCSTRESS_MMM]
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
```

This specification activates the Zerilli-Armstrong strength model for BCC metals with a Mie-Gruneisen EOS. The expression for the yield function of this model is:

$$\sigma = C_0 + C_1\ exp(-C_3 T + C_4 T\ ln\ \dot{\epsilon}) + C_5 \epsilon_p^n \tag{2.19}$$

where ϵ_p is the equivalent plastic strain, T is the absolute temperature, $\dot{\epsilon}$ is the equivalent total strain rate, and C_0, C_1, C_3, C_4, C_5, and n are material constants.

The pressure response is described by a cubit Mie-Gruneisen model – see equation (2.5) for more details.

The command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL ZABCCSTRESS_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL ZABCCSTRESS_MMM]
```

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material is defined with the `YIELD STRESS` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - The temperature at absolute zero is defined with the `ABS ZERO TEMP` command line.
  - The initial temperature is defined with the `INIT TEMPERATURE` command line.
  - The specific heat is defined with the `SPECIFIC HEAT` command line.
  - The material constants C_0, C_1, C_3, C_4, C_5, and n are defined with the corresponding command lines above.
  - The Gruneisen parameter Gamma is defined with the `GRUN COEF` command line.
  - The K2 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K2` command line.
  - The K3 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K3` command line.
  - The maximum permitted tensile pressure is defined with the `MAX TENS PRESS` command line.

More information about this model is available in Reference [1].

2.1.10 Zerilli-Armstrong strength model for FCC metals with Mie-Gruneisen EOS

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  DENSITY = <real>density_value
  #
 BEGIN PARAMETERS FOR MODEL ZAFCCSTRESS_MMM
    TWO MU = <real>two_mu
    YOUNGS MODULUS = <real>youngs_modulus
    BULK MODULUS = <real>bulk_modulus
    POISSONS RATIO = <real>poissons_ratio
    SHEAR MODULUS = <real>shear_modulus
    LAMBDA = <real>lambda
    YIELD STRESS = <real>yield_stress
    INIT DENSITY = <real>init_density
    ABS ZERO TEMP = <real>ABS_ZERO_TEMP
    INIT TEMPERATURE = <real>INIT_TEMPERATURE
    SPECIFIC HEAT = <real>SPECIFIC_HEAT
    STRAIN HARD COEF C2 = <real>STRAIN_HARD_COEF_C2
    STRAIN HARD EXP N = <real>STRAIN_HARD_EXP_N
    STRAIN RATE COEF C4 = <real>STRAIN_RATE_COEF_C4
    THERM SOFT COEF C3 = <real>THERM_SOFT_COEF_C3
    YIELD STRESS C0 = <real>YIELD_STRESS_C0
    GRUN COEF = <real>GRUN_COEF
    MIEGRU COEF K2 = <real>MIEGRU_COEF_K2
    MIEGRU COEF K3 = <real>MIEGRU_COEF_K3
    MAX TENS PRESS = <real>MAX_TENS_PRESS
  END [ PARAMETERS FOR MODEL ZAFCCSTRESS_MMM ]
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
```

This specification activates the Zerilli-Armstrong strength model for FCC metals with a Mie-Gruneisen EOS. The expression for the yield function of this model is:

$$\sigma = C_0 + C_2 \epsilon_p^n \, exp(-C_3 T + C_4 T \, ln \, \dot{\epsilon}) \tag{2.20}$$

where ϵ_p is the equivalent plastic strain, T is the absolute temperature, $\dot{\epsilon}$ is the equivalent total strain rate, and C_0, C_2, C_3, C_4, and n are material constants.

The pressure response is described by a cubit Mie-Gruneisen model – see equation (2.5) for more details.

The command block starts with the input line:

```
BEGIN PARAMETERS FOR MODEL ZAFCCSTRESS_MMM
```

and terminates with an input line of the following form:

```
END [PARAMETERS FOR MODEL ZAFCCSTRESS_MMM]
```

In the above command blocks:

- The density of the material is defined with the `DENSITY` command line.
- Only two of the following elastic constants are required to define the unscaled bulk behavior:
  - Young's modulus is defined with the `YOUNGS MODULUS` command line.
  - Poisson's ratio is defined with the `POISSONS RATIO` command line.
  - The bulk modulus is defined with the `BULK MODULUS` command line.
  - The shear modulus is defined with the `SHEAR MODULUS` command line.
  - Lambda is defined with the `LAMBDA` command line.
- The following command lines are required:
  - The yield stress of the material is defined with the `YIELD STRESS` command line.
  - The initial density of the material is defined with the `INITIAL DENSITY` command line. Set this equal to the density specified with the `DENSITY` command line.
  - The temperature at absolute zero is defined with the `ABS ZERO TEMP` command line.
  - The initial temperature is defined with the `INIT TEMPERATURE` command line.
  - The specific heat is defined with the `SPECIFIC HEAT` command line.
  - The material constants C_0, C_2, C_3, C_4, and n are defined with the corresponding command lines above.
  - The Gruneisen parameter Gamma is defined with the `GRUN COEF` command line.
  - The K2 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K2` command line.
  - The K3 parameter for the MMM cubic Mie-Gruneisen model is defined with the `MIEGRUN COEF K3` command line.
  - The maximum permitted tensile pressure is defined with the `MAX TENS PRESS` command line.

More information about this model is available in Reference [1].

2.2 CTH Model Specifications

This section describes material models that have been ported from CTH to the LAME material library[2]. Because of the ITAR export-control restrictions on these models, they are maintained separately from the standard LAME material library and only linked in with Presto_ITAR.

Warning: Support for CTH material models in Presto_ITAR is currently at an experimental level. As such, not all features may be fully implemented or tested and the analyst should use these models with caution.

Known Issue: The algorithms that apply when these energy-dependent models are in use are currently in a state of flux as they are being upgraded to the state-of-the-art. This transformation has currently been applied only to the midpoint-increment uniform-gradient hexahedron element. Attempting to use these models with any other element will likely result in code failure.

Implementation of the CTH material models departs from the typical behavior found for other material models present in Presto_ITAR. Generally, this allows the CTH models to be more flexible in the material behaviors they can represent, particularly for high strain rate, energy dependent materials. The main differences are in the treatment of the energy update, modularity, and parameter specification.

For energy dependent material models, such as those from this Section, Section 2.1, and Section 2.3, the internal energy is updated using a second order, implicit equation, see Reference [5]. For the traditional Presto_ITAR models of Sections 2.1 and 2.3, the energy update is performed as part of the material model. Additionally, all the models assume materials behave under the Mie-GrG̈runeisen assumption that pressure is linearly dependent upon the internal energy. This allows these models to explicitly solve the implicit energy equation. While this provides for an easy solution it limits the types of material behavior that can be modeled. The CTH models break from the Mie-Grüneisen assumption, allowing an arbitrary dependence of the pressure on the internal energy. This motivates several changes in how the elements treat these materials.

The energy update equation is a function of the host code in that its form and method of solution are code dependent. From a theoretical perspective, a material has no knowledge of such an equation. Additionally, for portability between codes a material model should not solve such an equation since it would possibly have to be different for every code in which it was used. For this reason the energy equation update was not pushed into the CTH material models. Instead it is computed in the element itself. In the future, other material models from Sections 2.1 and 2.3 may also have the energy update extracted from them, leading to less code duplication, better consistency across models, and better maintainability.

Not only was the energy update extracted from the material models for the CTH models, but the assumption of a Mie-Grüneisen form also had to be removed. This requires one to perform an iterative solve of the energy equation to be self consistent, since the explicit solution is no longer viable. The initial guess to the solution is based upon a predictor method for the hydrodynamic work. Later iterations include a fully implicit solve of the hydrodynamic and deviatoric work. For information on controlling the iterative solution of the energy equation, see Section 3.1.1.2.

The CTH models also depart from the other Presto_ITAR material models in that they adopt the concept of modularity. Typically, a solid might have an equation of state model and a yield model. Models from Sections 2.1 and 2.3 explicitly couple these models together. Thus, if one want to use an already implemented yield model with a new equation of state, then one has to write a new material model which couples them together. On the other hand, the CTH models are modular (although not completely) in that if a given model adheres to a certain interface, it may be used as a drop in replacement for other models using the same interface. Thus only the new submodel has to be implemented. Currently there is a single implementation of a CTH modular model in Presto_ITAR, the `CTH_EP` model of Section 2.2.3.

One side effect of the modularity concept is that not all models compute a stress. Those that do not cannot be called directly from an element, and hence cannot be used as the material model for an element. See, for example, the `CTH_JO` model of Section 2.2.4. On the other hand, equation of state models, such as the `CTH_MGR` model of Section 2.2.1, do compute a stress and so they can not only be used as a submodel in a modular model such as `CTH_EP`, but may also be used directly as an element material model.

Parameter parsing behavior has also been modified from the standard Presto_ITAR practice in the CTH models. Unlike most of the material models, which require all parameters to be specified, the CTH models have default values for most parameters. Additionally, the CTH models introduce the concept of material parameter libraries. These libraries are essentially look up tables for the parameters of predefined materials. Thus one need only specify a material model, such as `CTH_KSES`, and a material name like `MATLABEL = ALUMINUM`. All the parameters are then automatically loaded. Note that if a predefined material is specified, one may override library values by additionally specifying the desired properties. When no library material is specified, this is essentially what occurs, as the entry `MATLABEL = USER` is implicitly specified to read the default parameters from the material library.

Many models are unit independent, in that any set of parameters with a consistent set of units will work correctly with such models. This is the case for most of the models in Presto_ITAR. However, with the CTH models this assumption is broken for certain equation of state models as well as by the use of material libraries. Thus, all CTH models must specify a system of units. Note that while this is only required for full models and not sub models, submodel parameters should be specified in units consistent with their parent model. In general, the unit declarations have the form given by the following block.

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL <string>mod_name
    # SI    = International system of units
    # IPS   = inch-pound-seconds
    # CGSK  = centimeters-grams-seconds-kelvin
    # CGSEV = centimeters-grams-seconds-electron volts
    UNIT SYSTEM = <string>SI|IPS|CGSK|CGSEV|SESAME|SSHOCK(SI)
    LENGTH UNIT = <real>length_unit(1.0)
    MASS UNIT = <real>mass_unit(1.0)
```

```
    TIME UNIT = <real>time_unit(1.0)
    TEMPERATURE UNIT = <real>temperature_unit(1.0)
    AMOUNT UNIT = <real>amount_unit(1.0)
    CURRENT UNIT = <real>current_unit(1.0)
    LUMINOSITY UNIT = <real>luminosity_unit(1.0)
    ...
  END [PARAMETERS FOR MODEL <string>mod_name]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The `UNIT SYSTEM` command line may be used to specify an overall system of units to use. The values of that system may then be overridden for specific units by the other commands. The `...` denotes all the other parameters of the model, which have been omitted here. The meaning of a unit command is the value required to convert from SI to the desired unit system. Thus, for example, if one has a problem where length is measured in centimeters, one would specify `LENGTH UNIT = 1.e2`, since there are one hundred centimeters in a meter. Once the unit system has been specified in this manner, all the model parameters must be entered in this system.

Paths to the material libraries, as well as certain tabular data required by the CTH SESAME models, must be specified in the user input as well. Specific parameters are available for setting the names of data files in the model input. These may be relative or absolute paths. Additionally, the models recognize the existence of the environment variable `CTHPATH`. When `CTHPATH` is undefined, the default path for all CTH data is relative to the current directory. When `CTHPATH` is defined, then SESAME table data is searched for relative to the directory `CTHPATH/data/`. Also, in this case material libraries are first searched for relative to the working directory and upon failure of that, relative to the directory `CTHPATH/data/`. If a model cannot find its material library file, it will throw a fatal error.

2.2.1 Mie-Grüneisen Model (CTH_MGR)

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL CTH_MGR
    {unit parameters}
    MATLABEL = <string>material_label(USER)
    EOS DATAFILE = <string>eos_data_file(EOS_data)
    R0 = <real>density
    T0 = <real>temperature(298.0)
    CS = <real>sound_speed
    S1 = <real>us_up_slope(0.0)
    G0 = <real>gruneisen_parameter(0.0)
    CV = <real>heat_capacity
    ESFT = <real>energy_shift(0.0)
    RP = <real>porous_density(0.0)
    PS = <real>crushup_pressure(1.e9)
    PE = <real>elastic_pressure(0.0)
    CE = <real>elastic_sound_speed(0.0)
    NSUB = <real>num_subcycles(10.0)
    S2 = <real>us_up_quadratic(0.0)
    TYP = <real>model_type(1.0)
    RO = <real>density_alias
    TO = <real>temperature_alias
    S = <real>s1_alias
    GO = <real>g0_alias
    B = <real>low_pressure_coefficient(0.0)
    XB = <real>low_pressure_constant(1.e-4)
    NB = <real>low_pressure_power(1.0)
    PWR = <real>alpha_power(2.0)
  END [PARAMETERS FOR MODEL CTH_MGR]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Mie-Grüneisen material model describes the nonlinear pressure-volume (or equivalently pressure-density) response of solids or fluids in terms of a reference pressure-volume curve and deviations from the reference curve in energy space. The reference curve is taken to be the experimentally determined principal Hugoniot, which is the locus of end states that can be reached by a shock transition from the ambient state. For details about this model, see Reference [6].

For Mie-Grüneisen energy-dependent materials, the Mie-Grüneisen command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL CTH_MGR
```

and is terminated with an input line of the following form:

```
END [PARAMETERS FOR MODEL CTH_MGR]
```

In the above command blocks:

- The `{unit parameters}` line is a placeholder for the unit block described in Section 2.2.
- The `MATLABEL` command line specifies the name of a material parameter library entry from which to take default values for the other parameters. This name is searched for under the model name `MGR` in the data file specified by the command line `EOS DATAFILE`.
- The command lines `R0` (or `RO`), `CS`, and `CV` are required inputs to this model. Alternatively, one may specify a non-default `MATLABEL` command line. All other values are optional and may be left unspecified if the defaults are acceptable.
- The initial density for the Hugoniot is defined with the `R0` command line. If the material is porous, the `RP` command line defines the initial density and `R0` is the ambient density for the nonporous material.

For information about the CTH Mie-Grüneisen model, consult Reference [6].

2.2.2 SESAME Tabular EOS Model (CTH_KSES)

Known Issue:
The SESAME tabular interface currently reads tables from a platform-specific binary table format. Production of this format from the ASCII tables requires use of the `bcat` code, which is not built by default. If a current CTH installation is available, then one may use that installation's data by setting the `CTHPATH` environment variable, see Section 2.2.

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL CTH_KSES
    {unit parameters}
    MATLABEL = <string>material_label(USER)
    EOS DATAFILE = <string>eos_data_file(EOS_data)
    EOS = <real>eos_number
    SR = <real>scaling_factor(1.0)
    R0 = <real>density(table value)
    T0 = <real>temperature(table value)
    RMIN = <real>min_tension_density(0.8*R0)
    ZNUC = <real>avg_atomic_number(table value)
    ATWT = <real>avg_atomic_weight(table value)
    RP = <real>porous_density(0.0)
    PS = <real>crushup_pressure(1.e9)
    PE = <real>elastic_pressure(0.0)
    CE = <real>elastic_sound_speed(0.0)
    NSUB = <real>num_subcycles(10.0)
    ESFT = <real>energy_shift(table specific)
    TYP = <real>model_type(1.0)
    RO = <real>density_alias
    TO = <real>temperature_alias
    CLIP = <real>temperature_clip(0.0)
    PWR = <real>alpha_power(2.0)
    FEOS = <string>sesame_file
  END [PARAMETERS FOR MODEL CTH_KSES]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The SESAME tabular EOS model represents the thermodynamic state of a material through tabular representations of the pressure and internal energy as functions of density and temperature. Such tables may represent behavior as simple as an ideal gas to extremely complicated multi-phase behaviors. For more information on the implementation of this model, consult Reference [6]. Information on the SESAME format may be obtained from Reference [7].

For SESAME materials, the SESAME command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL CTH_KSES
```

and is terminated an input line of the following form:

```
END [PARAMETERS FOR MODEL CTH_KSES]
```

In the above command blocks:

- The `{unit parameters}` line is a placeholder for the unit block described in Section 2.2.
- The `MATLABEL` command line specifies the name of a material parameter library entry from which to take default values for the other parameters. This name is searched for under the model name `SES` in the data file specified by the command line `EOS DATAFILE`.
- The command lines `EOS` and `FEOS` are required inputs to this model. Alternatively, one may specify a non-default `MATLABEL` command line. All other values are optional and may be left unspecified if the defaults are acceptable.
- The command lines `R0`, `T0`, `ZNUC`, and `ATWT` default to the values given in the specified table.
- The command line `ESFT` defaults to a value such that the internal energy of the specified table will be strictly positive for all states. Care should be taken if setting this to a non-default value as one may break assumptions on the positivity of the internal energy present in other areas of the code.
- For a porous material the `RP` command line defines the initial density and `R0` becomes the ambient density for the nonporous material.
- The command line `CLIP` sets a delta in temperature from the edge of the table to which off-table temperatures are returned. In this implementation, extrapolation off of the tabulated region of a SESAME table can produce unphysical behavior. Thus, it is recommended to set `CLIP` to a non-zero value. The default, `CLIP = 0.0`, is to not clip off-table temperatures. The temperature delta is taken as the absolute value of `CLIP`. Setting a negative value suppresses error messages generated by this process.

For information about the SESAME tabular EOS model, consult Reference [6].

2.2.3 Elastic-Plastic Modular Model (CTH_EP)

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL CTH_EP
    {unit parameters}
    EOS MODEL = <string>CTH_KSES|CTH_MGR
    {eos model parameters}
    YIELD MODEL = <string>CTH_JO|CTH_ST|CTH_ZE|NONE(NONE)
    {yield model parameters}
    FRACTURE MODEL = <string>CTH_JFRAC|NONE(NONE)
    {fracture model parameters}
    RHOL = <real>lower_density(0.0)
    RHOU = <real>upper_density(0.0)
  END [PARAMETERS FOR MODEL CTH_EP]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Elastic-Plastic Modular model combines an EOS, yield, and fracture model in the manner that CTH employs. In particular, the yield models are all of the “traditional” version which compute a yield stress and shear modulus. The resultant stress is calculated from a radial return plasticity model. Density degradation of the yield stress is applied when the density lies between the upper and lower density limits.

For the Elastic-Plastic Modular model, the `CTH_EP` command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL CTH_EP
```

and is terminated an input line of the following form:

```
END [PARAMETERS FOR MODEL CTH_EP]
```

In the above command blocks:

- The `{unit parameters}` line is a placeholder for the unit block described in Section 2.2.
- The `{eos model parameters}`, `{yield model parameters}`, and `{fracture model parameters}` lines are placeholders for all the parameters of the desired eos, yield, and fracture models, respectively.
- An EOS model must be specified by the command line `EOS MODEL`. All other inputs are optional.
- Density degradation of the yield stress is performed only when the command lines `RHOL` and `RHOU` are specified and satisfy `RHOU` > `RHOL` > 0.

Output variables available for this model are listed in Table 6.2.

Known Issue:
The `CTH_EP` model does not yet implement a failure model. Thus, while the available fracture model does compute the damage of the material, this information is not acted upon.

2.2.4 Johnson-Cook Viscoplastic Model (CTH_JO)

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL <string>combined_model
    YIELD MODEL = CTH_JO
    VP DATAFILE = <string>vp_data_file(VP_data)
    YIELD MATLABEL = <string>yield_material_label(USER)
    AJO = <real>parameter_a(0.0)
    BJO = <real>parameter_b(0.0)
    CJO = <real>parameter_c(0.0)
    MJO = <real>exponent_m(0.0)
    NJO = <real>exponent_n(0.0)
    TJO = <real>melt_temperature(0.0)
    POISSON = <real>poissons_ratio(0.0)
  END [PARAMETERS FOR MODEL <string>combined_model]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Johnson-Cook Viscoplastic model updates the material yield stress based upon the plastic strain, plastic strain rate, and the temperature. For more details about this model, see Reference [8].

Since the Johnson-Cook model updates only the yield stress for a material, it must be used in combination with a plasticity model and equation of state. Currently, this means it must be used as a submodel of the Elastic-Plastic Modular model, see Section 2.2.3.

In the above command blocks:

- The `combined_model` must currently be `CTH_EP`.
- The `YIELD MATLABEL` command line specifies the name of a material parameter library entry from which to take default values for the other parameters. This name is searched for under the model name `JO` in the data file specified by the command line `VP DATAFILE`.

For information about the Johnson-Cook model, consult Reference [8].

2.2.5 Zerilli-Armstrong Plasticity Model (CTH_ZE)

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL <string>combined_model
    YIELD MODEL = CTH_ZE
    VP DATAFILE = <string>vp_data_file(VP_data)
    YIELD MATLABEL = <string>yield_material_label(USER)
    C1ZE = <real>constant_c1(0.0)
    C2ZE = <real>constant_c2(0.0)
    C3ZE = <real>constant_c3(0.0)
    C4ZE = <real>constant_c4(0.0)
    C5ZE = <real>constant_c5(0.0)
    AZE = <real>constant_a(0.0)
    NZE = <real>constant_n(0.0)
    POISSON = <real>poissons_ratio(0.0)
  END [PARAMETERS FOR MODEL <string>combined_model]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Zerilli-Armstrong Plasticity model updates the material yield stress based upon the plastic strain, plastic strain rate, and the temperature. For more details about this model, see Reference [8].

Since the Zerilli-Armstrong model updates only the yield stress for a material, it must be used in combination with a plasticity model and equation of state. Currently, this means it must be used as a submodel of the Elastic-Plastic Modular model, see Section 2.2.3.

In the above command blocks:

- The `combined_model` must currently be `CTH_EP`.
- The `YIELD MATLABEL` command line specifies the name of a material parameter library entry from which to take default values for the other parameters. This name is searched for under the model name `ZE` in the data file specified by the command line `VP DATAFILE`.

For information about the Zerilli-Armstrong model, consult Reference [8].

2.2.6 Steinberg-Guinan-Lund Plasticity Model (CTH_ST)

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL <string>combined_model
    YIELD MODEL = CTH_ST
    VP DATAFILE = <string>vp_data_file(VP_data)
    YIELD MATLABEL = <string>yield_material_label(USER)
    R0ST = <real>initial_density(0.0)
    TM0ST = <real>melt_temperature(0.0)
    ATMST = <real>melt_law_constant_a(0.0)
    GM0ST = <real>gruneisen_constant(0.0)
    AST = <real>shear_modulus_constant_a(0.0)
    BST = <real>shear_modulus_constant_b(0.0)
    NST = <real>work_hardening_constant_n(0.0)
    C1ST = <real>yield_stress_constant_c1(0.0)
    C2ST = <real>yield_stress_constant_c2(0.0)
    G0ST = <real>initial_shear_modulus(0.0)
    BTST = <real>work_hardening_constant_b(0.0)
    EIST = <real>initial_equivalent_plastic_strain(0.0)
    YPST = <real>peierls_stress(0.0)
    UKST = <real>activation_energy(0.0)
    YSMST = <real>athermal_yield_stress(0.0)
    YAST = <real>athermal_prefactor(0.0)
    Y0ST = <real>initial_yield_stress(0.0)
    YMST = <real>max_yield_stress(0.0)
    POISSON = <real>poissons_ratio(0.0)
  END [PARAMETERS FOR MODEL <string>combined_model]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Steinberg-Guinan-Lund Plasticity model updates the material yield stress and shear modulus based upon the plastic strain, plastic strain rate, the density, and the temperature. For more details about this model, see Reference [9].

Since the Steinberg-Guinan-Lund model updates only the yield stress and shear modulus for a material, it must be used in combination with a plasticity model and equation of state. Currently, this means it must be used as a submodel of the Elastic-Plastic Modular model, see Section 2.2.3.

In the above command blocks:

- The `combined_model` must currently be `CTH_EP`.
- The `YIELD MATLABEL` command line specifies the name of a material parameter library entry from which to take default values for the other parameters. This name is searched for under the model name `ST` in the data file specified by the command line `VP DATAFILE`.

For information about the Steinberg-Guinan-Lund model, consult Reference [9].

2.2.7 Johnson-Cook Fracture Model (CTH_JFRAC)

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  BEGIN PARAMETERS FOR MODEL <string>combined_model
    FRACTURE MODEL = CTH_JFRAC
    FRACTURE MATLABEL = <string>fracture_material_label(USER)
    FRACTURE DATAFILE = <string>fracture_data_file(VP_data)
    JFD1 = <real>parameter_d1(0.0)
    JFD2 = <real>parameter_d2(0.0)
    JFD3 = <real>parameter_d3(0.0)
    JFD4 = <real>parameter_d4(0.0)
    JFD5 = <real>parameter_d5(0.0)
    JFTM = <real>melt_temperature(0.0)
    JFPF0 = <real>initial_fracture_pressure(0.0)
    DYLDRD = <real>strength_degradation_damage(0.0)
    DPFRD = <real>stress_degradation_damage(0.0)
    YLDFLR = <real>minimum_yield_strength(0.0)
    FRCFLR = <real>minimum_fracture_stress(0.0)
    JFWM = <real>weibull_flag(0.0)
    JFIC = <real>random_seed_one(0.0)
    JFIT = <real>random_seed_two(0.0)
    JFVREF = <real>failure_volume(0.0)
    JFOUT = <real>output_message_flag(0.0)
  END [PARAMETERS FOR MODEL <string>combined_model]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Johnson-Cook Fracture model is a scalar damage model for predicting the failure of materials based upon the plastic strain, plastic strain rate, and yield stress. This model is completely independent of the similarly named Johnson-Cook Viscoplastic model, see Section 2.2.4. For details about this model, see Reference [10].

Since the Johnson-Cook fracture model only calculates a damage, it must be used in combination with a plasticity model, equation of state, and a failure model. Currently, this means it must be used as a submodel of the Elastic-Plastic Modular model, see Section 2.2.3.

In the above command blocks:

- The `combined_model` must currently be `CTH_EP`.
- The `FRACTURE MATLABEL` command line specifies the name of a material parameter library entry from which to take default values for the other parameters. This name is searched for under the model name `JFRAC` in the data file specified by the command line `FRACTURE DATAFILE`.
- The Weibull modulus capability is currently unimplemented.

Output variables available for this model are listed in Table 6.3. For more information about the Johnson-Cook Fracture model, consult Reference [10].

2.3 Equation-of-State Model Specifications

This section describes material models that are applicable only for use in Presto_ITAR. The algorithms that apply when these energy-dependent models are in use are currently in a state of flux as they are being upgraded to the state-of-the-art. This transformation has currently been applied only to the midpoint-increment uniform-gradient hexahedron element. When using this element with EOS models, the new algorithms are chosen by default.

2.3.1 Mie-Gruneisen Model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  # thermal strain option
  THERMAL STRAIN FUNCTION = <string>thermal_strain_function
  # or all three of the following
  THERMAL STRAIN X FUNCTION =
    <string>thermal_strain_x_function
  THERMAL STRAIN Y FUNCTION =
    <string>thermal_strain_y_function
  THERMAL STRAIN Z FUNCTION =
    <string>thermal_strain_z_function
  #
  BEGIN PARAMETERS FOR MODEL MIE_GRUNEISEN
    RHO_0 = <real>density
    C_0 = <real>sound_speed
    SHUG = <real>const_shock_velocity
    GAMMA_0 = <real>ambient_gruneisen_param
    POISSR = <real>poissons_ratio
    Y_0 = <real>yield_strength
    PMIN = <real>mean_stress(REAL_MAX)
  END [PARAMETERS FOR MODEL MIE_GRUNEISEN]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Mie-Gruneisen material model describes the nonlinear pressure-volume (or equivalently pressure-density) response of solids or fluids in terms of a reference pressure-volume curve and deviations from the reference curve in energy space. The reference curve is taken to be the experimentally determined principal Hugoniot, which is the locus of end states that can be reached by a shock transition from the ambient state. For details about this model, see Reference [5].

For Mie-Gruneisen energy-dependent materials, the Mie-Gruneisen command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL MIE_GRUNEISEN
```

and is terminated with an input line of the following form:

```
END [PARAMETERS FOR MODEL MIE_GRUNEISEN]
```

In the above command blocks:

- The `thermal strain option` is used to define thermal strains. See the Sierra/SolidMechanics 4.48 User's Guide for further information on defining and activating thermal strains.
- The ambient density, ρ_0, is defined with the `RHO_0` command line. The ambient density is the density at which the mean pressure is zero, not necessarily the initial density.

- The ambient bulk sound speed, c_0, is defined by the C_0 command line. The ambient bulk sound speed is also the first constant in the shock-velocity-versus-particle-velocity relation $D = c_0 + S u$, where u is the particle velocity. (See the following description of the SHUG command line for the definition of S.)

- The second constant in the shock-velocity-versus-particle-velocity equation, S, is defined by the SHUG command line. The shock-velocity-versus-particle-velocity relation is $D = c_0 + S u$, where u is the particle velocity. (See the previous description of the C_0 command line for the definition of c_0.)

- The ambient Gruneisen parameter, Γ_0, is defined by the GAMMA_0 command line.

- Poisson's ratio, ν, is defined by the POISSR command line. Poisson's ratio is assumed constant.

- The yield strength, y_0, is defined by the Y_0 command line. The yield strength is zero for the hydrodynamic case.

- The fracture stress is defined by the PMIN command line. The fracture stress is a mean stress or pressure, so it must be negative or zero. This is an optional parameter; if not specified, the parameter defaults to REAL_MAX (no fracture).

For information about the Mie-Gruneisen model, consult Reference [5].

2.3.2 Mie-Gruneisen Power-Series Model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  # thermal strain option
  THERMAL STRAIN FUNCTION = <string>thermal_strain_function
  # or all three of the following
  THERMAL STRAIN X FUNCTION =
    <string>thermal_strain_x_function
  THERMAL STRAIN Y FUNCTION =
    <string>thermal_strain_y_function
  THERMAL STRAIN Z FUNCTION =
    <string>thermal_strain_z_function
  #
  BEGIN PARAMETERS FOR MODEL MIE_GRUNEISEN_POWER_SERIES
    RHO_0 = <real>density
    C_0 = <real>sound_speed
    K1 = <real>power_series_coeff1
    K2 = <real>power_series_coeff2
    K3 = <real>power_series_coeff3
    K4 = <real>power_series_coeff4
    K5 = <real>power_series_coeff5
    GAMMA_0 = <real>ambient_gruneisen_param
    POISSR = <real>poissons_ratio
    Y_0 = <real>yield strength
    PMIN = <real>mean_stress(REAL_MAX)
  END [PARAMETERS FOR MODEL MIE_GRUNEISEN_POWER_SERIES]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The Mie-Gruneisen power-series model describes the nonlinear pressure-volume (or equivalently pressure-density) response of solids or fluids in terms of a reference pressure-volume curve and deviations from the reference curve in energy space. The reference curve is taken to be the experimentally determined principal Hugoniot, which is the locus of end states that can be reached by a shock transition from the ambient state. The Mie-Gruneisen power-series model is very similar to the Mie-Gruneisen model, except that the Mie-Gruneisen model bases the Hugoniot pressure-volume response on the assumption of a linear shock-velocity-versus-particle-velocity relation, while the Mie-Gruneisen power-series model uses a power-series expression. For details about this model, see Reference [5].

For Mie-Gruneisen power-series energy-dependent materials, the Mie-Gruneisen power-series command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL MIE_GRUNEISEN_POWER_SERIES
```

and is terminated an input line of the following form:

```
END [PARAMETERS FOR MODEL MIE_GRUNEISEN_POWER_SERIES]
```

In the above command blocks:

- The `thermal strain option` is used to define thermal strains. See the Sierra/SolidMechanics 4.48 User's Guide for further information on defining and activating thermal strains.
- The ambient density, ρ_0, is defined with the `RHO_0` command line. The ambient density is the density at which the mean pressure is zero, not necessarily the initial density.
- The ambient bulk sound speed, c_0, is defined by the `C_0` command line.
- The power-series coefficients k_1, k_2, k_3, k_4, and k_5 are defined by the command lines `K1`, `K2`, `K3`, `K4`, and `K5`, respectively. Only the nonzero power-series coefficients need be input, since coefficients not specified will default to zero.
- The ambient gruneisen parameter, Γ_0, is defined by the `GAMMA_0` command line.
- Poisson's ratio, ν, is defined by the `POISSR` command line. Poisson's ratio is assumed constant.
- The yield strength, y_0, is defined by the `Y_0` command line. The yield strength is zero for the hydrodynamic case.
- The fracture stress is defined by the `PMIN` command line. The fracture stress is a mean stress or pressure, so it must be negative or zero. This is an optional parameter; if not specified, the parameter defaults to `REAL_MAX` (no fracture).

For information about the Mie-Gruneisen power-series model, consult Reference [5].

2.3.3 JWL (Jones-Wilkins-Lee) Model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  # thermal strain option
  THERMAL STRAIN FUNCTION = <string>thermal_strain_function
  # or all three of the following
  THERMAL STRAIN X FUNCTION =
    <string>thermal_strain_x_function
  THERMAL STRAIN Y FUNCTION =
    <string>thermal_strain_y_function
  THERMAL STRAIN Z FUNCTION =
    <string>thermal_strain_z_function
  #
  BEGIN PARAMETERS FOR MODEL JWL
    RHO_0 = <real>initial_density
    D = <real>detonation_velocity
    E_0 = <real>init_chem_energy
    A = <real>jwl_const_pressure1
    B = <real>jwl_const_pressure2
    R1 = <real>jwl_const_nondim1
    R2 = <real>jwl_const_nondim2
    OMEGA = <real>jwl_const_nondim3
    XDET = <real>x_detonation_point
    YDET = <real>y_detonation_point
    ZDET = <real>z_detonation_point
    TDET = <real>time_of_detonation
    B5 = <real>burn_width_const(2.5)
  END [PARAMETERS FOR MODEL JWL]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The JWL model describes the pressure-volume-energy response of the gaseous detonation products of HE (High Explosive). For details about this model, see Reference [5].

For JWL energy-dependent materials, the `JWL` command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL JWL
```

and is terminated an input line of the following form:

```
END [PARAMETERS FOR MODEL JWL]
```

In the above command blocks:

- The `thermal strain option` is used to define thermal strains. See the Sierra/SolidMechanics 4.48 User's Guide for further information on defining and activating thermal strains.

- The initial density of the unburned explosive, ρ_0, is given by the `RHO_0` command line.

- The detonation velocity, D, is given by the `D` command line.
- The initial chemical energy per unit mass in the explosive, E_0, is given by the `E_0` command line. Note, this value has NO effect on the actual behavior of the model in terms of stresses returned or energy generated. The `E_0` term effects only what initial energy value is printed in the output log file. The energy generated by the JWL material is determined by the `A, B, R1, R2, and D` constants. Most compilations of JWL parameters give E_0 in units of energy per unit volume, rather than energy per unit mass. Thus, the tabulated value must be divided by ρ_0, the initial density of the unburned explosive.
- The JWL constants with units of pressure, A and B, are given by the `A` and `B` command lines, respectively.
- The dimensionless JWL constants, R_1, R_2, and ω, are given by the `R1`, `R2`, and `OMEGA` command lines, respectively.
- The x-coordinate of the detonation point, x_D, is given by the `XDET` command line.
- The y-coordinate of the detonation point, y_D, is given by the `YDET command` line.
- The z-coordinate of the detonation point, z_D, is given by the `ZDET` command line.
- The time of detonation, t_D, is given by the `TDET` command line.
- The burn-width constant, B_5, is given by the `B5` command line. The burn-width constant has a default value of 2.5.

For information about the JWL model, consult Reference [5].

2.3.4 JWL (Jones-Wilkins-Lee) Model with multiple detonation points

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  # thermal strain option
  THERMAL STRAIN FUNCTION = <string>thermal_strain_function
  # or all three of the following
  THERMAL STRAIN X FUNCTION =
    <string>thermal_strain_x_function
  THERMAL STRAIN Y FUNCTION =
    <string>thermal_strain_y_function
  THERMAL STRAIN Z FUNCTION =
    <string>thermal_strain_z_function
  #
  BEGIN PARAMETERS FOR MODEL JWL_MULTIPOINT
    RHO_0 = <real>initial_density
    D = <real>detonation_velocity
    E_0 = <real>init_chem_energy
    A = <real>jwl_const_pressure1
    B = <real>jwl_const_pressure2
    R1 = <real>jwl_const_nondim1
    R2 = <real>jwl_const_nondim2
    OMEGA = <real>jwl_const_nondim3
    B5 = <real>burn_width_const(2.5)
    XDET = <real>x_detonation_point... (up to 100 values)
    YDET = <real>y_detonation_point... (up to 100 values)
    ZDET = <real>z_detonation_point... (up to 100 values)
    TDET = <real>time_of_detonation... (up to 100 values)
  END [PARAMETERS FOR MODEL JWL_MULTIPOINT]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The JWL MULTIPOINT model describes the pressure-volume-energy response of the gaseous detonation products of HE (High Explosive). The mechanics for this model is identical to the JWL model, but this version permits up to 100 detonation points. Each detonation point can have its own detonation time. For details about this model, see Reference [5].

For JWL MULTIPOINT energy-dependent materials, the `JWL_MULTIPOINT` command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL JWL_MULTIPOINT
```

and is terminated an input line of the following form:

```
END [PARAMETERS FOR MODEL JWL_MULTIPOINT]
```

In the above command blocks:

- The `thermal strain option` is used to define thermal strains. See the Sierra/SolidMechanics 4.48 User's Guide for further information on defining and activating thermal strains.

- The initial density of the unburned explosive, ρ_0, is given by the `RHO_0` command line.

- The detonation velocity, D, is given by the `D` command line.

- The initial chemical energy per unit mass in the explosive, E_0, is given by the `E_0` command line. Note, this value has NO effect on the actual behavior of the model in terms of stresses returned or energy generated. The `E_0` term effects only what initial energy value is printed in the output log file. The energy generated by the JWL MULTIPOINT material is determined by the `A, B, R1, R2, and D` constants. Most compilations of JWL parameters give E_0 in units of energy per unit volume, rather than energy per unit mass. Thus, the tabulated value must be divided by ρ_0, the initial density of the unburned explosive.

- The JWL constants with units of pressure, A and B, are given by the `A` and `B` command lines, respectively.

- The dimensionless JWL constants, R_1, R_2, and ω, are given by the `R1`, `R2`, and `OMEGA` command lines, respectively.

- The x-coordinates of the detonation points, x_D, are given by the `XDET` command line. Note that the number of detonation points specified should be the same number specified in the y and z coordinate locations as well as the detonation times.

- The y-coordinates of the detonation points, y_D, are given by the `YDET command` line. Note that the number of detonation points specified should be the same number specified in the x and z coordinate locations as well as the detonation times.

- The z-coordinates of the detonation points, z_D, are given by the `ZDET` command line. Note that the number of detonation points specified should be the same number specified in the x and y coordinate locations as well as the detonation times.

- The times of detonation for the detonation points, t_D, are given by the `TDET` command line. The detonation times can be different for each detonation point. Note that the number of detonation points specified should be the same number specified for the x, y, and z coordinate locations.

- The burn-width constant, B_5, is given by the `B5` command line. The burn-width constant has a default value of 2.5.

For information about the JWL model, consult Reference [5].

2.3.5 Ideal Gas Model

```
BEGIN PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name
  # thermal strain option
  THERMAL STRAIN FUNCTION = <string>thermal_strain_function
  # or all three of the following
  THERMAL STRAIN X FUNCTION =
    <string>thermal_strain_x_function
  THERMAL STRAIN Y FUNCTION =
    <string>thermal_strain_y_function
  THERMAL STRAIN Z FUNCTION =
    <string>thermal_strain_z_function
  #
  BEGIN PARAMETERS FOR MODEL IDEAL_GAS
    RHO_0 = <real>initial_density
    C_0 = <real>initial_sound_speed
    GAMMA = <real>ratio_specific_heats
  END [PARAMETERS FOR MODEL IDEAL_GAS]
END [PROPERTY SPECIFICATION FOR MATERIAL <string>mat_name]
```

The ideal gas model provides a material description based on the ideal gas law. For details about this model, see Reference [5].

For ideal gas materials, the ideal gas command block begins with the input line:

```
BEGIN PARAMETERS FOR MODEL IDEAL_GAS
```

and is terminated with an input line of the following form:

```
END [PARAMETERS FOR MODEL IDEAL_GAS]
```

In the above command blocks:

- The `thermal strain option` is used to define thermal strains. See the Sierra/SolidMechanics 4.48 User's Guide for further information on defining and activating thermal strains.
- The initial density, ρ_0, is given by the `RHO_0` command line.
- The initial sound speed, c_0, is given by the `C_0` command line.
- The ratio of specific heats, γ, is given by the `GAMMA` command line.

For information about the ideal gas model, consult Reference [5].

2.4 Energy Deposition

```
BEGIN PRESCRIBED ENERGY DEPOSITION
  #
  # block set commands
  BLOCK = <string_list>block_names
  INCLUDE ALL BLOCKS
  REMOVE BLOCK
  #
  # function commands
  T FUNCTION = <string>t_func_name
  X FUNCTION = <string>x_func_name
  Y FUNCTION = <string>y_func_name
  Z FUNCTION = <string>z_func_name
  #
  # input mesh command
  READ VARIABLE = <string>mesh_var_name
  #
  # user subroutine commands
  ELEMENT BLOCK SUBROUTINE = <string>subroutine_name
  # other user subroutine command lines
  SUBROUTINE DEBUGGING OFF | SUBROUTINE DEBUGGING ON
  SUBROUTINE REAL PARAMETER: <string>param_name
    = <real>param_value
  SUBROUTINE INTEGER PARAMETER: <string>param_name
    = <integer>param_value
  SUBROUTINE STRING PARAMETER: <string>param_name
    = <string>param_value
END [PRESCRIBED ENERGY DEPOSITION]
```

The `PRESCRIBED ENERGY DEPOSITION` command block applies a set quantity of energy to energy-dependent material models for a given set of element blocks. Energy deposition defines a specific energy deposited (energy per unit mass) the code computes the actual energy added to each element by multiplying this applied specific energy by the element mass.

Energy deposition represents a particular type of boundary condition, and thus this command block follows the general specification of command blocks used to specify boundary conditions. See the Sierra/SolidMechanics 4.48 User's Guide for more information on general boundary condition specification. The `PRESCRIBED ENERGY DEPOSITION` command block must appear in the region scope.

There are three options for defining the energy deposition for a set of elements: with standard SIERRA functions, with a mesh variable in the input mesh file, and by a user subroutine. If the energy deposition is a reasonably simple description and can be defined using the standard SIERRA functions, the function option is recommended. If the energy deposition requires a more complex description, it is necessary to use either the input mesh option or the user subroutine option. Only one of the three options can be specified in the command block.

The `PRESCRIBED ENERGY DEPOSITION` command block contains four groups of commands: block set, function, input mesh, and user subroutine. Each of these command groups, with the exception of the `T FUNCTION` command line, is basically independent of the others. Following are descriptions of the different command groups.

2.4.1 Block Set Commands

The `block set commands` portion of the `PRESCRIBED ENERGY DEPOSITION` command block defines a set of element blocks associated with the prescribed energy deposition and can include some combination of the following command lines:

```
BLOCK = <string_list>block_names
INCLUDE ALL BLOCKS
REMOVE BLOCK
```

These command lines, taken collectively, constitute a set of Boolean operators for constructing a set of blocks. See the Sierra/SolidMechanics 4.48 User's Guide for more information about the use of these command lines for creating a set of blocks used in the command block. Either the `BLOCK` command line or the `INCLUDE ALL BLOCKS` command line must be present in the command block.

2.4.2 Function Commands

If the function option is used, either the T function or a set of T, X, Y, and Z function command lines must be included in the command block.

Following are the command lines related to the function option:

```
T FUNCTION = <string>t_func_name
X FUNCTION = <string>x_func_name
Y FUNCTION = <string>y_func_name
Z FUNCTION = <string>z_func_name
```

Each of the above command lines references a function name (defined in the SIERRA scope in a `DEFINITION FOR FUNCTION` command block). All the functions referenced in these four command lines must appear in the SIERRA scope.

The `T FUNCTION` command line gives the name of the user-defined T function. The T function describes how the applied input energy dose is integrated over time t. The T function should be 0 at the start time and 1 at the time at which all energy is deposited. The T function must be monotonically increasing over the time it is defined. The T function describes the total percentage of energy that is deposited at a given time.

The `X FUNCTION`, `Y FUNCTION`, and `Z FUNCTION` command lines define three functions, which we will denote as X, Y, and Z, respectively. The X, Y, and Z functions describe the total amount of energy to be deposited in an element as a function of position. Suppose we have element A with centroid (A_x, A_y, and A_x) and mass M. The total energy that will have been deposited in element A at time t is given by:

$$E_A = M_A X(A_x) Y\left(A_y\right) Z(A_z) T(t), \tag{2.21}$$

where E_A is the total energy deposited.

2.4.3 Input Mesh Command

If the input mesh option is used, the quantity of specific energy deposited for each element will be read from an element variable defined in the mesh file.

Following is the command line related to the input mesh option:

```
READ VARIABLE = <string>mesh_var_name
```

The string `mesh_var_name` must match the name of an element variable in the mesh file that defines the energy deposition. Suppose that the total specific energy to be deposited for element A is $v(A)$. The quantity of energy deposited at time t is then given by:

$$E_A = M_A v(A) T(t). \tag{2.22}$$

The T function in Equation (2.22) is the same as that described in Section 2.4.2.

2.4.4 User Subroutine Commands

The user subroutine option allows for a very general description of the energy deposition, but this option requires that you write a user subroutine to implement this capability. The subroutine will be called by `adagio` at the appropriate time to generate the energy deposition.

Energy deposition uses an element subroutine signature. The subroutine returns one value per element for all the elements selected by use of the `block set commands`. The returned value is the specific energy flux at an element at a given time. The output flags array is ignored. The total energy deposited in an element is found by a time integration of the returned subroutine specific energy fluxes times the element mass. See the Sierra/SolidMechanics 4.48 User's Guide for more information about user subroutines.

Following are the command lines related to the user subroutine option:

```
ELEMENT BLOCK SUBROUTINE = <string>subroutine_name
SUBROUTINE DEBUGGING OFF | SUBROUTINE DEBUGGING ON
SUBROUTINE REAL PARAMETER: <string>param_name
  = <real>param_value
SUBROUTINE INTEGER PARAMETER: <string>param_name
  = <integer>param_value
SUBROUTINE STRING PARAMETER: <string>param_name
  = <string>param_value
```

The user subroutine option is invoked by using the `ELEMENT BLOCK SUBROUTINE` command line. The string `subroutine_name` is the name of a FORTRAN subroutine written by the user. The other command lines listed here (SUBROUTINE DEBUGGING OFF, SUBROUTINE DEBUGGING ON, SUBROUTINE REAL PARAMETER, SUBROUTINE INTEGER PARAMETER, and SUBROUTINE STRING PARAMETER) are described in the Sierra/SolidMechanics 4.48 User's Guide.

2.4.5 Output Variables

When using prescribed energy deposition a few output variables become available.

- specific_internal_energy is an element variable that is the energy per unit mass that was applied by the boundary condition.
- deposited_internal_energy is the actual energy deposited by the boundary condition. This value is specific_internal_energy times the element mass.

References

[1] G.R. Johnson and S.R. Beissel. Modular material model subroutines for explicit lagrangian computer codes. Technical Report ARL-CR-556, Network Computing Services, Inc., Minneapolis, MN, February 2005.

[2] W.M. Scherzinger and D.C. Hammerand. Constitutive models in LAME. Technical Report SAND2007-5873, Sandia National Laboratories, Albuquerque, NM, September 2007. pdf.

[3] T. J. Holmquist G.R. Johnson and S.R. Beissel. Response of aluminum nitride (including a phase change to large strains. *Journal Applied Physics*, 94, 2003.

[4] A. Picklesimer. The joint DoD/DoE munitions technology program, progress report for FY01, dynamic properties of materials. Technical Report LA-14015-PR, Los Alamos National Laboratory, February 2003.

[5] J.W. Swegle. SIERRA: PRESTO theory documentation: Energy dependent materials version 1.0. Technical Report SAND2009-3801P, Sandia National Laboratories, Albuquerque, NM, October 2001.

[6] E.S. Hertel Jr. and G.I. Kerley. CTH reference manual: The equation of state package. Technical Report SAND98-0947, Sandia National Laboratories, Albuquerque, NM, 1998.

[7] S.P. Lyon and J.D. Johnson. SESAME: The Los Alamos National Laboratory equation of state database. Technical Report LA-UR-92-3407, Los Alamos National Laboratory, 1992.

[8] S.A. Silling. CTH reference manual: Viscoplastic models. Technical Report SAND91-0292, Sandia National Laboratories, Albuquerque, NM, 1991.

[9] P.A. Taylor. CTH reference manual: The Steinberg-Guinan-Lund viscoplastic model. Technical Report SAND92-0716, Sandia National Laboratories, Albuquerque, NM, 1992.

[10] S. Silling. Use of the Johnson-Cook fracture model in CTH. Technical Report Memo, Sandia National Laboratories, Albuquerque, NM, 1996.

Chapter 3

Elements

This chapter describes additional information in the elements that are relevant to the energy-dependent material models described in this document. General information about the elements used in the SIERRA Solid Mechanics codes can be found in the Sierra/SolidMechanics 4.48 User's Guide.

3.1 Finite Element Model

```
BEGIN FINITE ELEMENT MODEL <string>mesh_descriptor
  ...
  BEGIN PARAMETERS FOR BLOCK [<string list>block_names]


  END [PARAMETERS FOR BLOCK <string list>block_names]
END [FINITE ELEMENT MODEL <string>mesh_descriptor]
```

Not all of the elements in the SIERRA Solid Mechanics codes support energy-dependent material models. The status of the current element library with respect to these materials is as follows:

- Eight-node, uniform-gradient hexahedron: Only the midpoint-increment formulation [1] supports equation of state (EOS) models. This element is the most heavily tested with EOS models, and is the one currently recommended for use in this regime. This element is the only one that currently supports the use of CTH material models.

- Eight-node, selective-deviatoric hexahedron: this element cannot be used with the EOS models.

- Four-node tetrahedron: The regular element-based formulation and the node-based formulation for the four-node tetrahedron both support EOS models (but not the CTH models). However, both of these element formulations have problems with EOS models. The regular 4-node tetrahedral element is subject to volumetric and shear locking, which can lead to erroneous results. Recent investigations using the node-based tetrahedral element have shown problems with the computation of pressure, which is vital to the EOS computations. Thus it is recommended to avoid using either of these elements with EOS materials. Remeshing with the node-based tetrahedral further disturbs the quality of the solution, so much so that remeshing has been de-activated for EOS materials.

- Eight-node and ten-node tetrahedron: These elements support limited use of the EOS material models at this time (but not the CTH models).

- Smoothed particle hydrodynamics (SPH) elements: These are one-dimensional elements. These elements can be used with EOS models (but not the CTH models). These elements are subject to numerical (non physics-based) failure in tension for all materials, so should be used with caution. Some analyses using SPH for explosives have shown marked deviation from expected behavior, so close comparisons should be made to other approaches, such as using uniform-gradient hexahedral elements.

- None of the structural elements (membranes, shells, beams, trusses, or dampers) currently operate with EOS materials.

3.1.1 Descriptors of Element Blocks

```
BEGIN PARAMETERS FOR BLOCK [<string list>block_names]
  LINEAR BULK VISCOSITY =
    <real>linear_bulk_viscosity_value(0.06)
  QUADRATIC BULK VISCOSITY =
    <real>quad_bulk_viscosity_value(1.20)
  MAX ENERGY ITERATIONS = <integer>max_energy_iterations(1)
  ENERGY ITERATION TOLERANCE = <real>energy_iteration_tolerance(1.0e-5)
END [PARAMETERS FOR BLOCK <string list>block_names]
```

The finite element model consists of one or more element blocks. Associated with an element block or group of element blocks will be a `PARAMETERS FOR BLOCK` command block, which is also referred to in this document as an *element-block command block*. The basic information about the element blocks (number of elements, topology, connectivity, etc.) is contained in a mesh file. Specific attributes for an element block must be specified in the input file. The general commands for this block are described in the Sierra/SolidMechanics 4.48 User's Guide, but several commands are of particular use when employing EOS models.

3.1.1.1 Linear and Quadratic Bulk Viscosity

```
LINEAR BULK VISCOSITY =
  <real>linear_bulk_viscosity_value(0.06)
QUADRATIC BULK VISCOSITY =
  <real>quad_bulk_viscosity_value(1.20)
```

The linear and quadratic bulk viscosity are set with these two command lines. These terms assist with the handling of strong discontinuities in stress, such as those found in a shock front. Setting these parameters to a level that is too low will cause the simulation to exhibit excess noise ("ringing") in the simulation. Setting these too large, however, can cause excessive smearing of the discontinuity.

For more information, consult the documentation for the elements [2] for a description of the bulk viscosity parameters.

3.1.1.2 Energy Iterations

```
MAX ENERGY ITERATIONS =
  <integer>max_energy_iterations(1)
ENERGY ITERATION TOLERANCE =
  <real>energy_iteration_tolerance(1.0e-5)
```

When using an energy-dependent material model, the internal energy is updated using a second-order, implicit equation that includes terms for pressure-volume, entropy, and deposited work. The pressure-volume work is broken into hydrodynamic and deviatoric parts. Historically, Presto_ITAR has solved this equation under the assumption of a Mie-Gruneisen material, where the pressure is linearly dependent upon the internal energy, see Reference [3]. Sections 2.3 and 2.1

contain examples of materials which use this assumption. In these models, the energy equation is solved explicitly inside of the material model itself. However, the recent addition of more general material models (see Section 2.2) resulted in the need to remove this dependency. Also, for general portability of material models, the energy update was extracted from the material models and placed into the element for these general models.

Due to the implicit nature of the energy equation used by Presto_ITAR, an iteration is required to make the new state self-consistent. The `MAX ENERGY ITERATIONS` setting controls the maximum number of iterations performed in the self-consistent loop. When using a legacy material model, or a model from Section 2.2 that is purely hydrodynamic, the default value of `1` is sufficient. For the former models, this recovers the legacy behavior. In the case of the latter models, an isentropic predictor method is used that allows for an explicit solution of the implicit energy equation. For more general models from Section 2.2, one should set `MAX ENERGY ITERATIONS` to a value of at least `2`. This provides for a minimal amount of convergence in the energy equation.

The convergence criteria for exiting the self-consistent loop which calculates the implicit energy update may be set via the command `ENERGY ITERATION TOLERANCE`. For planar shock problems, the default value is typically reached after two or three iterations. Convergence to full double precision tolerance typically takes up to six or seven iterations. A warning message will be printed if the self consistent loop fails to converge to the desired tolerance within the maximum allowed number of iterations.

3.2 Element Sections

Element sections are defined by section command blocks. There are currently nine different types of section command blocks. The section command blocks appear in the SIERRA scope, at the same level as the FINITE ELEMENT MODEL command block. No special parameters in the sections are required for the use of EOS models. However, there are some inputs in the SPH section that can be useful for explosives computations. The relevant section from the standard user's guides is duplicated here, with a few additional comments.

3.2.1 SPH Section

```
BEGIN SPH SECTION <string>sph_section_name
  DENSITY FORMULATION = <string>MATERIAL|KERNEL(MATERIAL)
END [SPH SECTION <string>sph_section_name]
```

SPH (smoothed particle hydrodynamics) is useful for modeling fluids or for modeling materials that undergo extremely large distortions. One must be careful when using SPH for modeling. SPH tends to exhibit both accuracy and stability problems, particularly in tension. An SPH particle interacts with other nearest-neighbor SPH particles based on radius properties of all the elements involved; SPH particles react with other elements, such as tetrahedra, hexahedra, and shells, through contact. You should consult Reference [4] regarding the theoretical background for SPH. The full set of commands for the SPH section are listed in the SIERRA Solid Mechanics user's guides.

The DENSITY FORMULATION command can be used to define the way in which the particle radii are updated. For the default option MATERIAL the material densities and nodal masses are used to compute a volume associated with a particle at a given time. The radius is then updated to be the cube root of that volume. The alternative option KERNEL computes the particle densities based off of the SPH particles masses and the SPH kernel density function. The KERNEL option may be necessary if large expansion of particles is expected (for example, modeling large density changes in gases). The MATERIAL option is generally changes particle densities and thus radii less than the KERNEL option so is appropriate for analysis that do not have large density fluctuations. The KERNEL option is often necessary for EOS models for explosives (such as JWL) or for shocks in gaseous materials.

3.3 Remeshing

```
BEGIN REMESH
   #
   # Inputs to control remeshing
   #
END [REMESH]
```

The REMESH command block, which is used within the region scope, sets parameters for remeshing a portion of the mesh. Remeshing involves removing badly shaped elements and inserting new elements of better quality that occupy the same volume. Depending on the degree to which the original elements are deformed, the new elements may occupy slightly more or slightly less volume than the original mesh. If regions of the mesh cannot be meshed with well-shaped elements having reasonable time steps, they may be removed entirely, potentially changing the topology. Examples of such regions include exterior slivers or very thin parts.

Due to problems with EOS materials and node-based tetrahedrons, remeshing has been deactivated for EOS materials.

References

[1] L.M. Taylor and D.P. Flanagan. Pronto3D: A three-dimensional transient solid dynamics program. Technical Report SAND87-1912, Sandia National Laboratories, Albuquerque, NM, March 1989. pdf.

[2] T.A. Laursen, S.W. Attaway, and R.I. Zadoks. SEACAS theory manuals: Part III. finite element analysis in nonlinear solid mechanics. Technical Report SAND98-1760/3, Sandia National Laboratories, Albuquerque, NM, 1999. pdf.

[3] J.W. Swegle. SIERRA: PRESTO theory documentation: Energy dependent materials version 1.0. Technical Report SAND2009-3801P, Sandia National Laboratories, Albuquerque, NM, October 2001.

[4] J.W. Swegle, S.W. Attaway, M.W. Heinstein, F.J. Mello, and D.L. Hicks. An analysis of smoothed particle hydrodynamics. Technical Report SAND93-2513, Sandia National Laboratories, Albuquerque, NM, March 1994. pdf.

Chapter 4

Fortissimo

This chapter documents a code coupling capability known as "Fortissimo" that is currently only available in the ITAR versions of the codes. Fortissimo is a two-way coupled CTH and Sierra/SM capability that is available via the executable named "fortissimo". This capability couples a Lagrangian explicit dynamics region run in Sierra/SM with an Eulerian shock physics region run in CTH. Fortissimo couples these codes by inserting the solid material from Sierra/SM into CTH at each time step and returning surface pressures to Sierra/SM. The fortissimo capability was primarily designed to model air-blast loading on structures. Most features of Sierra/SM and CTH can be used with Fortissimo, including element death, CTH EOS material models, Sierra/SM rigid bodies, etc.

4.1 Coupling Algorithm Description

Using Fortissimo, the Lagrangian material enters the CTH region via insertion of incrementally immovable and rigid volume into CTH cells which overlap the Sierra/SM mesh. For solid elements like hexahedrons, calculation of the rigid volume is straightforward. It is based on the intersection of the solid element with the CTH cell. For shell elements, this is more challenging because it requires the lofting of the shells to create volume for the elements. The process for shells is illustrated in Section 4.7.

It is important to note that Fortissimo assumes rigid material for the CTH material insertion step. Although the solid material in Sierra/SM may or may not be rigid, for the CTH portion of the calculation it is assumed to be incrementally rigid, i.e. it doesn't move in CTH between Sierra/SM time steps but its position is updated at the beginning of each Sierra/SM time step. This assumption has two important effects.

First, the rigid material insertion implicitly assumes that the CTH material is much less stiff and massive than the Sierra/SM material. An obvious application that fits this assumption is air-blast on structure, which was the main application of interest during Fortissimo development. Other applications that fit this assumption may also be good candidates for modeling with Fortissimo.

Second, Fortissimo relies on CTH's rigid material modeling capabilities. Due to internal CTH features and assumptions, rigid material boundary conditions are not strictly enforced on surfaces of rigid material (e.g. this allows proper modeling of sliding contact between rigid materials). A result is that any rigid material that is less than three CTH cells thick may not be fully enforced in CTH and might allow non-rigid material to flow through it. To properly model rigid material, and by extension any Sierra/SM Lagrangian material in a Fortissimo simulation, there must be at least three, and preferably four or more, CTH cells through the minimum thickness of the material. Paying attention to this is especially important when modeling shells (see Sections 4.3 and 4.7).

Fortissimo completes the coupling of CTH back to Sierra/SM through surface pressures updated at each Sierra/SM time step. This places some restrictions on Fortissimo modeling, the clearest of which is that viscosity of CTH materials is not taken into account. Since only pressure is applied to Sierra/SM materials, and not traction, any viscous effects from CTH to Sierra/SM are ignored.

4.2 Running Coupled Analysis

Coupled runs must be handled with the “fortissimo” executable using a command line such as:

```
sierra fortissimo -i sierra_input.i
```

The `sierra_input.i` defines the input deck of the Lagrangian Sierra/SM portion of the model. In order to run Fortissimo the Sierra/SM input file will contain a specialized pressure boundary condition. This pressure boundary condition will handle the coupling of the codes by sending Lagrangian geometry to the CTH region and passing back a CTH pressure field to the Lagrangian region. This specialized pressure boundary condition will also specify the name of a CTH input deck that defines the CTH region.

Prior to running a fortissimo analysis it is recommended to load an appropriate CTH module in order for the CTH portion to have access to spyplot routines and material libraries. It is currently recommended to load the CTH module `cth-user-current`.

4.3 Command Syntax

CTH-Sierra/SM coupling is invoked by the following specialized pressure boundary condition syntax.

```
BEGIN PRESSURE
  SURFACE = <string>surface_names
  OBJECT TYPE = FACE
  FIELD VARIABLE = pressure_to_apply
  CTH INPUT            = <string>cth_input_deck
    IDRUN              = <string>cth_suffix
    COORDINATE SCALING = <real>coordScale(1.0)
    PRESSURE SCALING   = <real>pressScale(1.0)
  INSERTION ALGORITHM  = SURFACE|VOLUME (VOLUME)
  ARTIFICIAL SHELL THICKNESS = <real>shellThickVal
  TANGENTIAL SHELL SCALE = <real>shellTangvVal(0.1)
END PRESSURE
```

Generally, the entire Lagrangian domain is inserted into the CTH region. CTH based pressures will only be applied back to faces that are present in the side sets listed in the `SURFACE` command line. If it is desired for the CTH pressures to be applied to the entire exterior surface of the Sierra/SM domain the special surface name `cth_presto_interface_skin` must be used.

The `OBJECT TYPE = FACE` and `FIELD VARIABLE = pressure_to_apply` commands should always be used verbatim to inform the pressure boundary condition to read pressure values from the face based field `pressure_to_apply`. The face based `pressure_to_apply` field may be viewed on the Sierra/SM output database.

The `CTH_INPUT` defines parameters for the CTH portion of the run. The full path from the run directory to the CTH input deck should be given by `cth_input_deck`. The `cth_suffix` string defines a suffix (such as rsct# or osct#) that cth will append to the output files it generates.

Potentially CTH and Sierra/SM may be run with different base units. Scaling factors are available to correct differences in the unit system. The `coordScale` value defines how Sierra/SM nodal coordinates will be scaled prior to insertion of Lagrangian Sierra/SM volume in CTH. The `pressScale` value will be applied to pressures coming back from CTH prior to application of those pressures on the Sierra/SM Lagrangian region.

The `INSERTION ALGORITHM` command selects between two methodologies to insert rigid material from Sierra/SM into CTH. Both algorithms give nominally similar behavior. The `VOLUME` option is currently recommended due to better robustness and performance. Additionally, the `VOLUME` algorithm is able insert shell material while the `SURFACE` algorithm cannot.

The `ARTIFICIAL SHELL THICKNESS` command can be used to artificially change how thick shells will be for insertion into the CTH domain. In order for CTH to properly process shell elements and prevent the passing of material through a shell surface, the shell should be at least three CTH cells thick. If the actual shell thickness is less than this thickness value, the `ARTIFICIAL SHELL THICKNESS` can be used to increase the thickness for the purposes of insertion into CTH only (this command does not affect the thickness of the shells in Sierra/SM). The value specified by this command affects all shell elements in the model.

The `TANGENTIAL SHELL SCALE` command can be used to set the artificial shell tangential value. In order to smooth out discontinuities near the intersection of shell objects it is usually desirable to insert shell volumes slightly wider than the physical shells, thus this parameter has a default value of 0.1. The value specified by this command affects all shell elements in the model. For more details see Section 4.7.

In addition to the specific CTH coupling commands above most other commands that could be present in a standard Sierra/SM pressure boundary condition are also available.

4.4 Restart in Fortissimo

Coupled restart of Presto and CTH requires writing and reading of coordinated restart images to each code's respective restart files. The writing of restart files is scheduled through restart commands in Presto's input file (i.e., Presto's "`begin restart`" command block. When Presto performs a time step in which a restart image is written, the coupling code instructs CTH to also write its own restart image. The two restart images are written into standard Presto and CTH restart files, and no additional restart data is needed for the coupling code. The Presto and CTH restart images will not have the same simulation time due to the "leap frog" loose coupling algorithm.

There are two ways o properly restart fortissimo:

4.4.1 Specifying Restart at a Specific Time

The simulation time-tags of each written Presto and CTH restart images are noted in the Presto log file as follows.

```
COUPLED Presto + CTH WRITING RESTART

  Presto RESTART TIME TAG = <time for presto>
  CTH    RESTART TIME TAG = <time for cth>
```

To restart Fortissimo at a specific time, the respective read-restart commands must be entered into the presto and cth input files for the analysis. These restart commands must specify each code's restart simulation times as noted in Presto's log file. For Presto this is:

```
  RESTART TIME = <time for presto>

  begin restart data
   database name = sierra.rsout
   at time <time to start writing restart> interval = <increment to write restart>
  end
```

and for CTH this is:

```
restart
  time = <time for cth>
endr
```

4.4.2 Automatic Restart

When specifying automatic restart in Fortissimo, both codes find and then restart from the most recently stored restart time. This is accomplished from the following commands:

In the Sierra/SM input deck:

```
restart = auto

begin restart data
 database name = sierra.rsout
 at time 0.0 interval = <increment to write restart>
end
```

In the CTH input deck:

```
restt
  time = 0, dt = <increment to write restart>
endrestart
```

Note, as Sierra and CTH results generally do not line up exactly, some small discrepancy in the time stepping scheme may occur during restart leading to slight differences between restarted and non-restarted runs.

4.5 CTH Input Deck

No additional Sierra/SM commands need to be specified in the CTH input file. Users are directed towards the CTH users manual for information on properly setting up a CTH input file, Reference [1].

4.6 Hexahedron Model Example

Figure 4.1 A) and B) show an example of a coupled calculation between Sierra/SM and CTH using Fortissimo. A solid cube made up of deformable hexahedron elements in Sierra/SM is placed inside a 3-D CTH mesh, and a blast load is set off at position (0.0, 0.0, -7.5). The progression of the calculation is given in Figure 4.2 with pressure from the blast shown in color being deflected around the cube as well as reflected off the side facing the blast. Using Fortissimo, the Lagrangian material enters the CTH region via insertion of rigid volume into CTH cells which overlap the Sierra mesh. This insertion is straightforward for solid elements like hexahedrons, while for shell elements it is more challenging because it requires the lofting of the shells to create volume in the elements. An example using shell elements is given below in Section 4.7.

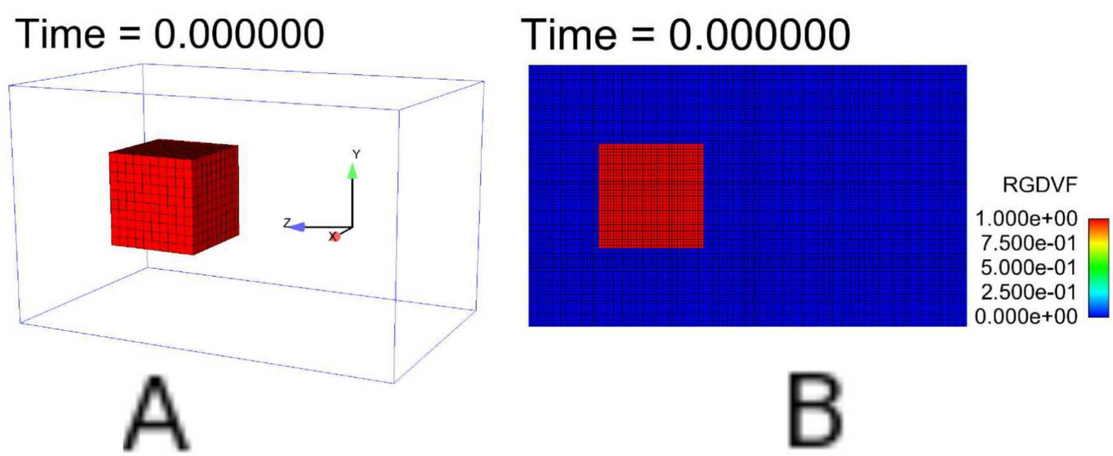

Figure 4.1: A) Cube Mesh in Wire Frame of CTH Mesh B) Volume Fraction of Cube Mesh in Fully Gridded CTH Mesh

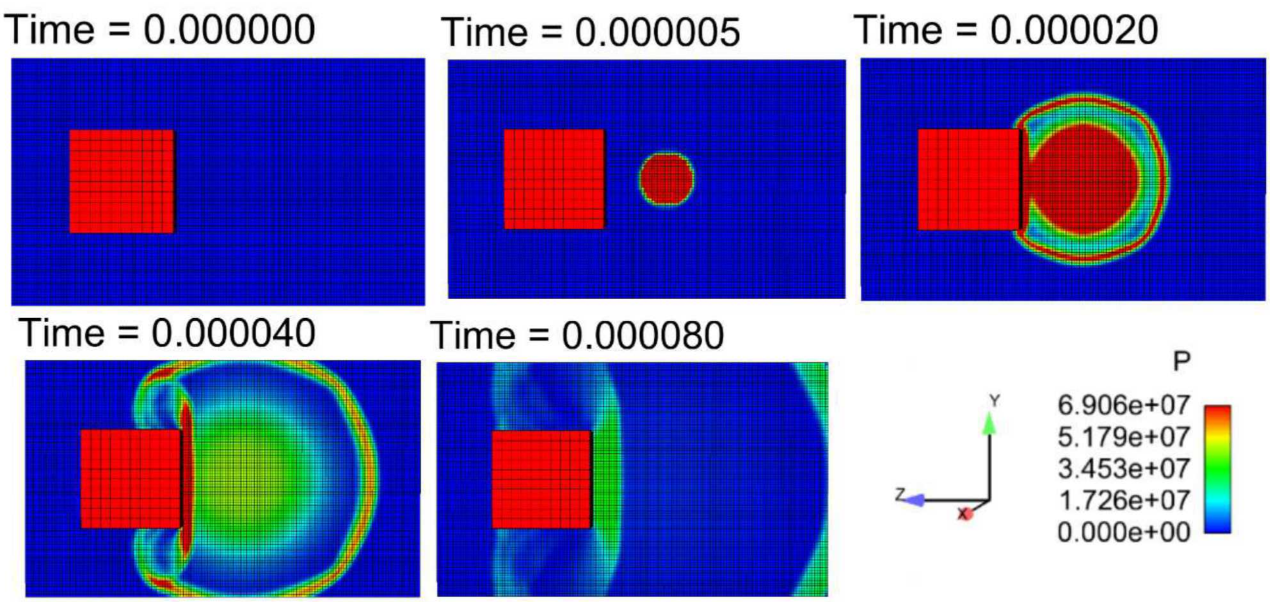

Figure 4.2: Example of Sierra Mesh Made up of Hex Elements inside CTH Mesh

4.7 Shell Model Example

An example using shells is shown in Figure 4.3 A) and B). This problem can be found in the Sierra regression test suite at fortissimo_rtest/cylinder-shell/. It should be noted that the termination time is slightly adjusted for this documentation. The Sierra/SM mesh is comprised of a hollow cylinder made up of shell elements with a radius of 3 and a height of 12 centered at the origin of the CTH region. The CTH mesh is a rectangular 3-D grid whose dimensions span $-5 \le X \le 5$, $-5 \le Y \le 5$, and $-20 \le Z \le 20$. A blast load is placed at (0.0, 0.0, -7.5).

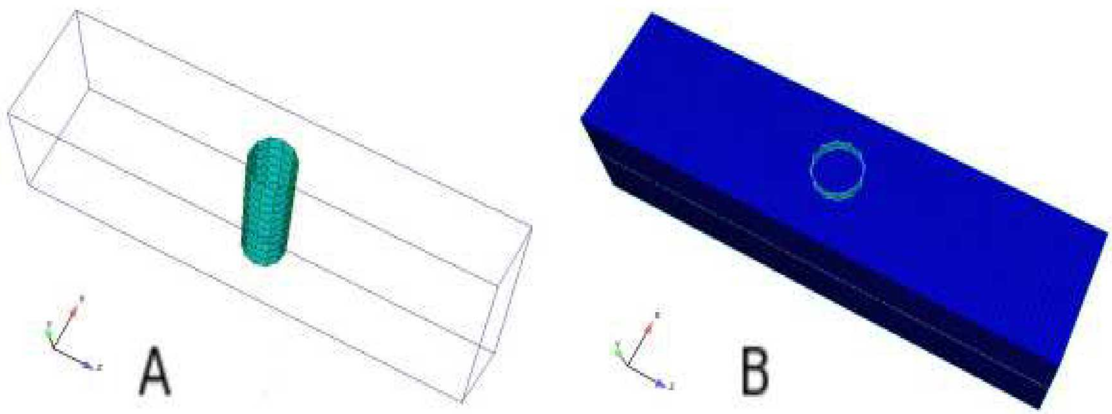

Figure 4.3: A) Cylindrical Shell Mesh in Wire Frame of CTH Mesh B) Cylindrical Shell Mesh in Fully Gridded CTH Mesh.

As explained in 4.3, the creation of CTH rigid volume material is based on either the actual shell thickness or, if it is set, the artificial thickness specified by `ARTIFICIAL SHELL THICKNESS`. For example, CTH would see a shell thickness of 1.0e-3 based on the current example's shell section:

```
  BEGIN SHELL SECTION FRONT_SHELL
    THICKNESS = 1.0e-3
    lofting factor = 1.0
  END SHELL SECTION FRONT_SHELL
```

However, to allow larger CTH cells to be used the solid material insertion into CTH is modified by the artificial shell thickness, set to 1.0, in the pressure boundary condition:

```
BEGIN PRESSURE
  ...
  ARTIFICIAL SHELL THICKNESS = 1.0
  ...
END PRESSURE
```

In addition, in the current example the shells are lofted outward from the center of the cylinder. If the `lofting factor` were set to 0.0, lofting would be done inward towards the center of the cylinder. Figure 4.4 shows a diagram of the lofting and how it appears numerically. The colors represent the volume fraction of the CTH cell that is treated as rigid.

As can be seen, the shells are enlarged slightly tangentially, which is controlled by `TANGENTIAL SHELL SCALE` with a default setting of 0.1. The motive for this is to prevent gaps that could form between the edges of the lofted shells leading to cell pinching where a CTH cell lies inside the gap. This could lead to infinite pressure in that gap. To overcome this, the shells are expanded to a slightly larger size to close these gaps (see Figure 4.5).

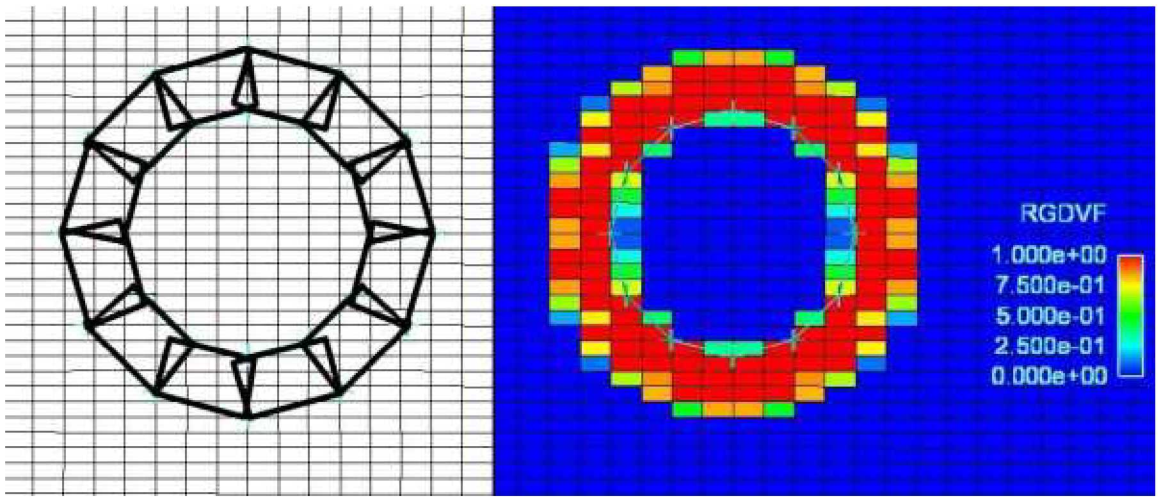

Figure 4.4: Shell Lofting

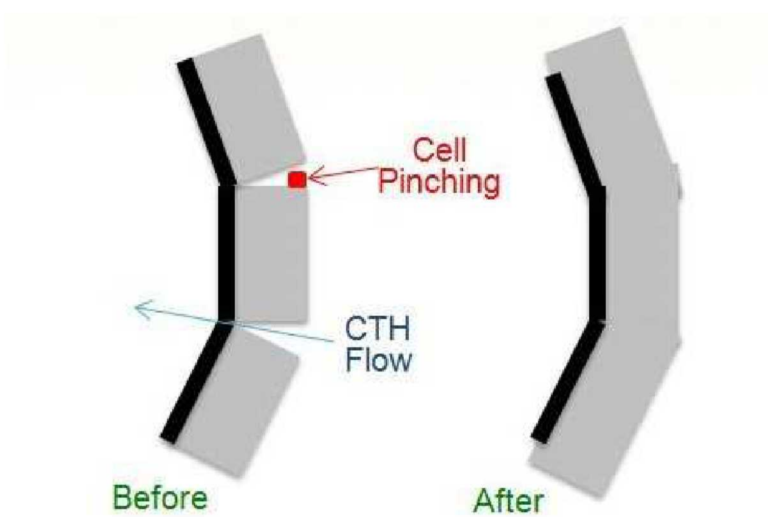

Figure 4.5: Tangential Shell Enlargement During Lofting

As the calculation runs, CTH returns pressure to the Lagrangian region by defining pressure values on the faces defined in the pressure boundary condition. In Figures 4.6 to 4.11, 6 moments in the calculation are recorded as seen from the ZX plane. Figure 4.8 shows parts of the pressure wave

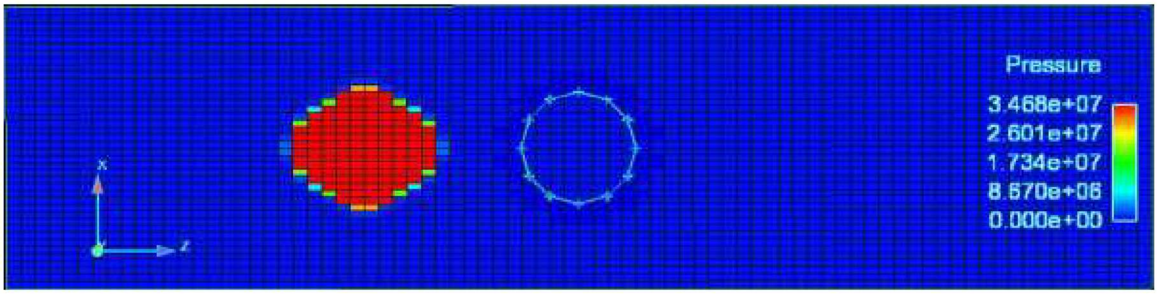

Figure 4.6: Step 22 out of 720; time = 1.96351e-06

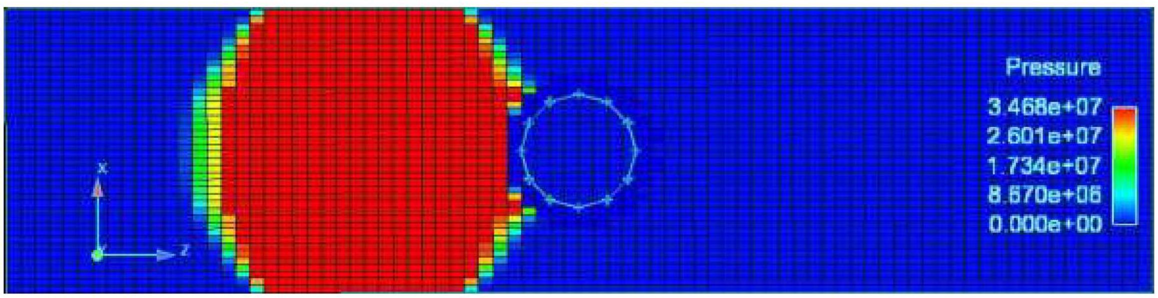

Figure 4.7: Step 116 out of 720; time = 1.0143e-05

going around the cylinder. As can be seen in Figure 4.9, a large amount of pressure is reflected back from the surface of the shell directly facing the blast.

Both the Lagrangian Sierra/SM model and the CTH region will compute their own stable critical time steps. If the CTH region has a time step greater than or equal to the Sierra/SM time step both regions will move forward using the Sierra/SM time step. If the CTH timestep is smaller than the Sierra/SM time step the CTH region may subcycle and perform several smaller CTH time steps within each Sierra/SM time step.

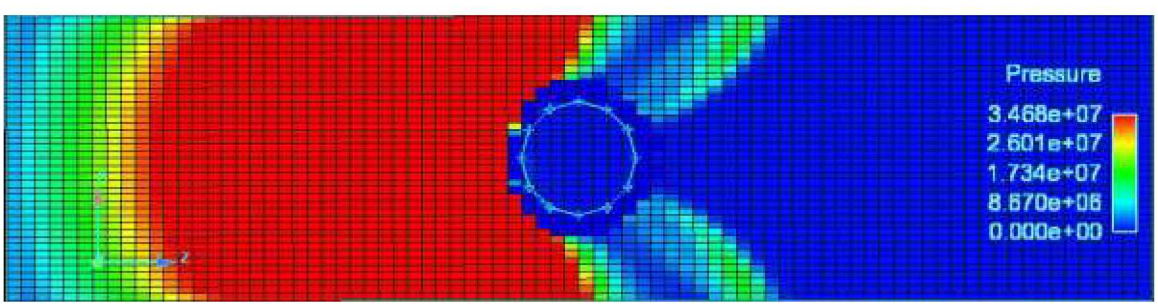

Figure 4.8: Step 320 out of 720; time = 3.85e-05

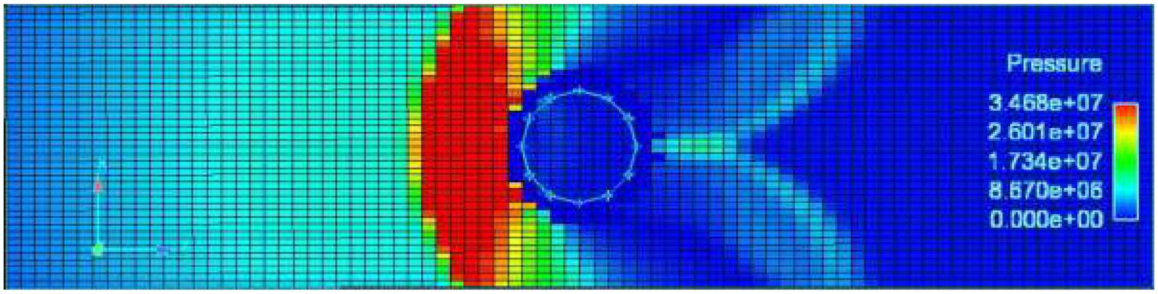

Figure 4.9: Step 468 out of 720; time = 6.05e-05

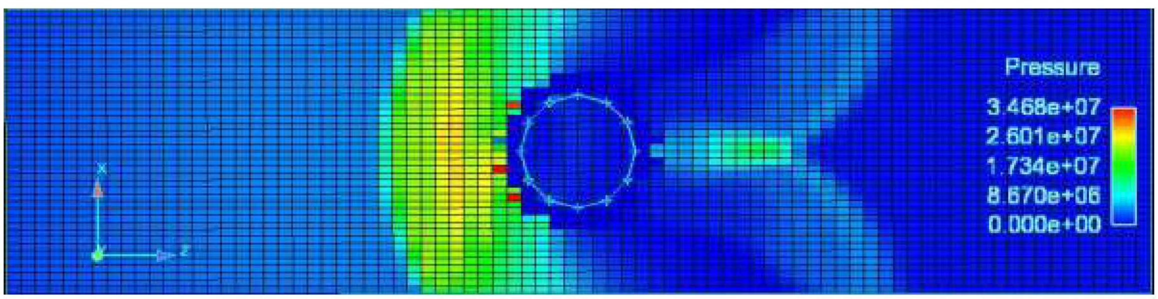

Figure 4.10: Step 568 out of 720; time = 7.15e-05

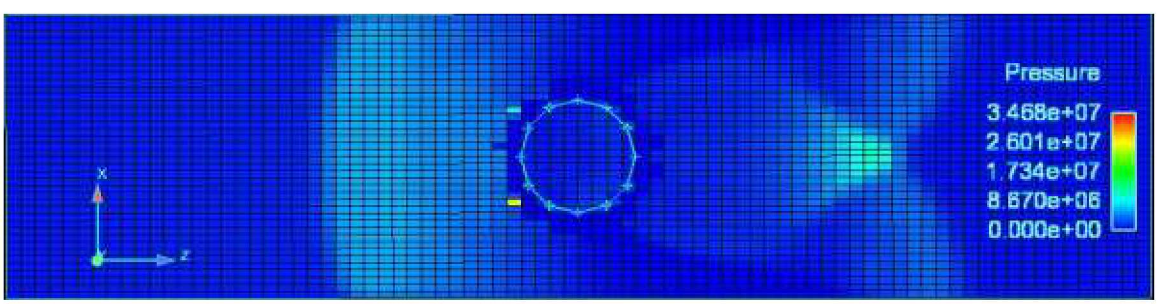

Figure 4.11: Step 720 out of 720; time = 9.80254e-05

4.8 Usage Guidelines and Restrictions

CTH/SierraSM coupling is currently incompatible with some capabilities including:

- Explicit dynamics multi-region time step subcycling within the Lagrangian region.
- Line and particle elements (truss, beams, SPH, etc.)
- Solution control, for example to perform coupled thermo-structural analysis.
- Multiple CTH regions used simultaneously
- Insertion of rigid material into the CTH domain (other than the rigid material automatically inserted via the code coupler)
- Analytic objects such as analytic contact planes defined in the Sierra/SM region.

If either the CTH run or the Sierra/SM portion of the run encounters a fatal error the analysis will abort. If CTH reaches its stop time prior to the the Sierra/SM analysis termination time Sierra/SM will continue running and applying no further CTH force. If the Sierra/SM analysis reaches its termination time both the CTH analysis and Sierra/SM analysis will stop.

References

[1] D.A. Crawford, A.L. Brundage, E.N. Harstad, E.S. Hertel Jr., R.G. Schmitt, S.C. Schumacher, and J.S. Simmons. *CTH User's Manual and Input Instructions, Version 10.0*. Sandia National Laboratories, 2011.

Chapter 5

Boundary Conditions

This chapter documents a specialized pressure boundary condition that is currently only available in the Presto_ITAR version of the Sierra/SM code. Refer to the Sierra/SM user's guide for documentation of other boundary conditions.

5.1 Blast Pressure

```
BEGIN BLAST PRESSURE <string>name
  SURFACE = <string list>surface_ids
  REMOVE SURFACE = <string list>surface_id
  BLOCK = <string list>block_ids
  REMOVE BLOCK = <string list>block_ids
  INCLUDE ALL BLOCKS

  BURST TYPE = <string>SURFACE|AIR
  TNT MASS IN LBS = <real>tnt_mass_lbs
  BLAST TIME = <real>blast_time
  BLAST LOCATION = <real>loc_x <real>loc_y <real>loc_z
  ATMOSPHERIC PRESSURE IN PSI = <real>atmospheric_press
  AMBIENT TEMPERATURE IN FAHRENHEIT = <real>temperature
  FEET PER MODEL UNITS = <real>feet
  MILLISECONDS PER MODEL UNITS = <real>milliseconds
  PSI PER MODEL UNITS = <real>psi
  PRESSURE SCALE FACTOR = <real>pressure_scale(1.0)
  IMPULSE SCALE FACTOR = <real>impulse_scale(1.0)
  POSITIVE DURATION SCALE FACTOR = <real>duration_scale(1.0)
  ACTIVE PERIODS = <string list>period_names
  INACTIVE PERIODS = <string list>period_names
  BLOCKING SURFACE CALCULATION = OFF|ON(OFF)
END [BLAST PRESSURE <string>name]
```

The `BLAST PRESSURE` command block is used to apply a pressure load resulting from a conventional explosive blast. This boundary condition is based on Reference [1] and Reference [2], and Sachs scaling is implemented to match ConWep (Reference [3]).

Warning: The data that `BLAST PRESSURE` utilizes has been updated to match data from ConWep 2.1.0.8 and no longer matches the curves reported in Reference [1] or Reference [2].

Angle of incidence is accounted for by transitioning from reflected pressure to incident pressure according to:

$$P_{total} = P_{ref} * \cos\theta + P_{inc} * (1 - \cos\theta) \tag{5.1}$$

where θ is the angle between the face normal vector and the direction to the blast from the face, P_{total} is the total pressure, P_{ref} is the reflected portion of the pressure, and P_{inc} is the incident portion of the pressure. P_{ref} and P_{inc} are based on Friedlander's equation, as described in Reference [2].

The `BLAST PRESSURE` command block can be used for surfaces that have faces derived from solid elements (eight-node hexahedra, four-node tetrahedra, eight-node tetrahedra, etc.), membranes, and shells.

The `BLAST PRESSURE` command block can also be used for particle like elements if the particle elements are created through the use of element death particle conversion. The surfaces must be defined on the original solid elements.

If θ is greater than 90 degrees (i.e. the face is pointing away from the blast), only P_{inc} is applied to the face. In this case, the face variable `cosa`, which contains cosθ, is set to zero.

This boundary condition is applied to the surfaces in the finite element model specified by the `SURFACE` command line or the exterior of blocks of elements via the `BLOCK` or `INCLUDE ALL BLOCKS` command line. (Any surface specified on the `REMOVE SURFACE` command line is then removed from this set.)

Table 5.1 lists the face variables used by the `BLAST PRESSURE` boundary condition. In the case that a `name` is specified the variables are prepended as `_name`. These can be requested for output in the standard manner, and can be useful for verifying that this boundary condition is correctly applied.

The type of burst load is specified with the `BURST TYPE` command, which can be `SURFACE` or `AIR`. The `SURFACE` option is used to define a hemispherical burst, while the `AIR` option is used for a spherical burst.

The equivalent amount of TNT (in pounds) is defined with the `TNT MASS IN LBS` command. The time at which the explosive is detonated is defined using the `BLAST TIME` command. This can be negative, and can be used to start the analysis at the time when the blast reaches the structure, saving computational time. The location of the blast is defined with the `BLAST LOCATION` command. Both `BLAST TIME` and `BLAST LOCATION` should be specified in the unit system of the model.

The current ambient pressure and temperature are defined using the `ATMOSPHERIC PRESSURE IN PSI` and `AMBIENT TEMPERATURE IN FAHRENHEIT` commands, respectively. As implied by the command names, these must be supplied in units of pounds per square inch and degrees Fahrenheit.

Because of the empirical nature of this method for computing an explosive load, appropriate conversion factors for the unit system used in the model must be supplied. The commands `FEET PER MODEL UNITS`, `MILLISECONDS PER MODEL UNITS`, and `PSI PER MODEL UNITS` are used to specify the magnitude of one foot, one millisecond, and one pound per square inch in the unit system of the model.

All of the commands listed above are required. Scaling factors can optionally be applied to modify the peak pressure, the impulse, and the duration of the loading. The `PRESSURE SCALE FACTOR` command scales the the peak value of both the reflected and incident portions of the applied pressure. The `IMPULSE SCALE FACTOR` command scales the impulse of the reflected and incident portions of the applied pressure. The `POSITIVE DURATION SCALE FACTOR` command scales the duration of the reflected and incident portions of the applied pressure. Each of these scaling factors only affects the quantity that it modifies, for example, scaling the pressure does not affect the impulse or duration.

The `ACTIVE PERIODS` and `INACTIVE PERIODS` commands can optionally be used to activate or deactivate this boundary condition for certain time periods.

Optionally a surface blocking calculation can be performed to determine if some surfaces shadow others from the blast loading. The blocking computation is turned on via the `BLOCKING SURFACE CALCULATION = ON` command line. The blocking surface calculation will determine what percentage of each face is able to “see” the blast source point. The incident and reflected pressure of the blast is then multiplied by the uncovered area of the face. See Figure 5.1 for an example. In the figure the purple faces shadow the green faces causing some of the green faces to have the applied blast load reduced or zeroed out entirely. When using the blocking surfaces command every face included in the blast pressure boundary condition acts as both a face for application of pressure and a face potentially blocking other faces.

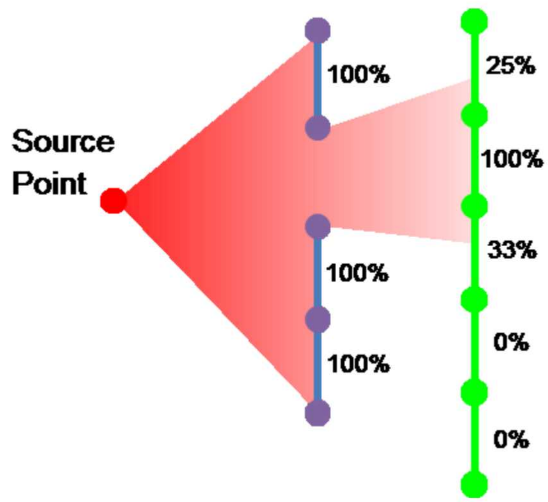

Figure 5.1: Example Blocking Surface Calculation.

Table 5.1: Face Variables for Blast Pressure Boundary Condition

Variable Name	Type	Comments
`pressure`	`Real`	Current total pressure. This is the only field for this boundary condition that varies in time.
`normal`	`Vector_3D`	Face normal vector
`incident_pressure`	`Real`	Peak incident pressure
`reflected_pressure`	`Real`	Peak reflected pressure
`alpha`	`Real`	Decay coefficient α
`beta`	`Real`	Decay coefficient β
`cosa`	`Real`	Cosine of θ
`arrival_time`	`Real`	Time for arrival of blast at face
`positive_duration`	`Real`	Duration of blast at face

References

[1] C.N. Kingery and G. Bulmash. Airblast parameters from TNT spherical air burst and hemispherical surface burst. Technical Report ARBBRL-TR-02555, Ballistic Research Laboratory, Aberdeen Proving Ground, MD.

[2] G. Randers-Pehrson and K.A. Bannister. Airblast loading model for DYNA2D and DYNA3D. Technical Report ARL-TR-1310, Army Research Laboratory, March 1997.

[3] Protective Design Center, United States Army Corps of Engineers. Conwep 2.1.0.8. link.

Chapter 6

Output Variables for Material Models

Most material models have state variables that can be output upon request. State variables can be accessed by name or index, although most of the time they are accessed by name. They are only accessed by index under special circumstances. Refer to the Sierra/SolidMechanics 4.48 User's Guide for more information on how state variables are requested for output. Tables of state variables for the material models that are only available in Presto_ITAR are provided below. These tables contain the indices and names used to access the state variables.

Table 6.1: State Variables for Bodner-Partom Model (Section 2.1.1)

Index	**Name**	**Variable Description**
0	`FAILURE_FLAG`	
1	`EQPS`	
2	`PLASTIC_WORK`	
3	`INTERNAL_ENERGY`	
4	`EQPS_RATE`	
5	`BULK_VISCOSITY`	
6	`SQ_SOUND_SPEED`	
7	`INITIAL_VOLUME`	
8	`VOLUME_STRAIN`	
9	`PRESSURE`	
10	`ELEMENT_LENGTH`	
11	`EQUIVALENT_STRESS`	
12	`TEMPERATURE`	

Table 6.2: State Variables for CTH_EP Model (Section 2.2.3)

Index	Name	Variable Description
0	`EQPS`	
1	`EQDOT`	

Table 6.3: State Variables for CTH_JFRAC Model (Section 2.2.7)

Index	Name	Variable Description
0	`DAMAGE`	
1	`FAILURE_ FRACTION`	
2	`FAILURE_ THRESHOLD`	

Table 6.4: State Variables for Holmquist-Johnson-Cook Concrete Model (Section 2.1.2)

Index	Name	Variable Description
0	`FAILURE_FLAG`	
1	`EQPS`	
2	`PLASTIC_WORK`	
3	`INTERNAL_ ENERGY`	
4	`EQPS_RATE`	
5	`BULK_VISCOSITY`	
6	`SQ_SOUND_SPEED`	
7	`INITIAL_VOLUME`	
8	`VOLUME_STRAIN`	
9	`VOLUME_STRAIN_ PER_CURRENT_ VOLUME`	
10	`PRESSURE`	
11	`ELEMENT_LENGTH`	
12	`EQUIVALENT_ STRESS`	
13	`MAX_ VOLUMETRIC_ STRAIN`	
14	`DAMAGE`	

Table 6.5: State Variables for Hull Concrete Model (Section 2.1.3)

Index	Name	Variable Description
0	FAILURE_FLAG	
1	EQPS	
2	PLASTIC_WORK	
3	INTERNAL_ ENERGY	
4	EQPS_RATE	
5	ARTIFICIAL_ VISCOSITY	
6	SQ_SOUND_SPEED	
7	VOLUME_STRAIN	
8	PRESSURE	
9	ELEMENT_LENGTH	
10	EFFECTIVE_ STRESS	
11	MAX_VOL_ STRAIN_CUR_VOL	

Table 6.6: State Variables for Johnson-Holmquist Ceramic Models (Section 2.1.5)

Index	Name	Variable Description
0	FAILURE_FLAG	
1	EQPS	
2	PLASTIC_WORK	
3	INTERNAL_ ENERGY	
4	EQPS_RATE	
5	BULK_VISCOSITY	
6	SQ_SOUND_SPEED	
7	INITIAL_VOLUME	
8	VOLUME_STRAIN	
9	VOLUME_STRAIN_ PER_CURRENT_ VOLUME	
10	PRESSURE	
11	ELEMENT_LENGTH	
12	EQUIVALENT_ STRESS	
13	DAMAGE	
14	BULKING_ PRESSURE	
15	Z_FORCE	

Table 6.7: State Variables for Johnson-Holmquist-Beissel Ceramic Models (Section 2.1.6)

Index	Name	Variable Description
0	FAILURE_FLAG	
1	EQPS	
2	PLASTIC_WORK	
3	INTERNAL_ ENERGY	
4	EQPS_RATE	
5	BULK_VISCOSITY	
6	SQ_SOUND_SPEED	
7	INITIAL_VOLUME	
8	VOLUME_STRAIN	
9	VOLUME_STRAIN_ PER_CURRENT_ VOLUME	
10	PRESSURE	
11	ELEMENT_LENGTH	
12	EQUIVALENT_ STRESS	
13	DAMAGE	
14	BULKING_ PRESSURE	
15	MAX_VOLUME_ STRAIN	
16	Z_FORCE	

Table 6.8: State Variables for Johnson-Cook Model (Section 2.1.4)

Index	Name	Variable Description
0	FAILURE_FLAG	
1	EQPS	
2	PLASTIC_WORK	
3	INTERNAL_ ENERGY	
4	EQPS_RATE	
5	BULK_VISCOSITY	
6	SQ_SOUND_SPEED	
7	INITIAL_VOLUME	
8	VOLUME_STRAIN	
9	VOLUME_STRAIN_ CURRENT	
10	PRESSURE	
11	SBAR	
12	EQUIVALENT_ STRESS	
13	TEMPERATURE	
14	DAMAGE	
15	INITIAL_FAIL_ STRAIN	

Chapter 7

Zapotec

Warning: Support for Zapotec coupling in Sierra/SM ITAR is currently at an developmental level. Work is progressing on making this capability a full production capability. As such, not all features may be fully implemented or tested and the analyst should use this capability with caution.

Note that there is now a dedicated Zapotec User's manual and an examples problem manual as part of the Sierra distribution for Zapotec. These are the first places to look for information on running Zapotec. The information in this chapter is a quick overview, and is secondary to the data in the full manuals. See Reference [1] and Reference [2] for more details.

This chapter documents a code coupling capability known as "Zapotec" that is currently only available in the ITAR versions of the codes. Zapotec is a two-way coupled CTH and Sierra/SM capability that is available via the executable named "zapotec". This capability couples a Lagrangian explicit dynamics region run in Sierra/SM with an Eulerian shock physics region run in CTH. Zapotec couples these codes by inserting the solid material from Sierra/SM into CTH at each time step and returning nodal forces to Sierra/SM. Zapotec is similar to Fortissimo, however material that is inserted into CTH is treated as a CTH material (termed a "placeholder") instead of being treated as a fully rigid material. By treating the inserted material as a deformable CTH material, Zapotec enables modeling of solid-on-solid interactions (i.e. impacts of materials with similar impedance) as well as blast-type simulations. A limited set of Sierra/SM and CTH capabilities can be used with Zapotec.

More complete information on using Zapotec can be found in the Zapotec documentation in the Sierra distribution, Reference [1]. In addition, an examples manual has been created which is very useful for users looking to start using Zapotec. See Reference [2] for more details.

7.1 Coupling Algorithm Description

Zapotec in Sierra is an update of the original Zapotec code, which couples CTH with the Lagrangian code Pronto3D. This Zapotec update preserves the fundamentals of the original Zapotec code, simply replacing the calls to Pronto3D to calls to Sierra/SM. Details on the Zapotec coupling algorithm can be found in Reference [1].

In brief, for each time step, Zapotec applies several processes to make the coupled codes consistent with each other, and then instructs the codes to independently compute a timestep of the same size. Consistency is then re-enforced on the codes, and a new timestep is computed.

To make the CTH description consistent with Sierra/SM, the Lagrangian material from Sierra/SM is inserted into the CTH region as a CTH "placeholder" material. Some key values are also inserted into the CTH cells, such as the density and stress state of the material. The insertion algorithm converts the mesh geometry into a set of tetrahedrons, which are then inserted into CTH cells using a volume-overlap approach. Shell elements are made into volumes based on their thickness, and the volume is inserted in the same way as solid elements. Zapotec does permit lofting of shells, though in practice lofting is not necessary if the CTH cell dimensions are no more than 4-5 times the shell thickness. Not all elements have to be inserted into the CTH region, though any that are left out will not be included in the coupling.

To make the Sierra/SM description consistent with CTH, pressures from the previous CTH solution are sampled and applied as forces to the the nodes of the Lagrangian mesh. Several sampling methods are available; see the Zapotec documentation (`force option` in Reference [1]) for more details.

The coupling algorithm also adds a "donation" capability, whereby any elements that are killed in Sierra/SM using element death can be permanently inserted into the CTH domain. This permits mass/momentum conservation during an analysis. Only materials explicitly identified will be donated upon element death. More details on this capability are available in the Zapotec documentation (Reference [1]).

7.2 Running Coupled Analysis

Zapotec runs must be handled with the "zapotec" executable using a command line such as:

```
sierra zapotec -i sierra_input.i
```

In all, five files are required to run Zapotec:

- Sierra/SM input file to describe the Lagrangian portion of the analysis
- corresponding mesh file for the Lagrangian portion of the analysis
- a CTH input file to describe the Eulerian portion
- a Zapotec input file to specify coupling details
- a summary file which provides the names of the main input files. The `sierra_input.i` file in the command above is the name of the summary file

The inputs for each of these files are described below. This document provides only cursory information on the CTH and Zapotec input files; it is recommended to reference separate documentation for details on the inputs for CTH (Reference [3]) and Zapotec (Reference [1]).

Prior to running a Zapotec analysis, it is required that the environment variable CTHPATH is set to an appropriate location in order for CTH to have access to spyplot routines and material libraries. Current Sierra distribution methods typically provide these files in a "cth" directory at the same level as the executable files.

7.3 Summary file Command Syntax

The summary file provides names for the Sierra/SM and CTH input files, and the run id for the CTH run. The file has the following syntax:

```
SIERRA INPUT DECK  = <string>sierra_input_deck
CTH INPUT DECK     = <string>cth_input_deck
CTH RUN ID         = <character>cth_run_id
ZAPOTEC INPUT DECK = <string>zapotec_input_deck [zapotec.inp]
```

The `SIERRA INPUT DECK` command specifies the name of the Sierra/SM input deck, which describes the Lagrangian portion of the analysis.

The `CTH INPUT DECK` command specifies the name of the CTH input deck, which describes the Eulerian portion of the analysis.

The `CTH RUN ID` command specifies the run id used by CTH to name its output files. The character can be a number (0-9) or a letter (a-z). The character is then used in the names of the generated output files (e.g. rsct# or osct#).

The `ZAPOTEC INPUT DECK` command specifies the name of the Zapotec input deck, which describes the details of the coupling. If this line is not specified, the code looks for a coupling file titled "zapotec.inp".

Note that the Sierra/SM input must be in centimeter-grams-seconds units, as are required in CTH.

7.4 Sierra/SM Input Deck

The Sierra/SM input deck describes the Lagrangian portion of the problem. The Sierra/SM input must be in centimeter-grams-seconds units, as are required in CTH.

Zapotec does require that a contact definition be in place that includes every block that is included in the coupling. An error is generated if the contact definition is not included.

Not all capabilities in Sierra/SM are supported in Zapotec. Examples of unsupported capabilities include:

- Coupling for any elements other than hexahedral solids or quadrilateral shells. Other elements can be present in the Sierra/SM analysis, but they cannot be coupled into the CTH solution.

- Mesh rebalancing

The Zapotec driver controls the frequency of output of results and restart files. Descriptions of results output and restart data are required in the Sierra/SM input deck, but no specification of output frequency is required. If one is given, then outputs are done at both the Zapotec and Sierra/SM specified times.

7.5 CTH Input Deck

The CTH input deck describes the Eulerian portion of the domain. Users are directed towards the CTH users manual for information on properly setting up a CTH input file (Reference [3]). Further restrictions of the CTH input deck for the Zapotec coupling algorithm are described in Reference [1].

The CTH input file will generally have three sets of material definitions. The first set are materials that are purely in the CTH domain, such as air and explosives. The second set are descriptions of materials that will be donated from the Lagrangian side, i.e. any elements that may be killed in Sierra/SM that should then be inserted permanently in the CTH domain. The third set of materials are the placeholder materials for materials present in Sierra/SM. Note that material numbers must follow this order; all the CTH-only materials must be numbered lower than the donated materials, which must in turn be numbered lower than the placeholder materials. For placeholder materials, it is best to only provide an EOS and no strength model. For donated materials, strength models are permitted, but it is important to have both the EOS and the strength model match as consistently as possible with the Lagrangian model. For both placeholder and donated models, it is usually most robust to use a simple Mie-Gruneisen model for the EOS.

Note also that no diatom insertion is needed for donated or placeholder materials. The coupling algorithm will automatically provide the insertion of these materials.

Note that Zapotec does not have the ability to control the output frequency of spymaster output, so all spy output frequencies should match the frequency specification as given in the Zapotec input deck.

7.6 Zapotec Input Deck

The Zapotec input deck provides details on the coupling algorithm. The file includes:

- specification of element blocks to be inserted into CTH
- specification of element blocks whose elements are eligible for donation
- output frequency (though this is only partially implemented; for best results, specify the outputs directly in both the CTH and Sierra/SM input files)
- specification of the pressure sampling methodology (force option)
- shell lofting, if desired
- various additional computational parameters

Proper execution of Zapotec requires careful attention to the parameters used in the zapotec input file. The user is encouraged to carefully read the user instructions in Reference [1].

7.7 Usage Guidelines and Restrictions

CTH/Sierra/SM coupling is currently incompatible with some capabilities including:

- Explicit multi-region time step subcycling within the Lagrangian region.
- Coupling with elements other than hexahedrons or quadrilateral shells.
- Solution control, for example to perform coupled thermo-structural analysis.
- Only one CTH region may be used at a time

If either the CTH run or the Sierra/SM portion of the run encounters a fatal error the analysis will abort. If CTH reaches its stop time prior to the the Sierra/SM analysis termination time Sierra/SM will continue running and applying no further CTH force. If the Sierra/SM analysis reaches its termination time, the CTH analysis will continue until its end time.

References

[1] G.C. Bessette, J.K. Prentice, R.L. Bell, C.T. Vaughan, R.A. Cole, and Sierra Solid Mechanics Team. *Zapotec: A Coupled Eulerian-Lagrangian Computer Code, Methodology and User Manual, Version 3.0.* Sandia National Laboratories, 2016.

[2] SIERRA Solid Mechanics Team. *Zapotec 3.0 Example Problems Manual*. Sandia National Laboratories, 2016.

[3] D.A. Crawford, A.L. Brundage, E.N. Harstad, E.S. Hertel Jr., R.G. Schmitt, S.C. Schumacher, and J.S. Simmons. *CTH User's Manual and Input Instructions, Version 10.0.* Sandia National Laboratories, 2011.

Index

Distribution

1	0899	Technical Library, 9536 (1 electronic)

Sandia National Laboratories