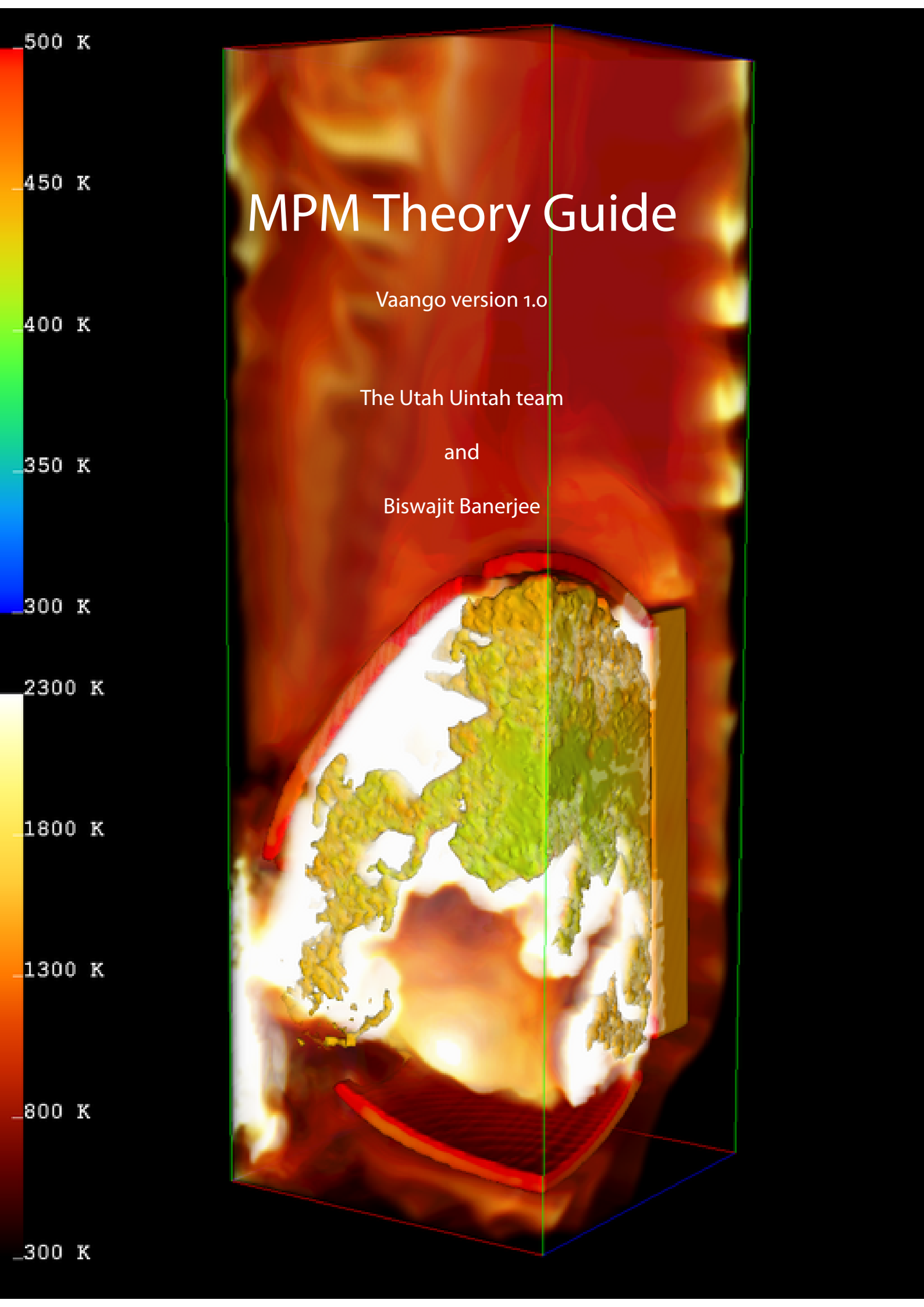


MPM Theory Guide

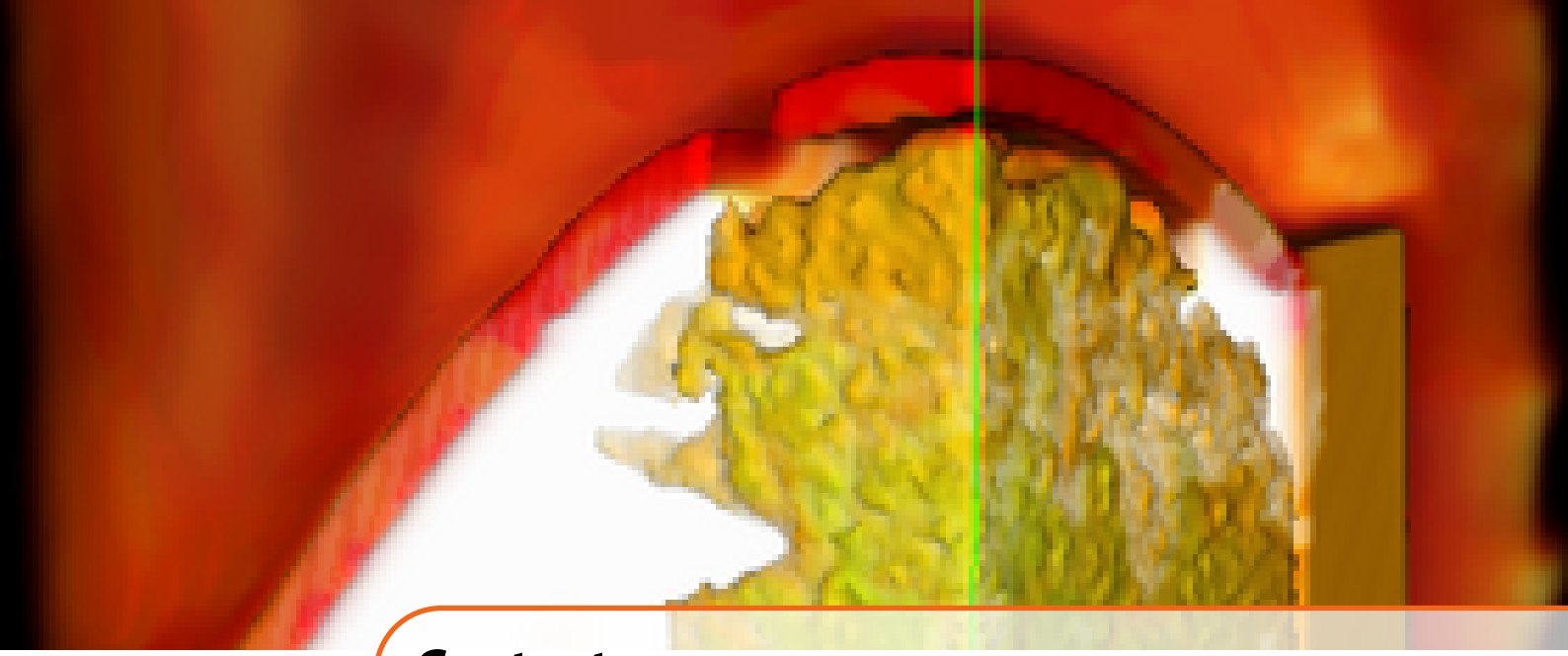
Vaango version 1.0

The Utah Uintah team

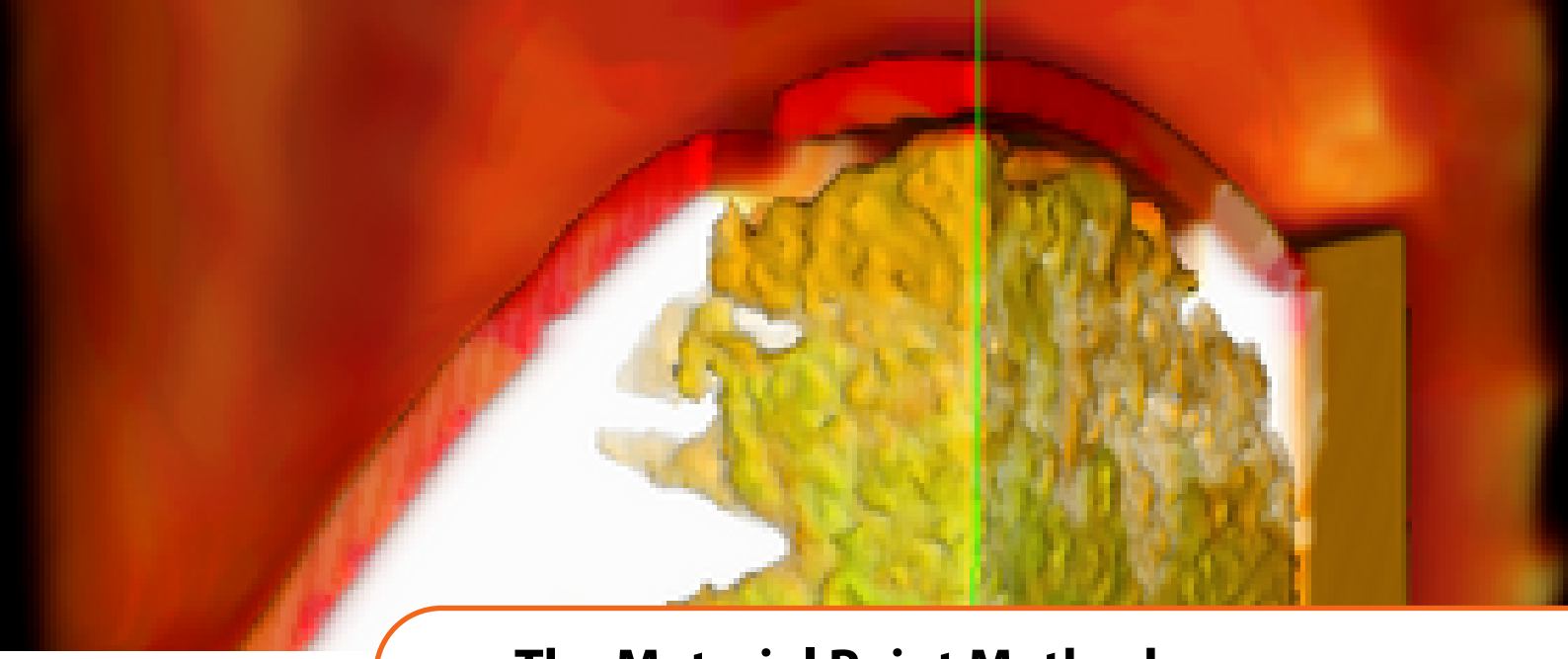
and

Biswajit Banerjee

The contents of this manual can and will change significantly over time. Please make sure that all the information is up to date.



Contents



1 — The Material Point Method

1.1 Introduction

The Material Point Method (MPM) as described by Sulsky, et al. [SCS94; SZS95] is a particle method for structural mechanics simulations. Solid objects are represented by a collection of particles, or "material points." Each of these particles carries with it information for that part of the solid object that it represents. This includes the mass, volume, position, velocity and stress of that material. MPM differs from other so called "mesh-free" particle methods in that, while each object is primarily represented by a collection of particles, a computational mesh is also an important part of the calculation. Particles do not interact with each other directly, rather the particle information is interpolated to the grid, where the equations of motion are integrated forward in time. This time advanced solution is then used to update the particle state. An example of two disks initially approaching each other represented by material points on an overlying mesh is show in Figure 1.1.

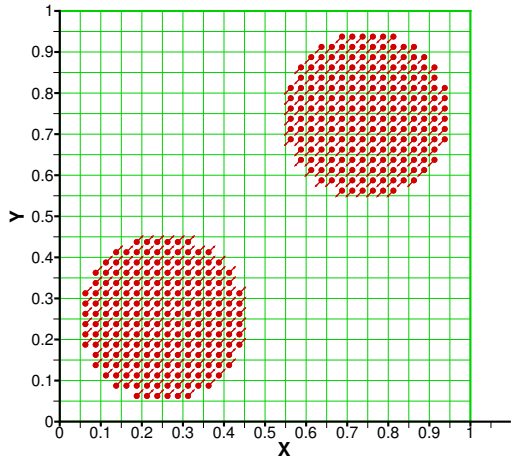


Figure 1.1: Initial particle representation of two colliding disks on an overlying mesh.

The method usually uses a regular structured grid as a computational mesh. While this grid, in principle, deforms as the material that it is representing deforms, at the end of each timestep, it is reset to it's original undeformed position, in effect providing a new computational grid for each timestep. The use of

a regular structured grid for each time step has a number of computational advantages. Computation of spatial gradients is simplified. Mesh entanglement, which can plague fully Lagrangian techniques, such as the Finite Element Method (FEM), is avoided. MPM has also been successful in solving problems involving contact between colliding objects, having an advantage over FEM in that the use of the regular grid eliminates the need for doing costly searches for contact surfaces[BBS00].

The choice of MPM over FEM as the C-SAFE structural mechanics method was only in small part for the above mentioned criteria. The primary motivation was the ability to use MPM together with a multimaterial CFD algorithm for solving tightly coupled fluid-structure interaction problems. This capability was first demonstrated in the CFDLIB codes from Los Alamos by Bryan Kashiwa and co-workers. There, as in Uintah, MPM serves as the Lagrangian description of the solid material in a multimaterial CFD code. Certain elements of the solution procedure are based in the Eulerian CFD algorithm, including intermaterial heat and momentum transfer as well as satisfaction of a multimaterial equation of state. The use of a Lagrangian method such as MPM to advance the solution of the solid material eliminates the diffusion typically associated with Eulerian methods.

1.2 Algorithm

While a more detailed description of MPM can be found in [SZS95], the algorithm is laid out here. The equations of motion are cast in the form:

$$\mathbf{M}_g \cdot \mathbf{a}_g = \mathbf{Fext}_g - \mathbf{Fint}_g \tag{1.1}$$

where $\mathbf{M}_g$ is the mass matrix, $\mathbf{a}_g$ is the acceleration vector, $\mathbf{Fext}_g$ is the external force vector (sum of the body forces and tractions), and $\mathbf{Fint}_g$ is the internal force vector resulting from the divergence of the material stresses. In general, $\mathbf{M}_g$ is a large, sparse matrix. In practice, and in what follows here, a "lumped" mass matrix is used, which only has entries on the diagonal, and is thus represented as a column matrix.

The solution procedure begins by interpolating the particle state to the grid, to form $\mathbf{M}_g$, $\mathbf{Fext}_g$, and to get a velocity on the grid $\mathbf{v}_g$. These quantities are calculated at each grid node by the following equations:

$$\mathbf{M}_i = \sum_p S_{ip} m_p \tag{1.2}$$

$$\mathbf{v}_i = \frac{\sum_p S_{ip} m_p \mathbf{v}_p}{\mathbf{M}_i} \tag{1.3}$$

$$\mathbf{Fext}_i = \sum_p S_{ip} \mathbf{Fext}_p. \tag{1.4}$$

m_p is the particle mass, $\mathbf{v}_p$ is the particle velocity, and $\mathbf{Fext}_p$ is the external force on the particle. The external force on the particle is generally an applied load of some type. In Equation 1.3, the numerator is the nodal momentum, which is then divided by the nodal mass to get a velocity. S_{ip} is a "shape function" for the ith node evaluated at $\mathbf{x}_p$. Traditionally, the shape functions are multiplicative combinations of one dimensional tent functions, as shown in Figure 1.2. The shape functions serve to distance weight the contribution of each particle to the grid nodes.

At this point, a velocity gradient, $\nabla \mathbf{v}_p$ is computed at each particle using the grid velocities $\mathbf{v}_g$:

$$\nabla \mathbf{v}_p = \sum_i \mathbf{G}_{ip} \mathbf{v}_I \tag{1.5}$$

where $\mathbf{G}_{ip}$ is the gradient of the ith node's shape function, evaluated at $\mathbf{x}_p$. A one dimensional example of $\mathbf{G}_{ip}$ is shown in Figure 1.3. Note that in going to multiple dimensions, the $\mathbf{G}_{ip}$ are found by taking gradients of the multidimensional S_{ip} NOT by forming multiplicative combinations of the one-dimensional $\mathbf{G}_{ip}$.

This velocity gradient is used as input to a constitutive model (stress-strain relationship) which is evaluated at each particle. The specifics of this calculation are dependent on the constitutive model. An example of a simple elastic material model is described in the appendix. The result of this calculation is the Cauchy stress at each particle, $\boldsymbol{\sigma}_p$. With this, the internal force due to the divergence of the stress is calculated:

$$\mathbf{Fint}_g = \sum_p \mathbf{G}_{ip} \boldsymbol{\sigma}_p v_p \tag{1.6}$$

where v_p is the particle volume. The internal force can be thought of as the force that holds a material together. For a given deformation, this force is larger for stiffer materials.

Everything is now available to solve Equation 1.1 for $\mathbf{a}_g$. With that, the backward Euler method is used for all time integrations. A convective grid velocity $\mathbf{v}_g^L$ is computed:

$$\mathbf{v}_g^L = \mathbf{v}_g + \mathbf{a}_g dt \tag{1.7}$$

While the following calculation is never carried out, in principal, the nodes of the grid also move with that convective velocity:

$$\mathbf{x}_g^L = \mathbf{x}_g + \mathbf{v}_g^L dt \tag{1.8}$$

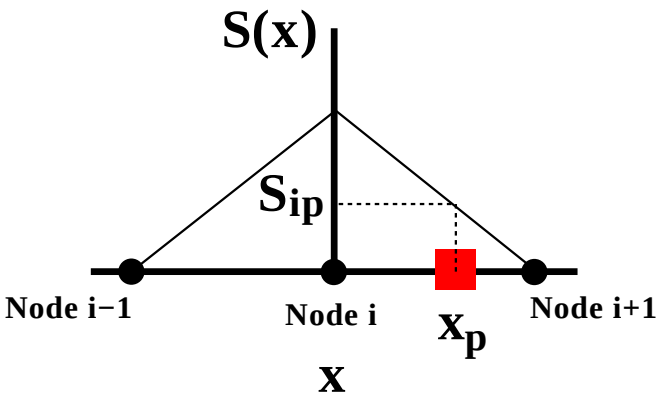
During this part of the computation, the particles move with the deforming grid. Their position and velocity is explicitly updated by:

$$\mathbf{v}_p(t + dt) = \mathbf{v}_p(t) + \sum_i S_{ip} \mathbf{a}_i dt \tag{1.9}$$

$$\mathbf{x}_p(t + dt) = \mathbf{x}_p(t) + \sum_i S_{ip} \mathbf{v}_i^L dt \tag{1.10}$$

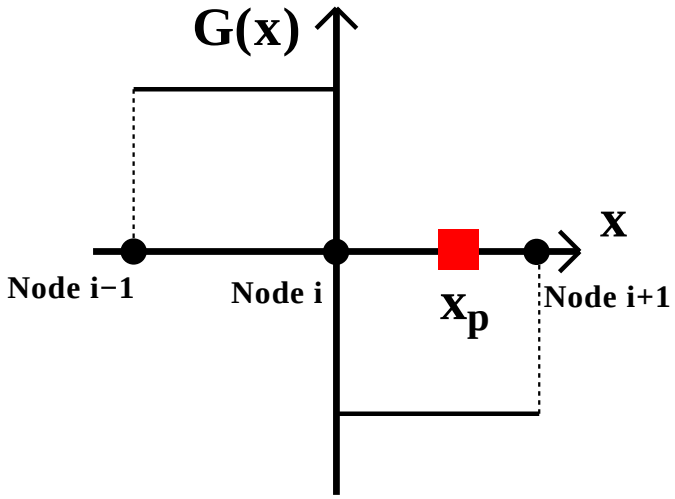
This completes one timestep. Note that not carrying out the calculation in 1.8 explicitly has the effect of resetting the deformed grid to it's undeformed position at the end of the timestep cycle.

As with all explicit time integration methods, a timestep size limit must be enforced such that $dt < dx/(|\mathbf{v}_p| + c)$ for all particles, where dx is the computational grid spacing and c is the speed at which stress waves propagate through the material. Failure to adhere to this condition will cause the solution to become unstable and blow up. The material wavespeed depends on the material model used, as well as on the particular parameters chosen for that model. Specifics of calculating the wavespeed are given in the appendix.



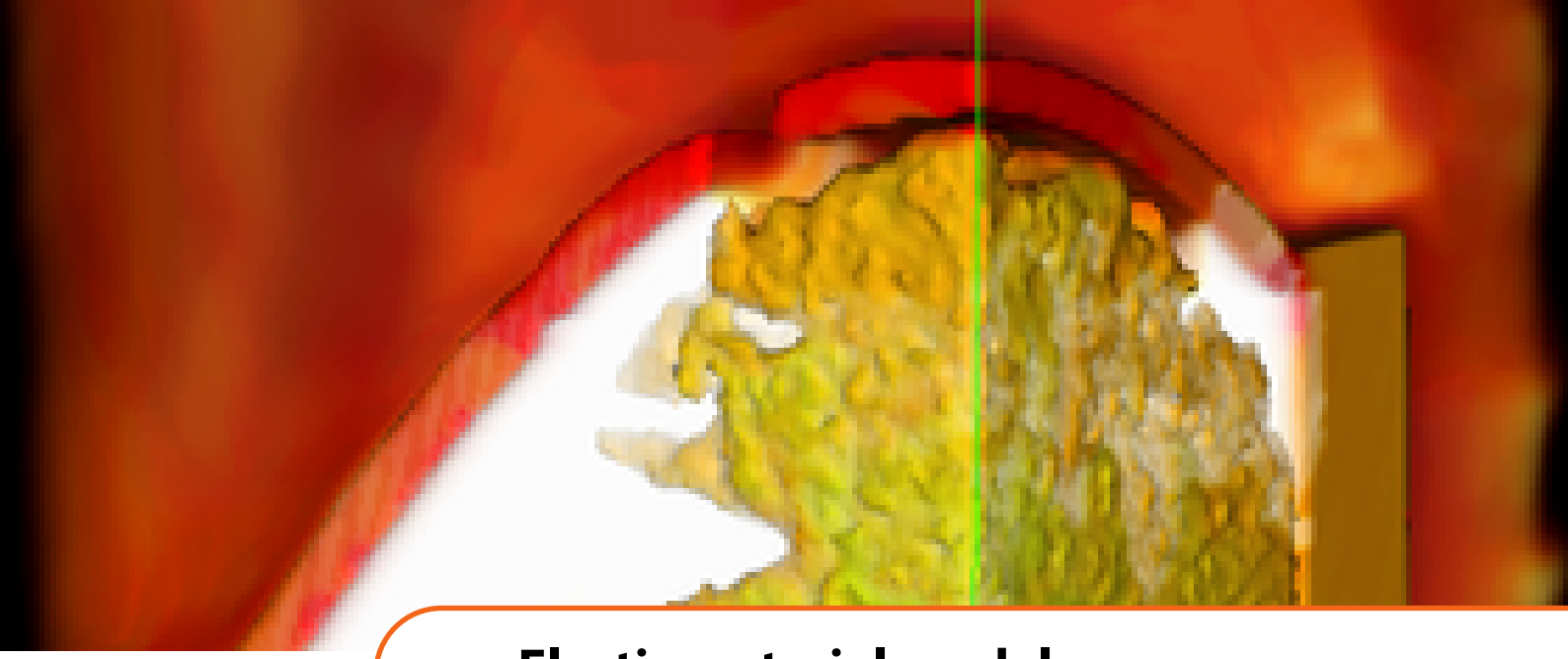
$$S(x) = (x - x_{i-1})/(x_i - x_{i-1}) \qquad x_{i-1} < x < x_i$$
$$S(x) = (x_{i+1} - x)/(x_{i+1} - x_i) \qquad x_i < x < x_{i+1}$$
$$S(x) = 0 \qquad x < x_{i-1} \qquad x > x_{i+1}$$

Figure 1.2: One dimensional linear shape function, $S(x)$.



$$G(x) = 1/(x_i - x_{i-1}) \qquad x_{i-1} < x < x_i$$
$$G(x) = -1/(x_{i+1} - x_i) \qquad x_i < x < x_{i+1}$$
$$G(x) = 0 \qquad x < x_{i-1} \qquad x > x_{i+1}$$

Figure 1.3: One dimensional linear shape function derivative, $G(x)$.



2 — Elastic material models

2.1 Hyperelastic Material Models

The subject of modeling the response of materials to deformation is a subject that has filled numerous textbooks. Therefore, rather than attempt to condense these volumes, here the reader will be simply be given a simple material response model. Other more complex material response models can be interchanged in the framework discussed above quite readily.

The author has come to prefer a class of models known as hyperelastic models. What this means is that the stress response of these materials is derived from a strain energy function. A strain energy function gives a relationship between the state of deformation that a material is in, and the amount of stored strain energy that this material has. This is akin to the familiar relationship for the stored energy in a spring, $W = \frac{1}{2}kdx^2$ where k is the spring constant, and dx is the distance that the spring has been compressed or extended.

One such strain energy function is given by:

$$W = \frac{\lambda}{4}(J^2 - 1) - (\frac{\lambda}{2} + \mu)lnJ + \frac{\mu}{2}(trace(\mathbf{F}^T\mathbf{F} - 3) \tag{2.1}$$

from which the following relationship for the stress can be derived:

$$\boldsymbol{\sigma} = \frac{\lambda}{2}(J - \frac{1}{J})\mathbf{I} + \mu(\mathbf{F}\mathbf{F}^T) - \mathbf{I}) \tag{2.2}$$

where λ and μ are material constants, while J and $\mathbf{F}$ describe the state of deformation. These will be defined shortly.

In the Algorithm section, the calculation of the velocity gradient, $\nabla\mathbf{v}_p$ is given in Equation 1.5. Starting from there, we can then compute an increment in the deformation gradient, $\mathbf{F}(dt)$ by:

$$\mathbf{F}(dt) = \nabla\mathbf{v}_p dt + \mathbf{I}. \tag{2.3}$$

This increment in the deformation gradient can then be used to compute a new total deformation gradient using:

$$\mathbf{F}(t+dt) \quad = \quad \mathbf{F}(dt)\mathbf{F}(t). \tag{2.4}$$

Note that the initial (t=0) deformation gradient is simply the identity, i.e. $\mathbf{F}(0) = \mathbf{I}$. Now with the deformation gradient, one can compute J by:

$$J \quad = \quad det(\mathbf{F}(t+dt)). \tag{2.5}$$

Note that J represents the volumetric part of the deformation. Specifically, it is the ratio of the current volume of an element of material to it's original volume. Similarly, we can define an increment in J as:

$$J_{inc} \quad = \quad det(\mathbf{F}(dt)) \tag{2.6}$$

which is the ratio of the current volume of an element of material to it's volume at the previous timestep. Thus we can write:

$$v_p(t+dt) \quad = \quad J_{inc}v_p(t). \tag{2.7}$$

Elastic material properties are frequently given in terms of bulk and shear moduli, or κ and μ. The shear is sometimes denoted by G. The shear modulus μ appears in Equation 2.2 above. λ can be computed from κ and μ by:

$$\lambda \quad = \quad \kappa - \frac{2}{3}\mu. \tag{2.8}$$

Lastly, based on material properties λ and μ, a material wavespeed can be computed:

$$c^2 \quad = \quad (\lambda + 3\mu)\frac{m_p}{v_p}. \tag{2.9}$$

This wavespeed can be used in computing the timestep size as described above.

2.2 Other Material Models In the UCF

Other material models implemented into the Uintah Computational Framework (UCF) have been chosen for three purposes:

- To verify the accuracy of the material point method (MPM) and to validate the coupling between the computational fluid dynamics code (ICE) and MPM.
- To model the elastic-plastic deformation of the steel container and the consequent damage in the regimes of both high and low strain rates and high and low temperatures.
- To model the polymer bonded explosive contained in the container under various strain rates and temperatures.

2.2.1 Material models for the validation of MPM

The models that have been implemented for the verification of MPM are:

- Isotropic hypoelastic model using the Jaumann rate of stress.
  1. MPM predictions have been compared with exact results for thick cylinders under internal pressure for small strains, three-point beam bending, etc.
  2. MPM predictions for the strain/stress contours for a set of disks in contact have been found to match experimental results.
- Isotropic hyperelastic material models for Mooney-Rivlin rubber and a modified compressible Neo-Hookean material. Isotropic strain hardening plasticity for the Neo-Hookean material.
  1. A billet compression problem has been simulated using MPM and the results have been found to closely match finite element simulations.
  2. MPM simulations for a thick cylinder under internal pressure with plastic deformation (perfect plasticity) compare well with the exact solution.

2.2.2 Material models for the container

The material model for the steel container is used to determine the state of stress in the container for an applied deformation rate and deformation gradient at each material point. The strain rates can vary from 10^{-3}/s to 10^{6}/s and temperatures in the container can vary from 250 K to 1000 K. Plasticity dominates the deformation of the container during the expansion of the explosive gases inside. At high strain rates the volumetric response of the container is best obtained using an equation of state. After the plastic strain in the container has reached a threshold value a damage/erosion model is required to rupture the container.

Two plasticity models with strain rate and temperature dependency are the Johnson-Cook and the Mechanical Threshold Stress (MTS) models. The volumetric response is calculated using a modified Mie-Gruneisen equation of state. A damage model that ties in well with the Johnson-Cook plasticity model is the Johnson-Cook damage model. The erosion algorithm either removes the contribution of the mass of the material point or forces the material point to undergo no tension or shear under further loading.

The stress update at each material point is performed using either of the two methods discussed below.

- Isotropic Hypoelastic-plastic material model using an additive decomposition of the rate of deformation tensor.
  1. The rate of deformation tensor a material point is calculated using the grid velocities.
  2. An incremental update of the left stretch and the rate of rotation tensors is calculated.
  3. The stress and the rate of deformation are rotated into the material coordinates.
  4. A trial elastic deviatoric stress state is calculated.
  5. The flow stress is calculated using the plasticity model and compared with the vonMises yield condition.
  6. If the stress state is elastic, an update of the stress is computed using the Mie-Gruneisen equation of state or the isotropic hypoelastic constitutive equation.
  7. If the stress state is plastic, all the strain rate is considered to the plastic and an elastic correction along with a radial return step move the stress state to the yield surface. The hydrostatic part of the stress is calculated using the equation of state or the hypoelastic constitutive equation.
  8. A scalar damage parameter is calculated and used to determine whether material points are to be eroded or not.
  9. Stresses and deformation rates are rotated back to the laboratory coordinates.
- Isotropic Hyperelastic-plastic material model using a multiplicative decomposition of the deformation gradient.
  1. The velocity gradient at a material point is calculated using the grid velocities.
  2. An incremental update of the deformation gradient and the left Cauchy-Green tensor is cal-

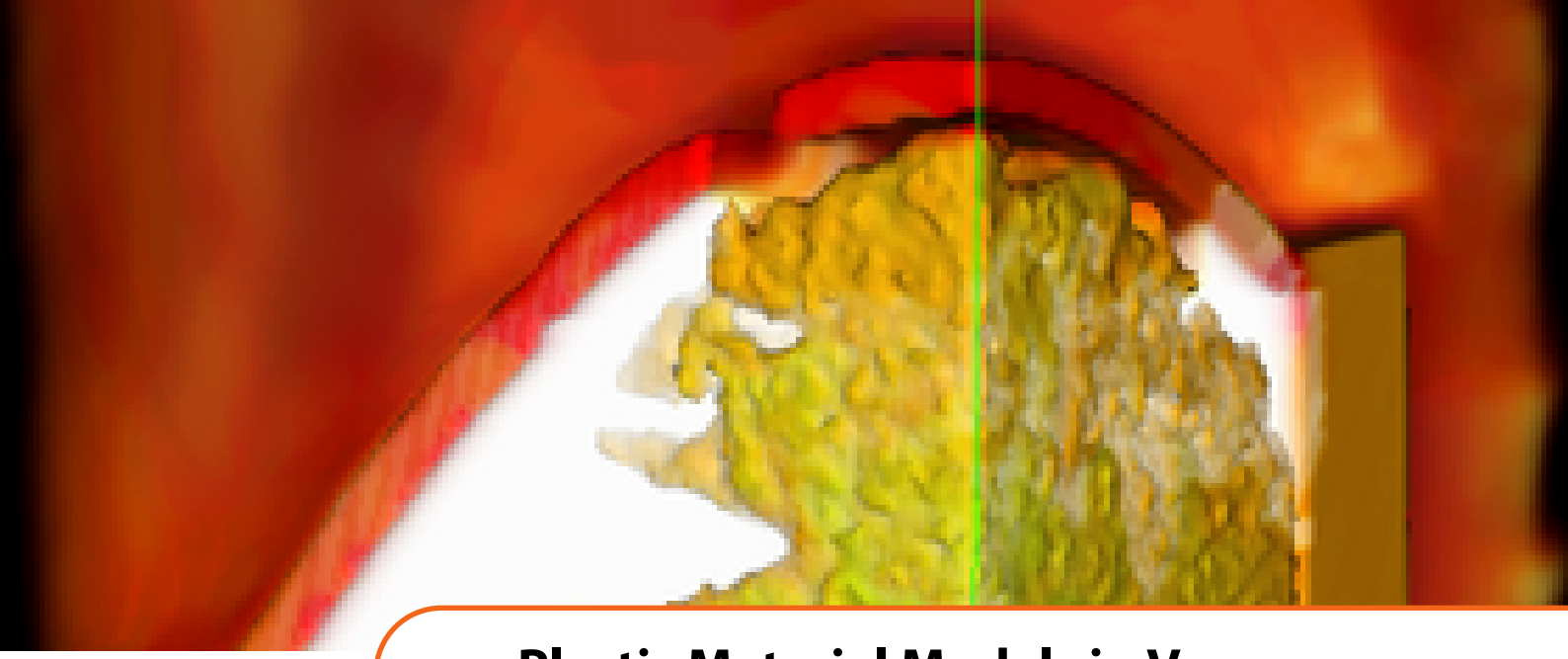
culated.

3. A trial elastic deviatoric stress state is calculated assuming a compressible Neo-Hookean elastic model.
4. The flow stress is calculated using the plasticity model and compared with the vonMises yield condition.
5. If the stress state is elastic, an update of the stress is computed using the Mie-Gruneisen equation of state or the compressible Neo-Hookean constitutive equation.
6. If the stress state is plastic, all the strain rate is considered to the plastic and an elastic correction along with a radial return step move the stress state to the yield surface. The hydrostatic part of the stress state is calculated using the Mie-Gruneisen equation of state or the Neo-Hookean model.
7. A scalar damage parameter is calculated and used to determine whether material points are to be eroded or not.

The implementations have been tested against Taylor impact test data for 4340 steel and HY 100 steel as well as one-dimensional problems which have been compared with experimental stress-strain data. At large tensile strains, material points tend to separate from the main body. This issue is currently being explored and solutions are being sought in the framework of MPM.

2.2.3 Material models for the explosive

The explosive is modeled using the ViscoSCRAM constitutive model. Since large deformations or strains are not expected in the explosive, a small strain formulation has been implemented into the UCF. The model consists of five generalized Maxwell elements arranged in parallel, crack growth, friction at the crack interfaces and heating due to friction and reactions at the crack surfaces. The implementation has been verified with experimental data and found to be accurate.



3 — Plastic Material Models in Vaango

3.1 Introduction

This document deals with some features of Uintah and some material models for solids that have been implemented in the Uintah Computational Framework (for use with the Material Point Method). The approach taken has been to separate the stress-strain relations from the numerical stress update algorithms as far as possible. A moderate rotation/small strain hypoelastic-plastic stress update algorithm is discussed and the manner in which plasticity flow rules, damage models and equations of state fit into the stress update algorithms are shown.

3.2 Stress Update Algorithms

The hypoelastic-plastic stress update is based on an additive decomposition of the rate of deformation tensor into elastic and plastic parts while the hyperelastic-plastic stress update is based on a multiplicative decomposition of the elastic and plastic deformation gradients. Incompressibility is assumed for plastic deformations. The volumetric response is therefore determined either by a bulk modulus and the trace of the rate of deformation tensor or using an equation of state. The deviatoric response is determined either by an elastic constitutive equation or using a plastic flow rule in combination with a yield condition.

The material models that can be varied in these stress update approaches are (not all are applicable to both hypo- and hyperelastic formulations nor is the list exhaustive):

1. The elasticity model, for example,
   - Isotropic linear elastic model.
   - Anisotropic linear elastic models.
   - Isotropic nonlinear elastic models.
   - Anisotropic nonlinear elastic models.
2. Isotropic hardening or Kinematic hardening using a back stress evolution rule, for example,
   - Ziegler evolution rule .
3. The flow rule and hardening/softening law, for example,
   - Perfect plasticity/power law hardening plasticity.
   - Johnson-Cook plasticity .
   - Mechanical Threshold Stress (MTS) plasticity .
   - Anand plasticity .
4. The yield condition, for example,

- von Mises yield condition.
- Drucker-Prager yield condition.
- Mohr-Coulomb yield condition.

5. A continuum or nonlocal damage model with damage evolution given by, for example,
   - Johnson-Cook damage model.
   - Gurson-Needleman-Tvergaard model.
   - Sandia damage model.
6. An equation of state to determine the pressure (or volumetric response), for example,
   - Mie-Gruneisen equation of state.

The currently implemented stress update algorithms in the Uintah Computational Framework do not allow for arbitrary elasticity models, kinematic hardening, arbitrary yield conditions and continuum or nonlocal damage (however the a damage parameter is updated and used in the erosion algorithm). The models that can be varied are the flow rule models, damage variable evolution models and the equation of state models.

Note that there are no checks in the Unitah Computational Framework to prevent users from mixing and matching inappropriate models.

This section describes the current implementation of the hypoelastic- plastic model. The stress update algorithm is a slightly modified version of the approach taken by Nemat-Nasser et al. (1991,1992) [Nem91; NC92], Wang (1994) [WA94], Maudlin (1996) [MS96], and Zocher et al. (2000) [Zoc+00].

3.2.1 Simplified theory for hypoelastic-plasticity

A simplified version of the theory behind the stress update algorithm (in the contex of von Mises plasticity) is given below.

Following [MS96], the rotated spatial rate of deformation tensor ($\boldsymbol{d}$) is decomposed into an elastic part ($\boldsymbol{d}^e$) and a plastic part ($\boldsymbol{d}^p$)

$$\boldsymbol{d} = \boldsymbol{d}^e + \boldsymbol{d}^p \tag{3.1}$$

If we assume plastic incompressibility ($\mathrm{tr}(\boldsymbol{d}^p) = 0$), we get

$$\boldsymbol{\eta} = \boldsymbol{\eta}^e + \boldsymbol{\eta}^p \tag{3.2}$$

where $\boldsymbol{\eta}$, $\boldsymbol{\eta}^e$, and $\boldsymbol{\eta}^p$ are the deviatoric parts of $\boldsymbol{d}$, $\boldsymbol{d}^e$, and $\boldsymbol{d}^p$, respectively. For isotropic materials, the hypoelastic constitutive equation for deviatoric stress is

$$\dot{\boldsymbol{s}} = 2\mu(\boldsymbol{\eta} - \boldsymbol{\eta}^p) \tag{3.3}$$

where $\boldsymbol{s}$ is the deviatoric part of the stress tensor and μ is the shear modulus. We assume that the flow stress obeys the Huber-von Mises yield condition

$$f := \sqrt{\frac{3}{2}}\,\|\boldsymbol{s}\| - \sigma_y \le 0 \quad \text{or,} \quad F := \frac{3}{2}\boldsymbol{s} : \boldsymbol{s} - \sigma_y^2 \le 0 \tag{3.4}$$

where σ_y is the flow stress. Assuming an associated flow rule, and noting that $\boldsymbol{d}^p = \boldsymbol{\eta}^p$, we have

$$\boldsymbol{\eta}^p = \boldsymbol{d}^p = \lambda\frac{\partial f}{\partial \boldsymbol{\sigma}} = \Lambda\frac{\partial F}{\partial \boldsymbol{\sigma}} = 3\Lambda\boldsymbol{s} \tag{3.5}$$

where $\boldsymbol{\sigma}$ is the stress. Let $\mathbf{u}$ be a tensor proportional to the plastic straining direction, and define γ as

$$\mathbf{u} = \sqrt{3}\frac{\boldsymbol{s}}{\|\boldsymbol{s}\|}; \quad \gamma := \sqrt{3}\Lambda\,\|\boldsymbol{s}\| \quad \Longrightarrow \gamma\mathbf{u} = 3\Lambda\boldsymbol{s} \tag{3.6}$$

Therefore, we have

$$\boldsymbol{\eta}^p = \gamma \mathbf{u}; \quad \dot{\boldsymbol{s}} = 2\mu(\boldsymbol{\eta} - \gamma \mathbf{u}) \tag{3.7}$$

From the consistency condition, if we assume that the deviatoric stress remains constant over a timestep, we get

$$\gamma = \frac{\boldsymbol{s} : \boldsymbol{\eta}}{\boldsymbol{s} : \mathbf{u}} \tag{3.8}$$

which provides an initial estimate of the plastic strain-rate. To obtain a semi-implicit update of the stress using equation (3.7), we define

$$\tau^2 := \frac{3}{2}\boldsymbol{s} : \boldsymbol{s} = \sigma_y^2 \tag{3.9}$$

Taking a time derivative of equation (3.9) gives us

$$\sqrt{2}\dot{\tau} = \sqrt{3}\frac{\boldsymbol{s} : \dot{\boldsymbol{s}}}{\|\boldsymbol{s}\|} \tag{3.10}$$

Plugging equation (3.10) into equation $(3.7)_2$ we get

$$\dot{\tau} = \sqrt{2}\mu(\mathbf{u} : \boldsymbol{\eta} - \gamma \mathbf{u} : \mathbf{u}) = \sqrt{2}\mu(d - 3\gamma) \tag{3.11}$$

where $d = \mathbf{u} : \boldsymbol{\eta}$. If the initial estimate of the plastic strain-rate is that all of the deviatoric strain-rate is plastic, then we get an approximation to γ, and the corresponding error (γ_{er}) given by

$$\gamma_{\text{approx}} = \frac{d}{3}; \quad \gamma_{\text{er}} = \gamma_{\text{approx}} - \gamma = \frac{d}{3} - \gamma \tag{3.12}$$

The incremental form of the above equation is

$$\Delta\gamma = \frac{d^* \Delta t}{3} - \Delta\gamma_{\text{er}} \tag{3.13}$$

Integrating equation (3.11) from time t_n to time $t_{n+1} = t_n + \Delta t$, and using equation (3.13) we get

$$\tau_{n+1} = \tau_n + \sqrt{2}\mu(d^* \Delta t - 3\Delta\gamma) = \tau_n + 3\sqrt{2}\mu\Delta\gamma_{\text{er}} \tag{3.14}$$

where d^* is the average value of d over the timestep. Solving for $\Delta\gamma_{\text{er}}$ gives

$$\Delta\gamma_{\text{er}} = \frac{\tau_{n+1} - \tau_n}{3\sqrt{2}\mu} = \frac{\sqrt{2}\sigma_y - \sqrt{3}\,\|\boldsymbol{s}_n\|}{6\mu} \tag{3.15}$$

The direction of the total strain-rate ($\mathbf{u}^\eta$) and the direction of the plastic strain-rate ($\mathbf{u}^s$) are given by

$$\mathbf{u}^\eta = \frac{\boldsymbol{\eta}}{\|\boldsymbol{\eta}\|}; \quad \mathbf{u}^s = \frac{\boldsymbol{s}}{\|\boldsymbol{s}\|} \tag{3.16}$$

Let θ be the fraction of the time increment that sees elastic straining. Then

$$\theta = \frac{d^* - 3\gamma_n}{d^*} \tag{3.17}$$

where $\gamma_n = d_n/3$ is the value of γ at the beginning of the timestep. We also assume that

$$d^* = \sqrt{3}\boldsymbol{\eta} : \left[(1-\theta)\mathbf{u}^\eta + \frac{\theta}{2}(\mathbf{u}^\eta + \mathbf{u}^s)\right] \tag{3.18}$$

Plugging equation (3.17) into equation (3.18) we get a quadratic equation that can be solved for d^* as follows

$$\frac{2}{\sqrt{3}}(d^*)^2 - (\boldsymbol{\eta} : \mathbf{u}^s + \|\boldsymbol{\eta}\|)d^* + 3\gamma_n(\boldsymbol{\eta} : \mathbf{u}^s - \|\boldsymbol{\eta}\|) = 0 \tag{3.19}$$

The real positive root of the above quadratic equation is taken as the estimate for d. The value of $\Delta\gamma$ can now be calculated using equations (3.13) and (3.15). A semi-implicit estimate of the deviatoric stress can be obtained at this stage by integrating equation $(3.7)_2$

$$\tilde{\boldsymbol{s}}_{n+1} = \boldsymbol{s}_n + 2\mu\left(\eta\Delta t - \sqrt{3}\Delta\gamma\frac{\tilde{\boldsymbol{s}}_{n+1}}{\|\boldsymbol{s}_{n+1}\|}\right) \tag{3.20}$$

$$= \boldsymbol{s}_n + 2\mu\left(\eta\Delta t - \frac{3}{\sqrt{2}}\Delta\gamma\frac{\tilde{\boldsymbol{s}}_{n+1}}{\sigma_y}\right) \tag{3.21}$$

Solving for $\tilde{\boldsymbol{s}}_{n+1}$, we get

$$\tilde{\boldsymbol{s}}_{n+1} = \frac{\boldsymbol{s}^{\text{trial}}_{n+1}}{1 + 3\sqrt{2}\mu\dfrac{\Delta\gamma}{\sigma_y}} \tag{3.22}$$

where $\boldsymbol{s}^{\text{trial}}_{n+1} = \boldsymbol{s}_n + 2\mu\Delta t\boldsymbol{\eta}$. A final radial return adjustment is used to move the stress to the yield surface

$$\boldsymbol{s}_{n+1} = \sqrt{\frac{2}{3}}\sigma_y\frac{\tilde{\boldsymbol{s}}_{n+1}}{\|\tilde{\boldsymbol{s}}_{n+1}\|} \tag{3.23}$$

A pathological situation arises if $\gamma_n = \mathbf{u}_n : \boldsymbol{\eta}_n$ is less than or equal to zero or $\Delta\gamma_{\text{er}} \geq \frac{d^*}{3}\Delta t$. This can occur is the rate of plastic deformation is small compared to the rate of elastic deformation or if the timestep size is too small (see [NC92]). In such situations, we use a locally implicit stress update that uses Newton iterations (as discussed in [SH98], page 124) to compute $\tilde{\boldsymbol{s}}$.

3.2.2 Models

Below are some of the strain-rate, strain, and temperature dependent models for metals that are implemented in Uintah.

3.2.3 Equation of State Models

The elastic-plastic stress update assumes that the volumetric part of the Cauchy stress can be calculated using an equation of state. There are three equations of state that are implemented in Uintah. These are

1. A default hypoelastic equation of state.
2. A neo-Hookean equation of state.
3. A Mie-Gruneisen type equation of state.

Default hypoelastic equation of state

In this case we assume that the stress rate is given by

$$\dot{\boldsymbol{\sigma}} = \lambda\,\text{tr}(\boldsymbol{d}^e)\,\boldsymbol{1} + 2\,\mu\,\boldsymbol{d}^e \tag{3.24}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress, $\boldsymbol{d}^e$ is the elastic part of the rate of deformation, and λ, μ are constants.

If $\boldsymbol{\eta}^e$ is the deviatoric part of $\boldsymbol{d}^e$ then we can write

$$\dot{\boldsymbol{\sigma}} = \left(\lambda + \frac{2}{3}\,\mu\right)\text{tr}(\boldsymbol{d}^e)\,\boldsymbol{1} + 2\,\mu\,\boldsymbol{\eta}^e = \kappa\,\text{tr}(\boldsymbol{d}^e)\,\boldsymbol{1} + 2\,\mu\,\boldsymbol{\eta}^e\,. \tag{3.25}$$

If we split $\boldsymbol{\sigma}$ into a volumetric and a deviatoric part, i.e., $\boldsymbol{\sigma} = p\,\mathbf{1} + \boldsymbol{s}$ and take the time derivative to get $\dot{\boldsymbol{\sigma}} = \dot{p}\,\mathbf{1} + \dot{\boldsymbol{s}}$ then

$$\dot{p} = \kappa\,\mathrm{tr}(\boldsymbol{d}^e)\ . \tag{3.26}$$

In addition we assume that $\boldsymbol{d} = \boldsymbol{d}^e + \boldsymbol{d}^p$. If we also assume that the plastic volume change is negligible, we can then write that

$$\dot{p} = \kappa\,\mathrm{tr}(\boldsymbol{d})\ . \tag{3.27}$$

This is the equation that is used to calculate the pressure p in the default hypoelastic equation of state, i.e.,

$$\boxed{p_{n+1} = p_n + \kappa\,\mathrm{tr}(\boldsymbol{d}_{n+1})\,\Delta t\ .} \tag{3.28}$$

To get the derivative of p with respect to J, where $J = \det(\boldsymbol{F})$, we note that

$$\dot{p} = \frac{\partial p}{\partial J}\,\dot{J} = \frac{\partial p}{\partial J}\,J\,\mathrm{tr}(\boldsymbol{d})\ . \tag{3.29}$$

Therefore,

$$\boxed{\frac{\partial p}{\partial J} = \frac{\kappa}{J}\,.} \tag{3.30}$$

This model is invoked in Uintah using

```
<equation_of_state type="default_hypo">
</equation_of_state>
```

The code is in `.../MPM/ConstitutiveModel/PlasticityModels/DefaultHypoElasticEOS.cc`.

Default hyperelastic equation of state

In this model the pressure is computed using the relation

$$p = \frac{1}{2}\,\kappa\,\left(J^e - \frac{1}{J^e}\right) \tag{3.31}$$

where κ is the bulk modulus and J^e is determinant of the elastic part of the deformation gradient.

We can also compute

$$\frac{dp}{dJ} = \frac{1}{2}\,\kappa\,\left(1 + \frac{1}{(J^e)^2}\right)\ . \tag{3.32}$$

This model is invoked in Uintah using

```
<equation_of_state type="default_hyper">
</equation_of_state>
```

The code is in `.../MPM/ConstitutiveModel/PlasticityModels/HyperElasticEOS.cc`. If an EOS is not specified then this model is the **default**.

Mie-Gruneisen equation of state

The pressure (p) is calculated using a Mie-Grüneisen equation of state of the form ([Wil99; Zoc+00])

$$\boxed{p_{n+1} = -\frac{\rho_0\, C_0^2\, (1 - J^e_{n+1})[1 - \Gamma_0(1 - J^e_{n+1})/2]}{[1 - S_\alpha(1 - J^e_{n+1})]^2} - \Gamma_0\, e_{n+1}\,; \quad J^e := \det \boldsymbol{F}^e} \tag{3.33}$$

where C_0 is the bulk speed of sound, ρ_0 is the initial mass density, Γ_0 is the Grüneisen's gamma at the reference state, $S_\alpha = dU_s/dU_p$ is a linear Hugoniot slope coefficient, U_s is the shock wave velocity, U_p is the particle velocity, and e is the internal energy density (per unit reference volume), $\boldsymbol{F}^e$ is the elastic part of the deformation gradient. For isochoric plasticity,

$$J^e = J = \det(\boldsymbol{F}) = \frac{\rho_0}{\rho}\,.$$

The internal energy is computed using

$$E = \frac{1}{V_0}\int C_v dT \approx \frac{C_v(T - T_0)}{V_0} \tag{3.34}$$

where $V_0 = 1/\rho_0$ is the reference specific volume at temperature $T = T_0$, and C_v is the specific heat at constant volume.

Also,

$$\boxed{\frac{\partial p}{\partial J^e} = \frac{\rho_0\, C_0^2\, [1 + (S_\alpha - \Gamma_0)\,(1 - J^e)]}{[1 - S_\alpha\,(1 - J^e)]^3} - \Gamma_0\,\frac{\partial e}{\partial J^e}.} \tag{3.35}$$

We neglect the $\frac{\partial e}{\partial J^e}$ term in our calculations.

This model is invoked in Uintah using

```
<equation_of_state type="mie_gruneisen">
  <C_0>5386</C_0>
  <Gamma_0>1.99</Gamma_0>
  <S_alpha>1.339</S_alpha>
</equation_of_state>
```

The code is in `.../MPM/ConstitutiveModel/PlasticityModels/MieGruneisenEOS.cc`.

Melting Temperature

Default model

The default model is to use a constant melting temperature. This model is invoked using

```
<melting_temp_model type="constant_Tm">
</melting_temp_model>
```

SCG melt model

We use a pressure dependent relation to determine the melting temperature (T_m). The Steinberg-Cochran-Guinan (SCG) melt model ([SCG80]) has been used for our simulations of copper. This model is based on a modified Lindemann law and has the form

$$T_m(\rho) = T_{m0}\exp\left[2a\left(1 - \frac{1}{\eta}\right)\right]\eta^{2(\Gamma_0 - a - 1/3)}; \quad \eta = \frac{\rho}{\rho_0} \tag{3.36}$$

where T_{m0} is the melt temperature at $\eta = 1$, a is the coefficient of the first order volume correction to Grüneisen's gamma (Γ_0).

This model is invoked with

```
<melting_temp_model type="scg_Tm">
  <T_m0> 2310.0 </T_m0>
  <Gamma_0> 3.0 </Gamma_0>
  <a> 1.67 </a>
</melting_temp_model>
```

BPS melt model

An alternative melting relation that is based on dislocation-mediated phase transitions - the Burakovsky-Preston-Silbar (BPS) model ([BPSoo]) can also be used. This model has been used to determine the melt temperature for 4340 steel. The BPS model has the form

$$T_m(p) = T_m(0)\left[\frac{1}{\eta} + \frac{1}{\eta^{4/3}}\frac{\mu_0'}{\mu_0}p\right]; \quad \eta = \left(1 + \frac{K_0'}{K_0}p\right)^{1/K_0'} \tag{3.37}$$

$$T_m(0) = \frac{\kappa\lambda\mu_0 v_{WS}}{8\pi\ln(z-1)\,k_b}\ln\left(\frac{\alpha^2}{4\,b^2\rho_c(T_m)}\right) \tag{3.38}$$

where p is the pressure, $\eta = \rho/\rho_0$ is the compression, μ_0 is the shear modulus at room temperature and zero pressure, $\mu_0' = \partial\mu/\partial p$ is the derivative of the shear modulus at zero pressure, K_0 is the bulk modulus at room temperature and zero pressure, $K_0' = \partial K/\partial p$ is the derivative of the bulk modulus at zero pressure, κ is a constant, $\lambda = b^3/v_{WS}$ where b is the magnitude of the Burgers' vector, v_{WS} is the Wigner-Seitz volume, z is the coordination number, α is a constant, $\rho_c(T_m)$ is the critical density of dislocations, and k_b is the Boltzmann constant.

This model is invoked with

```
<melting_temp_model type="bps_Tm">
  <B0> 137e9 </B0>
  <dB_dp0> 5.48 <dB_dp0>
  <G0> 47.7e9 <G0>
  <dG_dp0> 1.4 <dG_dp0>
  <kappa> 1.25 <kappa>
  <z> 12 <z>
  <b2rhoTm> 0.64 <b2rhoTm>
  <alpha> 2.9 <alpha>
  <lambda> 1.41 <lambda>
  <a> 3.6147e-9<a>
  <v_ws_a3_factor> 1/4 <v_ws_a3_factor>
  <Boltzmann_Constant> <Boltzmann_Constant>
</melting_temp_model>
```

Shear Modulus

Three models for the shear modulus (μ) have been tested in our simulations. The first has been associated with the Mechanical Threshold Stress (MTS) model and we call it the MTS shear model. The second is the model used by Steinberg-Cochran-Guinan and we call it the SCG shear model while the third is a model developed by Nadal and Le Poac that we call the NP shear model.

Default model

The default model gives a constant shear modulus. The model is invoked using

```
<shear_modulus_model type="constant_shear">
</shear_modulus_model>
```

MTS Shear Modulus Model

The simplest model is of the form suggested by [Var70] ([CG96])

$$\mu(T) = \mu_0 - \frac{D}{exp(T_0/T) - 1} \tag{3.39}$$

where μ_0 is the shear modulus at 0K, and D, T_0 are material constants.

The model is invoked using

```
<shear_modulus_model type="mts_shear">
  <mu_0>28.0e9</mu_0>
  <D>4.50e9</D>
  <T_0>294</T_0>
</shear_modulus_model>
```

SCG Shear Modulus Model

The Steinberg-Cochran-Guinan (SCG) shear modulus model ([SCG80; Zoc+00]) is pressure dependent and has the form

$$\mu(p, T) = \mu_0 + \frac{\partial \mu}{\partial p}\frac{p}{\eta^{1/3}} + \frac{\partial \mu}{\partial T}(T - 300); \quad \eta = \rho/\rho_0 \tag{3.40}$$

where, μ_0 is the shear modulus at the reference state(T = 300 K, p = 0, η = 1), p is the pressure, and T is the temperature. When the temperature is above T_m, the shear modulus is instantaneously set to zero in this model.

The model is invoked using

```
<shear_modulus_model type="scg_shear">
  <mu_0> 81.8e9 </mu_0>
  <A> 20.6e-12 </A>
  <B> 0.16e-3 </B>
</shear_modulus_model>
```

NP Shear Modulus Model

A modified version of the SCG model has been developed by [NL03] that attempts to capture the sudden drop in the shear modulus close to the melting temperature in a smooth manner. The Nadal-LePoac (NP) shear modulus model has the form

$$\mu(p, T) = \frac{1}{\mathcal{J}(\hat{T})}\left[\left(\mu_0 + \frac{\partial \mu}{\partial p}\frac{p}{\eta^{1/3}}\right)(1 - \hat{T}) + \frac{\rho}{Cm}\, k_b\, T\right]; \quad C := \frac{(6\pi^2)^{2/3}}{3} f^2 \tag{3.41}$$

where

$$\mathcal{J}(\hat{T}) := 1 + \exp\left[-\frac{1 + 1/\zeta}{1 + \zeta/(1 - \hat{T})}\right] \quad \text{for} \quad \hat{T} := \frac{T}{T_m} \in [0, 1 + \zeta], \tag{3.42}$$

μ_0 is the shear modulus at 0 K and ambient pressure, ζ is a material parameter, k_b is the Boltzmann constant, m is the atomic mass, and f is the Lindemann constant.

The model is invoked using

```
<shear_modulus_model type="np_shear">
  <mu_0>26.5e9</mu_0>
  <zeta>0.04</zeta>
  <slope_mu_p_over_mu0>65.0e-12</slope_mu_p_over_mu0>
  <C> 0.047 </C>
  <m> 26.98 </m>
</shear_modulus_model>
```

PTW Shear model

The PTW shear model is a simplified version of the SCG shear model. The inputs can be found in `.../MPM/ConstitutiveModel/PlasticityModel/PTWShear.h`.

Flow Stress

We have explored five temperature and strain rate dependent models that can be used to compute the flow stress:

1. the Johnson-Cook (JC) model
2. the Steinberg-Cochran-Guinan-Lund (SCG) model.
3. the Zerilli-Armstrong (ZA) model.
4. the Mechanical Threshold Stress (MTS) model.
5. the Preston-Tonks-Wallace (PTW) model.

JC Flow Stress Model

The Johnson-Cook (JC) model ([JC83]) is purely empirical and gives the following relation for the flow stress (σ_y)

$$\sigma_y(\varepsilon_p, \dot{\varepsilon}_p, T) = \left[A + B(\varepsilon_p)^n\right]\left[1 + C\ln(\dot{\varepsilon}_p^*)\right]\left[1 - (T^*)^m\right] \tag{3.43}$$

where ε_p is the equivalent plastic strain, $\dot{\varepsilon}_p$ is the plastic strain rate, A, B, C, n, m are material constants,

$$\dot{\varepsilon}_p^* = \frac{\dot{\varepsilon}_p}{\dot{\varepsilon}_{p0}}; \quad T^* = \frac{(T - T_0)}{(T_m - T_0)}, \tag{3.44}$$

$\dot{\varepsilon}_{p0}$ is a user defined plastic strain rate, T_0 is a reference temperature, and T_m is the melt temperature. For conditions where $T^* < 0$, we assume that $m = 1$.

The inputs for this model are

```
<plasticity_model type="johnson_cook">
  <A>792.0e6</A>
  <B>510.0e6</B>
  <C>0.014</C>
  <n>0.26</n>
  <m>1.03</m>
  <T_r>298.0</T_r>
  <T_m>1793.0</T_m>
  <epdot_0>1.0</epdot_0>
</plasticity_model>
```

SCG Flow Stress Model

The Steinberg-Cochran-Guinan-Lund (SCG) model is a semi-empirical model that was developed by [SCG80] for high strain rate situations and extended to low strain rates and bcc materials by [SL89]. The flow stress in this model is given by

$$\sigma_y(\varepsilon_p, \dot{\varepsilon}_p, T) = \left[\sigma_a f(\varepsilon_p) + \sigma_t(\dot{\varepsilon}_p, T)\right]\frac{\mu(p, T)}{\mu_0} \tag{3.45}$$

where σ_a is the athermal component of the flow stress, $f(\varepsilon_p)$ is a function that represents strain hardening, σ_t is the thermally activated component of the flow stress, $\mu(p, T)$ is the shear modulus, and μ_0 is the shear modulus at standard temperature and pressure. The strain hardening function has the form

$$f(\varepsilon_p) = \left[1 + \beta(\varepsilon_p + \varepsilon_{pi})\right]^n; \quad \sigma_a f(\varepsilon_p) \le \sigma_{\max} \tag{3.46}$$

where β, n are work hardening parameters, and ε_{pi} is the initial equivalent plastic strain. The thermal component σ_t is computed using a bisection algorithm from the following equation (based on the work of [HM77])

$$\dot{\varepsilon}_p = \left[\frac{1}{C_1}\exp\left[\frac{2U_k}{k_b\,T}\left(1-\frac{\sigma_t}{\sigma_p}\right)^2\right]+\frac{C_2}{\sigma_t}\right]^{-1}; \quad \sigma_t \le \sigma_p \tag{3.47}$$

where $2U_k$ is the energy to form a kink-pair in a dislocation segment of length L_d, k_b is the Boltzmann constant, σ_p is the Peierls stress. The constants C_1, C_2 are given by the relations

$$C_1 := \frac{\rho_d L_d a b^2 \nu}{2w^2}; \quad C_2 := \frac{D}{\rho_d b^2} \tag{3.48}$$

where ρ_d is the dislocation density, L_d is the length of a dislocation segment, a is the distance between Peierls valleys, b is the magnitude of the Burgers' vector, ν is the Debye frequency, w is the width of a kink loop, and D is the drag coefficient.

The inputs for this model are of the form

```
<plasticity_model type="steinberg_cochran_guinan">
  <mu_0> 81.8e9 </mu_0>
  <sigma_0> 1.15e9 </sigma_0>
  <Y_max> 0.25e9 </Y_max>
  <beta> 2.0 </beta>
  <n> 0.50 </n>
  <A> 20.6e-12 </A>
  <B> 0.16e-3 </B>
  <T_m0> 2310.0 </T_m0>
  <Gamma_0> 3.0 </Gamma_0>
  <a> 1.67 </a>
  <epsilon_p0> 0.0 </epsilon_p0>
</plasticity_model>
```

ZA Flow Stress Model

The Zerilli-Armstrong (ZA) model ([Zero4; ZA87; ZA93]) is based on simplified dislocation mechanics. The general form of the equation for the flow stress is

$$\sigma_y(\varepsilon_p, \dot{\varepsilon}_p, T) = \sigma_a + B\exp(-\beta(\dot{\varepsilon}_p)T) + B_0\sqrt{\varepsilon_p}\exp(-\alpha(\dot{\varepsilon}_p)T) \tag{3.49}$$

where σ_a is the athermal component of the flow stress given by

$$\sigma_a := \sigma_g + \frac{k_h}{\sqrt{l}} + K\varepsilon_p^n, \tag{3.50}$$

σ_g is the contribution due to solutes and initial dislocation density, k_h is the microstructural stress intensity, l is the average grain diameter, K is zero for fcc materials, B, B_0 are material constants. The functional forms of the exponents α and β are

$$\alpha = \alpha_0 - \alpha_1\ln(\dot{\varepsilon}_p); \quad \beta = \beta_0 - \beta_1\ln(\dot{\varepsilon}_p); \tag{3.51}$$

where $\alpha_0, \alpha_1, \beta_0, \beta_1$ are material parameters that depend on the type of material (fcc, bcc, hcp, alloys). The Zerilli-Armstrong model has been modified by [AV05] for better performance at high temperatures. However, we have not used the modified equations in our computations.

The inputs for this model are of the form

```
<bcc_or_fcc> fcc </bcc_or_fcc>
<c2> </c2>
<c3> </c3>
<c4> </c4>
<n> </n>
```

MTS Flow Stress Model

The Mechanical Threshold Stress (MTS) model ([FK88; Got+00; Koc01]) gives the following form for the flow stress

$$\sigma_y(\varepsilon_p, \dot{\varepsilon}_p, T) = \sigma_a + (S_i\sigma_i + S_e\sigma_e)\frac{\mu(p,T)}{\mu_0} \tag{3.52}$$

where σ_a is the athermal component of mechanical threshold stress, μ_0 is the shear modulus at 0 K and ambient pressure, σ_i is the component of the flow stress due to intrinsic barriers to thermally activated dislocation motion and dislocation-dislocation interactions, σ_e is the component of the flow stress due to microstructural evolution with increasing deformation (strain hardening), (S_i, S_e) are temperature and strain rate dependent scaling factors. The scaling factors take the Arrhenius form

$$S_i = \left[1 - \left(\frac{k_b\ T}{g_{0i}b^3\mu(p,T)}\ln\frac{\dot{\varepsilon}_{poi}}{\dot{\varepsilon}_p}\right)^{1/q_i}\right]^{1/p_i} \tag{3.53}$$

$$S_e = \left[1 - \left(\frac{k_b\ T}{g_{0e}b^3\mu(p,T)}\ln\frac{\dot{\varepsilon}_{poe}}{\dot{\varepsilon}_p}\right)^{1/q_e}\right]^{1/p_e} \tag{3.54}$$

where k_b is the Boltzmann constant, b is the magnitude of the Burgers' vector, (g_{0i}, g_{0e}) are normalized activation energies, $(\dot{\varepsilon}_{poi}, \dot{\varepsilon}_{poe})$ are constant reference strain rates, and (q_i, p_i, q_e, p_e) are constants. The strain hardening component of the mechanical threshold stress (σ_e) is given by a modified Voce law

$$\frac{d\sigma_e}{d\varepsilon_p} = \theta(\sigma_e) \tag{3.55}$$

where

$$\theta(\sigma_e) = \theta_0[1 - F(\sigma_e)] + \theta_{IV}F(\sigma_e) \tag{3.56}$$

$$\theta_0 = a_0 + a_1\ln\dot{\varepsilon}_p + a_2\sqrt{\dot{\varepsilon}_p} - a_3T \tag{3.57}$$

$$F(\sigma_e) = \frac{\tanh\left(\alpha\frac{\sigma_e}{\sigma_{es}}\right)}{\tanh(\alpha)} \tag{3.58}$$

$$\ln\left(\frac{\sigma_{es}}{\sigma_{0es}}\right) = \left(\frac{kT}{g_{0es}b^3\mu(p,T)}\right)\ln\left(\frac{\dot{\varepsilon}_p}{\dot{\varepsilon}_{poes}}\right) \tag{3.59}$$

and θ_0 is the hardening due to dislocation accumulation, θ_{IV} is the contribution due to stage-IV hardening, $(a_0, a_1, a_2, a_3, \alpha)$ are constants, σ_{es} is the stress at zero strain hardening rate, σ_{0es} is the saturation threshold stress for deformation at 0 K, g_{0es} is a constant, and $\dot{\varepsilon}_{poes}$ is the maximum strain rate. Note that the maximum strain rate is usually limited to about 10^7/s.

The inputs for this model are of the form

```
<plasticity_model type="mts_model">
  <sigma_a>363.7e6</sigma_a>
  <mu_0>28.0e9</mu_0>
  <D>4.50e9</D>
  <T_0>294</T_0>
  <koverbcubed>0.823e6</koverbcubed>
  <g_0i>0.0</g_0i>
  <g_0e>0.71</g_0e>
  <edot_0i>0.0</edot_0i>
  <edot_0e>2.79e9</edot_0e>
  <p_i>0.0</p_i>
  <q_i>0.0</q_i>
```

```
    <p_e>1.0</p_e>
    <q_e>2.0</q_e>
    <sigma_i>0.0</sigma_i>
    <a_0>211.8e6</a_0>
    <a_1>0.0</a_1>
    <a_2>0.0</a_2>
    <a_3>0.0</a_3>
    <theta_IV>0.0</theta_IV>
    <alpha>2</alpha>
    <edot_es0>3.42e8</edot_es0>
    <g_0es>0.15</g_0es>
    <sigma_es0>1679.3e6</sigma_es0>
  </plasticity_model>
```

PTW Flow Stress Model

The Preston-Tonks-Wallace (PTW) model ([PTW03]) attempts to provide a model for the flow stress for extreme strain rates (up to 10^{11}/s) and temperatures up to melt. The flow stress is given by

$$\sigma_y(\varepsilon_p, \dot{\varepsilon}_p, T) = \begin{cases} 2\left[\tau_s + \alpha \ln\left[1 - \varphi \exp\left(-\beta - \dfrac{\theta \varepsilon_p}{\alpha \varphi}\right)\right]\right]\mu(p,T) & \text{thermal regime} \\ 2\tau_s \mu(p,T) & \text{shock regime} \end{cases} \tag{3.60}$$

with

$$\alpha := \frac{s_0 - \tau_y}{d}; \quad \beta := \frac{\tau_s - \tau_y}{\alpha}; \quad \varphi := \exp(\beta) - 1 \tag{3.61}$$

where τ_s is a normalized work-hardening saturation stress, s_0 is the value of τ_s at 0K, τ_y is a normalized yield stress, θ is the hardening constant in the Voce hardening law, and d is a dimensionless material parameter that modifies the Voce hardening law. The saturation stress and the yield stress are given by

$$\tau_s = \max\left\{s_0 - (s_0 - s_\infty)\operatorname{erf}\left[\kappa \hat{T} \ln\left(\frac{\gamma\dot{\xi}}{\dot{\varepsilon}_p}\right)\right], s_0\left(\frac{\dot{\varepsilon}_p}{\gamma\dot{\xi}}\right)^{s_1}\right\} \tag{3.62}$$

$$\tau_y = \max\left\{y_0 - (y_0 - y_\infty)\operatorname{erf}\left[\kappa \hat{T} \ln\left(\frac{\gamma\dot{\xi}}{\dot{\varepsilon}_p}\right)\right], \min\left\{y_1\left(\frac{\dot{\varepsilon}_p}{\gamma\dot{\xi}}\right)^{y_2}, s_0\left(\frac{\dot{\varepsilon}_p}{\gamma\dot{\xi}}\right)^{s_1}\right\}\right\} \tag{3.63}$$

where s_∞ is the value of τ_s close to the melt temperature, (y_0, y_∞) are the values of τ_y at 0K and close to melt, respectively, (κ, γ) are material constants, $\hat{T} = T/T_m$, (s_1, y_1, y_2) are material parameters for the high strain rate regime, and

$$\dot{\xi} = \frac{1}{2}\left(\frac{4\pi\rho}{3M}\right)^{1/3}\left(\frac{\mu(p,T)}{\rho}\right)^{1/2} \tag{3.64}$$

where ρ is the density, and M is the atomic mass.

The inputs for this model are of the form

```
  <plasticity_model type="preston_tonks_wallace">
    <theta> 0.025 </theta>
    <p> 2.0 </p>
    <s0> 0.0085 </s0>
    <sinf> 0.00055 </sinf>
    <kappa> 0.11 </kappa>
    <gamma> 0.00001 </gamma>
    <y0> 0.0001 </y0>
    <yinf> 0.0001 </yinf>
    <y1> 0.094 </y1>
```

```
    <y2> 0.575 </y2>
    <beta> 0.25 </beta>
    <M> 63.54 </M>
    <G0> 518e8 </G0>
    <alpha> 0.20 </alpha>
    <alphap> 0.20 </alphap>
  </plasticity_model>
```

Adiabatic Heating and Specific Heat

A part of the plastic work done is converted into heat and used to update the temperature of a particle. The increase in temperature (ΔT) due to an increment in plastic strain ($\Delta\epsilon_p$) is given by the equation

$$\Delta T = \frac{\chi\sigma_y}{\rho C_p}\Delta\epsilon_p \tag{3.65}$$

where χ is the Taylor-Quinney coefficient, and C_p is the specific heat. The value of the Taylor-Quinney coefficient is taken to be 0.9 in all our simulations (see [Rav+01] for more details on the variation of χ with strain and strain rate).

The Taylor-Quinney coefficient is taken as input in the ElasticPlastic model using the tags

```
<taylor_quinney_coeff> 0.9 </taylor_quinney_coeff>
```

Default specific heat model

The default model returns a constant specific heat and is invoked using

```
<specific_heat_model type="constant_Cp">
</specific_heat_model>
```

Specific heat model for copper

The specific heat model for copper is of the form

$$Cp = \begin{cases} A_0\ T^3 - B_0\ T^2 + C_0\ T - D_0 & \text{if } T < T_0 \\ A_1\ T + B_1 & \text{if } T \geq T_0\ . \end{cases} \tag{3.66}$$

The model is invoked using

```
<specific_heat_model type = "copper_Cp"> </specific_heat_model>
```

Specific heat model for steel

A relation for the dependence of C_p upon temperature is used for the steel ([LSS74]).

$$C_p = \begin{cases} A_1 + B_1\ t + C_1\ |t|^{-\alpha} & \text{if } T < T_c \\ A_2 + B_2\ t + C_2\ t^{-\alpha'} & \text{if } T > T_c \end{cases} \tag{3.67}$$

$$t = \frac{T}{T_c} - 1 \tag{3.68}$$

where T_c is the critical temperature at which the phase transformation from the α to the γ phase takes place, and $A_1, A_2, B_1, B_2, \alpha, \alpha'$ are constants.

The model is invoked using

```
<specific_heat_model type = "steel_Cp"> </specific_heat_model>
```

The heat generated at a material point is conducted away at the end of a time step using the transient heat equation. The effect of conduction on material point temperature is negligible (but non-zero) for the high strain-rate problems simulated using Uintah.

3.2.4 Adding new models

In the parallel implementation of the stress update algorithm, sockets have been added to allow for the incorporation of a variety of plasticity, damage, yield, and bifurcation models without requiring any change in the stress update code. The algorithm is shown in Algorithm 3.1. The equation of state, plasticity model, yield condition, damage model, and the stability criterion are all polymorphic objects created using a factory idiom in C++ ([Cop92]).

Addition of a new model requires the following steps (the example below is only for the flow stress model but the same idea applies to other models) :

1. Creation of a new class that encapsulates the plasticity model. The template for this class can be copied from the existing plasticity models. The data that is unique to the new model are specified in the form of
   - A structure containing the constants for the plasticity model.
   - Particle variables that specify the variables that evolve in the plasticity model.
2. The implementation of the plasticity model involves the following steps.
   - Reading the input file for the model constants in the constructor.
   - Adding the variables that evolve in the plasticity model appropriately to the task graph.
   - Adding the appropriate flow stress calculation method.
3. The `PlasticityModelFactory` is then modified so that it recognizes the added plasticity model.

3.2.5 Damage Models and Failure

Only the Johnson-Cook damage evolution rule has been added to the DamageModelFactory so far. The damage model framework is designed to be similar to the plasticity model framework. New models can be added using the approach described in the previous section.

A particle is tagged as "failed" when its temperature is greater than the melting point of the material at the applied pressure. An additional condition for failure is when the porosity of a particle increases beyond a critical limit and the strain exceeds the fracture strain of the material. Another condition for failure is when a material bifurcation condition such as the Drucker stability postulate is satisfied. Upon failure, a particle is either removed from the computation by setting the stress to zero or is converted into a material with a different velocity field which interacts with the remaining particles via contact. Either approach leads to the simulation of a newly created surface. More details of the approach can be found in [Ban04a; Ban04b; Ban05].

Yield conditions

When failure is to be simulated we can use the Gurson-Tvergaard-Needleman yield condition instead of the von Mises condition.

The von Mises yield condition

The von Mises yield condition is the default and is invoked using the tags

```
<yield_condition type="vonMises">
</yield_condition>
```

The Gurson-Tvergaard-Needleman (GTN) yield condition

The Gurson-Tvergaard-Needleman (GTN) yield condition [Gur77; TN84] depends on porosity. An associated flow rule is used to determine the plastic rate parameter in either case. The GTN yield condition can be written as

$$\Phi = \left(\frac{\sigma_{eq}}{\sigma_f}\right)^2 + 2q_1 f_* \cosh\left(q_2 \frac{Tr(\sigma)}{2\sigma_f}\right) - (1 + q_3 f_*^2) = 0 \tag{3.69}$$

Table 3.1: Stress Update Algorithm

Persistent:Initial moduli, temperature, porosity,
scalar damage, equation of state, plasticity model,
yield condition, stability criterion, damage model
Temporary:Particle state at time t
Output: Particle state at time $t + \Delta t$

For *all the patches in the domain*
Read the particle data and initialize updated data storage
For *all the particles in the patch*
Compute the velocity gradient and the rate of deformation tensor
Compute the deformation gradient and the rotation tensor
Rotate the Cauchy stress and the rate of deformation tensor
to the material configuration
Compute the current shear modulus and melting temperature
Compute the pressure using the equation of state,
update the hydrostatic stress, and
compute the trial deviatoric stress
Compute the flow stress using the plasticity model
Evaluate the yield function
If *particle is elastic*
Update the elastic deviatoric stress from the trial stress
Rotate the stress back to laboratory coordinates
Update the particle state
Else
Compute the elastic-plastic deviatoric stress
Compute updated porosity, scalar damage, and
temperature increase due to plastic work
Compute elastic-plastic tangent modulus and evaluate stability condition
Rotate the stress back to laboratory coordinates
Update the particle state
End If
If *Temperature* > *Melt Temperature* or *Porosity* > *Critical Porosity* or *Unstable*
Tag particle as failed
End If
Convert failed particles into a material with a different velocity field
End For
End For

where q_1, q_2, q_3 are material constants and f_* is the porosity (damage) function given by

$$f* = \begin{cases} f & \text{for } f \le f_c, \\ f_c + k(f - f_c) & \text{for } f > f_c \end{cases} \tag{3.70}$$

where k is a constant and f is the porosity (void volume fraction). The flow stress in the matrix material is computed using either of the two plasticity models discussed earlier. Note that the flow stress in the matrix material also remains on the undamaged matrix yield surface and uses an associated flow rule.

This yield condition is invoked using

```
<yield_condition type="gurson">
  <q1> 1.5 </q1>
  <q2> 1.0 </q2>
  <q3> 2.25 </q3>
  <k> 4.0 </k>
  <f_c> 0.05 </f_c>
</yield_condition>
```

Porosity model

The evolution of porosity is calculated as the sum of the rate of growth and the rate of nucleation [RA98b]. The rate of growth of porosity and the void nucleation rate are given by the following equations [CN80]

$$\dot{f} = \dot{f}_{\text{nucl}} + \dot{f}_{\text{grow}} \tag{3.71}$$

$$\dot{f}_{\text{grow}} = (1 - f)\text{Tr}(\boldsymbol{D}_p) \tag{3.72}$$

$$\dot{f}_{\text{nucl}} = \frac{f_n}{(s_n\sqrt{2\pi})} \exp\left[-\frac{1}{2}\frac{(\epsilon_p - \epsilon_n)^2}{s_n^2}\right]\dot{\epsilon}_p \tag{3.73}$$

where $\boldsymbol{D}_p$ is the rate of plastic deformation tensor, f_n is the volume fraction of void nucleating particles , ϵ_n is the mean of the distribution of nucleation strains, and s_n is the standard deviation of the distribution.

The inputs tags for porosity are of the form

```
<evolve_porosity> true </evolve_porosity>
<initial_mean_porosity>          0.005 </initial_mean_porosity>
<initial_std_porosity>           0.001 </initial_std_porosity>
<critical_porosity>              0.3   </critical_porosity>
<frac_nucleation>                0.1   </frac_nucleation>
<meanstrain_nucleation>          0.3   </meanstrain_nucleation>
<stddevstrain_nucleation>        0.1   </stddevstrain_nucleation>
<initial_porosity_distrib>       gauss </initial_porosity_distrib>
```

Damage model

After the stress state has been determined on the basis of the yield condition and the associated flow rule, a scalar damage state in each material point can be calculated using the Johnson-Cook model [JC85]. The Johnson-Cook model has an explicit dependence on temperature, plastic strain, ans strain rate.

The damage evolution rule for the Johnson-Cook damage model can be written as

$$\dot{D} = \frac{\dot{\epsilon_p}}{\epsilon_p^f}; \quad \epsilon_p^f = \left[D_1 + D_2 \exp\left(\frac{D_3}{3}\sigma^*\right)\right]\left[1 + D_4 \ln(\dot{\epsilon_p}^*)\right]\left[1 + D_5 T^*\right] \; ; \quad \sigma^* = \frac{\text{Tr}(\boldsymbol{\sigma})}{\sigma_{eq}}; \tag{3.74}$$

where D is the damage variable which has a value of 0 for virgin material and a value of 1 at fracture, ϵ_p^f is the fracture strain, D_1, D_2, D_3, D_4, D_5 are constants, $\boldsymbol{\sigma}$ is the Cauchy stress, and T^* is the scaled temperature as in the Johnson-Cook plasticity model.

The input tags for the damage model are :

```
<damage_model type="johnson_cook">
  <D1>0.05</D1>
  <D2>3.44</D2>
  <D3>-2.12</D3>
  <D4>0.002</D4>
  <D5>0.61</D5>
</damage_model>
```

An initial damage distribution can be created using the following tags

```
<evolve_damage>                  true   </evolve_damage>
<initial_mean_scalar_damage>     0.005   </initial_mean_scalar_damage>
<initial_std_scalar_damage>      0.001 </initial_std_scalar_damage>
<critical_scalar_damage>         1.0    </critical_scalar_damage>
<initial_scalar_damage_distrib> gauss </initial_scalar_damage_distrib>
```

3.2.6 Erosion algorithm

Under normal conditions, the heat generated at a material point is conducted away at the end of a time step using the heat equation. If special adiabatic conditions apply (such as in impact problems), the heat is accumulated at a material point and is not conducted to the surrounding particles. This localized heating can be used to determine whether a material point has melted.

The determination of whether a particle has failed can be made on the basis of either or all of the following conditions:

- The particle temperature exceeds the melting temperature.
- The TEPLA-F fracture condition [JA88] is satisfied. This condition can be written as

$$(f/f_c)^2 + (\epsilon_p/\epsilon_p^f)^2 = 1 \tag{3.75}$$

  where f is the current porosity, f_c is the maximum allowable porosity, ϵ_p is the current plastic strain, and ϵ_p^f is the plastic strain at fracture.
- An alternative to ad-hoc damage criteria is to use the concept of bifurcation to determine whether a particle has failed or not. Two stability criteria have been explored in this paper - the Drucker stability postulate [Dru59] and the loss of hyperbolicity criterion (using the determinant of the acoustic tensor) [Per98; RR75].

The simplest criterion that can be used is the Drucker stability postulate [Dru59] which states that time rate of change of the rate of work done by a material cannot be negative. Therefore, the material is assumed to become unstable (and a particle fails) when

$$\dot{\boldsymbol{\sigma}} : \boldsymbol{D}^p \leq 0 \tag{3.76}$$

Another stability criterion that is less restrictive is the acoustic tensor criterion which states that the material loses stability if the determinant of the acoustic tensor changes sign [Per98; RR75]. Determination of the acoustic tensor requires a search for a normal vector around the material point and is therefore computationally expensive. A simplification of this criterion is a check which assumes that the direction of instability lies in the plane of the maximum and minimum principal stress [Bec02]. In this approach, we assume that the strain is localized in a band with normal $\mathbf{n}$, and the magnitude of the velocity difference across the band is $\mathbf{g}$. Then the bifurcation condition leads to the relation

$$R_{ij}g_j = 0\,; \quad R_{ij} = M_{ikjl}n_k n_l + M_{ilkj}n_k n_l - \sigma_{ik}n_j n_k \tag{3.77}$$

where M_{ijkl} are the components of the co-rotational tangent modulus tensor and σ_{ij} are the components of the co-rotational stress tensor. If $\det(R_{ij}) \leq 0$, then g_j can be arbitrary and there is a possibility of strain localization. If this condition for loss of hyperbolicity is met, then a particle deforms in an unstable

manner and failure can be assumed to have occurred at that particle. We use a combination of these criteria to simulate failure.

Since the material in the container may unload locally after fracture, the hypoelastic-plastic stress update may not work accurately under certain circumstances. An improvement would be to use a hyperelastic-plastic stress update algorithm. Also, the plasticity models are temperature dependent. Hence there is the issue of severe mesh dependence due to change of the governing equations from hyperbolic to elliptic in the softening regime [BB85; HH75; TN90]. Viscoplastic stress update models or nonlocal/gradient plasticity models [HLQ00; RA98a] can be used to eliminate some of these effects and are currently under investigation.

The tags used to control the erosion algorithm are in two places. In the `<MPM> </MPM>` section the following flags can be set

```
<erosion algorithm = "ZeroStress"/>
<create_new_particles>           false      </create_new_particles>
<manual_new_material>            false      </manual_new_material>
```

If the erosion algorithm is `"none"` then no particle failure is done.

In the `<constitutive_model type="elastic_plastic">` section, the following flags can be set

```
<evolve_porosity>                true  </evolve_porosity>
<evolve_damage>                  true  </evolve_damage>
<do_melting>                     true  </do_melting>
<useModifiedEOS>                 true  </useModifiedEOS>
<check_TEPLA_failure_criterion> true  </check_TEPLA_failure_criterion>
<check_max_stress_failure>       false </check_max_stress_failure>
<critical_stress>               12.0e9 </critical_stress>
```

3.2.7 Implementation

The elastic response is assumed to be isotropic. The material constants that are taken as input for the elastic response are the bulk and shear modulus. The flow rule is determined from the input and the appropriate plasticity model is created using the `PlasticityModelFactory` class. The damage evolution rule is determined from the input and a damage model is created using the `DamageModelFactory` class. The equation of state that is used to determine the pressure is also determined from the input. The equation of state model is created using the `MPMEquationOfStateFactory` class.

In addition, a damage evolution variable (D) is stored at each time step (this need not be the case and will be transfered to the damage models in the future). The left stretch and rotation are updated incrementally at each time step (instead of performing a polar decomposition) and the rotation tensor is used to rotate the Cauchy stress and rate of deformation to the material coordinates at each time step (instead of using a objective stress rate formulation).

Any evolution variables for the plasticity model, damage model or the equation of state are specified in the class that encapsulates the particular model.

The flow stress is calculated from the plasticity model using a function call of the form

```
double flowStress = d_plasticity->computeFlowStress(tensorEta, tensorS,
                                                    pTemperature[idx],
                                                    delT, d_tol, matl, idx)
                                                        ;
```

A number of plasticity models can be evaluated using the inputs in the `computeFlowStress` call. The variable `d_plasticity` is polymorphic and can represent any of the plasticity models that can be created by the plasticity model factory. The plastic evolution variables are updated using a polymorphic function along the lines of `computeFlowStress`.

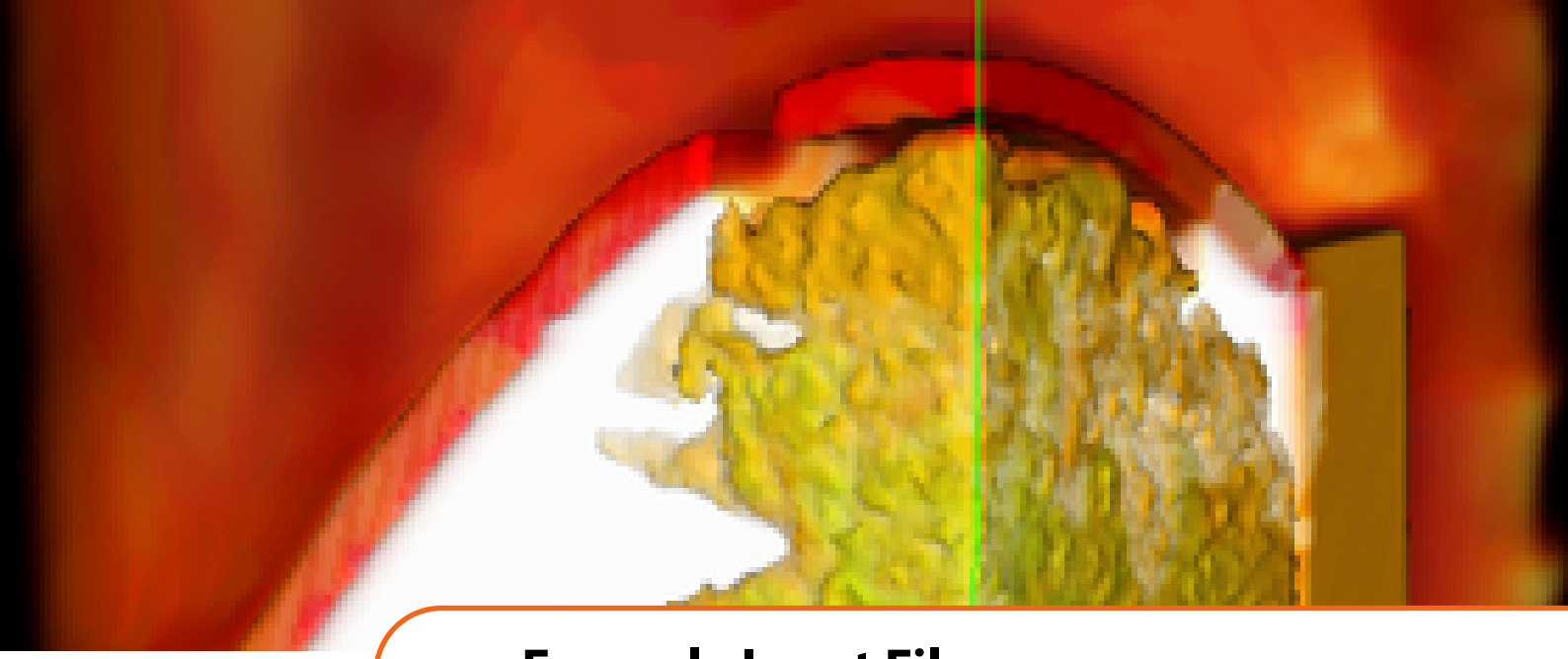
The equation of state is used to calculate the hydrostatic stress using a function call of the form

```
Matrix3 tensorHy = d_eos->computePressure(matl, bulk, shear,
                                          tensorF_new, tensorD,
                                          tensorP, pTemperature[idx],
                                          rho_cur, delT);
```

Similarly, the damage model is called using a function of the type

```
double damage = d_damage->computeScalarDamage(tensorEta, tensorS,
                                              pTemperature[idx],
                                              delT, matl, d_tol,
                                              pDamage[idx]);
```

Therefore, the plasticity, damage and equation of state models are easily be inserted into any other type of stress update algorithm without any change being needed in them as can be seen in the hyperelastic-plastic stress update algorithm discussed below.



4 — Example Input Files

4.1 Hypoelastic-plastic model

An example of the portion of an input file that specifies a copper body with a hypoelastic stress update, Johnson-Cook plasticity model, Johnson-Cook Damage Model and Mie-Gruneisen Equation of State is shown below.

```
<material>

  <include href="inputs/MPM/MaterialData/MaterialConstAnnCopper.xml"/>
  <constitutive_model type="hypoelastic_plastic">
    <tolerance>5.0e-10</tolerance>
    <include href="inputs/MPM/MaterialData/IsotropicElasticAnnCopper.xml"/>
    <include href="inputs/MPM/MaterialData/JohnsonCookPlasticAnnCopper.xml"/>
    <include href="inputs/MPM/MaterialData/JohnsonCookDamageAnnCopper.xml"/>
    <include href="inputs/MPM/MaterialData/MieGruneisenEOSAnnCopper.xml"/>
  </constitutive_model>

  <burn type = "null" />
  <velocity_field>1</velocity_field>

  <geom_object>
    <cylinder label = "Cylinder">
      <bottom>[0.0,0.0,0.0]</bottom>
      <top>[0.0,2.54e-2,0.0]</top>
      <radius>0.762e-2</radius>
    </cylinder>
    <res>[3,3,3]</res>
    <velocity>[0.0,-208.0,0.0]</velocity>
    <temperature>294</temperature>
  </geom_object>

</material>
```

The general material constants for copper are in the file `MaterialConstAnnCopper.xml`. The contents are shown below

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<Uintah_Include>
  <density>8930.0</density>
  <toughness>10.e6</toughness>
  <thermal_conductivity>1.0</thermal_conductivity>
```

```
    <specific_heat>383</specific_heat>
    <room_temp>294.0</room_temp>
    <melt_temp>1356.0</melt_temp>
  </Uintah_Include>
```

The elastic properties are in the file `IsotropicElasticAnnCopper.xml`. The contents of this file are shown below.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<Uintah_Include>
  <shear_modulus>45.45e9</shear_modulus>
  <bulk_modulus>136.35e9</bulk_modulus>
</Uintah_Include>
```

The constants for the Johnson-Cook plasticity model are in the file `JohnsonCookPlasticAnnCopper.xml`. The contents of this file are shown below.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<Uintah_Include>
  <plasticity_model type="johnson_cook">
    <A>89.6e6</A>
    <B>292.0e6</B>
    <C>0.025</C>
    <n>0.31</n>
    <m>1.09</m>
  </plasticity_model>
</Uintah_Include>
```

The constants for the Johnson-Cook damage model are in the file `JohnsonCookDamageAnnCopper.xml`. The contents of this file are shown below.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<Uintah_Include>
  <damage_model type="johnson_cook">
    <D1>0.54</D1>
    <D2>4.89</D2>
    <D3>-3.03</D3>
    <D4>0.014</D4>
    <D5>1.12</D5>
  </damage_model>
</Uintah_Include>
```

The constants for the Mie-Gruneisen model (as implemented in the Uintah Computational Framework) are in the file `MieGruneisenEOSAnnCopper.xml`. The contents of this file are shown below.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<Uintah_Include>
  <equation_of_state type="mie_gruneisen">
    <C_0>3940</C_0>
    <Gamma_0>2.02</Gamma_0>
    <S_alpha>1.489</S_alpha>
  </equation_of_state>
</Uintah_Include>
```

As can be seen from the input file, any other plasticity model, damage model and equation of state can be used to replace the Johnson-Cook and Mie-Gruneisen models without any extra effort (provided the models have been implemented and the data exist).

The material data can easily be taken from a material database or specified for a new material in an input file kept at a centralized location. At this stage material data for a range of materials is kept in the directory `.../Uintah/StandAlone/inputs/MPM/MaterialData`.

4.2 Elastic-plastic model

The <constitutive_model type="elastic_plastic"> model is more stable (and also more general) than the <constitutive_model type="hypoelastic_plastic"> model. A sample input file for this model is shown below.

```
<MPM>
  <do_grid_reset> false </do_grid_reset>
  <time_integrator>explicit</time_integrator>
  <boundary_traction_faces>[zminus,zplus]</boundary_traction_faces>
  <dynamic>true</dynamic>
  <solver>simple</solver>
  <convergence_criteria_disp>1.e-10</convergence_criteria_disp>
  <convergence_criteria_energy>4.e-10</convergence_criteria_energy>
  <DoImplicitHeatConduction>true</DoImplicitHeatConduction>
  <interpolator>linear</interpolator>
  <minimum_particle_mass> 1.0e-8</minimum_particle_mass>
  <maximum_particle_velocity> 1.0e8</maximum_particle_velocity>
  <artificial_damping_coeff> 0.0 </artificial_damping_coeff>
  <artificial_viscosity> true </artificial_viscosity>
  <accumulate_strain_energy> true </accumulate_strain_energy>
  <use_load_curves> false </use_load_curves>
  <turn_on_adiabatic_heating>   false    </turn_on_adiabatic_heating>
  <do_contact_friction_heating> false    </do_contact_friction_heating>
  <create_new_particles>        false    </create_new_particles>
  <erosion algorithm = "none"/>
</MPM>

<MaterialProperties>
  <MPM>
    <material name = "OFHCCu">
      <density>              8930.0 </density>
      <thermal_conductivity> 386.0  </thermal_conductivity>
      <specific_heat>        414.0  </specific_heat>
      <room_temp>            294.0  </room_temp>
      <melt_temp>            1356.0 </melt_temp>
      <constitutive_model type="elastic_plastic">
        <isothermal>                    false   </isothermal>
        <tolerance>                     1.0e-12 </tolerance>
        <do_melting>                    false   </do_melting>
        <evolve_porosity>               false   </evolve_porosity>
        <evolve_damage>                 false   </evolve_damage>
        <check_TEPLA_failure_criterion> false   </
            check_TEPLA_failure_criterion>
        <check_max_stress_failure>      false   </check_max_stress_failure>
        <initial_material_temperature>  696.0   </
            initial_material_temperature>

        <shear_modulus>           46.0e9  </shear_modulus>
        <bulk_modulus>            129.0e9 </bulk_modulus>
        <coeff_thermal_expansion> 1.76e-5 </coeff_thermal_expansion>
        <taylor_quinney_coeff>    0.9     </taylor_quinney_coeff>
        <critical_stress>         129.0e9 </critical_stress>

        <equation_of_state type = "mie_gruneisen">
          <C_0>     3940  </C_0>
          <Gamma_0> 2.02  </Gamma_0>
          <S_alpha> 1.489 </S_alpha>
        </equation_of_state>

        <plasticity_model type="mts_model">
          <sigma_a>40.0e6</sigma_a>
          <mu_0>47.7e9</mu_0>
```

```
            <D>3.0e9</D>
            <T_0>180</T_0>
            <koverbcubed>0.823e6</koverbcubed>
            <g_0i>0.0</g_0i>
            <g_0e>1.6</g_0e>
            <edot_0i>0.0</edot_0i>
            <edot_0e>1.0e7</edot_0e>
            <p_i>0.0</p_i>
            <q_i>0.0</q_i>
            <p_e>0.666667</p_e>
            <q_e>1.0</q_e>
            <sigma_i>0.0</sigma_i>
            <a_0>2390.0e6</a_0>
            <a_1>12.0e6</a_1>
            <a_2>1.696e6</a_2>
            <a_3>0.0</a_3>
            <theta_IV>0.0</theta_IV>
            <alpha>2</alpha>
            <edot_es0>1.0e7</edot_es0>
            <g_0es>0.2625</g_0es>
            <sigma_es0>770.0e6</sigma_es0>
          </plasticity_model>

          <shear_modulus_model type="mts_shear">
            <mu_0>47.7e9</mu_0>
            <D>3.0e9</D>
            <T_0>180</T_0>
          </shear_modulus_model>

          <melting_temp_model type = "constant_Tm">
          </melting_temp_model>

          <yield_condition type = "vonMises">
          </yield_condition>

          <stability_check type = "none">
          </stability_check>

          <damage_model type = "hancock_mackenzie">
            <D0> 0.0001 </D0>
            <Dc> 0.7    </Dc>
          </damage_model>

          <compute_specfic_heat> false </compute_specfic_heat>
          <specific_heat_model type="constant_Cp">
          </specific_heat_model>

        </constitutive_model>
        <geom_object>
          <box label = "box">
            <min>[0.0,    0.0,    0.0]</min>
            <max>[1.0e-2, 1.0e-2, 1.0e-2]</max>
          </box>
          <res>[1,1,1]</res>
          <velocity>[0.0, 0.0, 0.0]</velocity>
          <temperature>696</temperature>
        </geom_object>
      </material>

    </MPM>
  </MaterialProperties>
```

4.2.1 An exploding ring experiment

The follwing shows the complete input file for an expanding ring test.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<Uintah_specification>
<!--Please use a consistent set of units, (mks, cgs,...)-->
    <!-- First crack at the tuna can problem -->

   <Meta>
       <title>Pressurization of a container via burning w/o fracture</title>
   </Meta>&gt;
   <SimulationComponent>
     <type> mpmice </type>
   </SimulationComponent>
    <!--______________________________________________________________-->
    <!--   T  I  M  E     V  A  R  I  A  B  L  E  S                    -->
    <!--______________________________________________________________-->
   <Time>
       <max_Timesteps>   99999          </max_Timesteps>>
       <maxTime>             2.00e-2    </maxTime>
       <initTime>            0.0        </initTime>
       <delt_min>            1.0e-12    </delt_min>
       <delt_max>            1.0        </delt_max>
       <delt_init>           2.1e-8     </delt_init>
       <timestep_multiplier> 0.5        </timestep_multiplier>
   </Time>
    <!--______________________________________________________________-->
    <!--   G  R  I  D     V  A  R  I  A  B  L  E  S                    -->
    <!--______________________________________________________________-->
    <Grid>
    <BoundaryConditions>
      <Face side = "x-">
        <BCType id = "all" label = "Symmetric" var = "symmetry">
        </BCType>
      </Face>
      <Face side = "x+">
        <BCType id = "0" label = "Pressure" var = "Neumann">
                            <value>  0.0 </value>
        </BCType>
        <BCType id = "all" label = "Velocity" var = "Dirichlet">
                            <value> [0.,0.,0.] </value>
        </BCType>
        <BCType id = "all" label = "Temperature" var = "Neumann">
                            <value> 0.0  </value>
        </BCType>
        <BCType id = "all" label = "Density" var = "Neumann">
                            <value> 0.0  </value>
        </BCType>
      </Face>
      <Face side = "y-">
        <BCType id = "all" label = "Symmetric" var = "symmetry">
        </BCType>
      </Face>
      <Face side = "y+">
        <BCType id = "0" label = "Pressure" var = "Neumann">
                            <value> 0.0   </value>
        </BCType>
        <BCType id = "all" label = "Velocity" var = "Dirichlet">
                            <value> [0.,0.,0.] </value>
        </BCType>
        <BCType id = "all" label = "Temperature" var = "Neumann">
                            <value> 0.0  </value>
```

```
      </BCType>
      <BCType id = "all" label = "Density" var = "Neumann">
                           <value> 0.0  </value>
      </BCType>
    </Face>
    <Face side = "z-">
      <BCType id = "all" label = "Symmetric" var = "symmetry">
      </BCType>
    </Face>
    <Face side = "z+">
      <BCType id = "all" label = "Symmetric" var = "symmetry">
      </BCType>
    </Face>
  </BoundaryConditions>
     <Level>
         <Box label = "1">
            <lower>        [ -0.08636, -0.08636, -0.0016933] </lower>
            <upper>        [ 0.08636, 0.08636, 0.0016933]    </upper>
            <extraCells>   [1,1,1]    </extraCells>
            <patches>      [2,2,1]    </patches>
            <resolution>   [102, 102, 1]                 </resolution>
         </Box>
     </Level>
  </Grid>

  <!--__________________________________________________________________-->
  <!--   O  U  P  U  T     V  A  R  I  A  B  L  E  S                     -->
  <!--__________________________________________________________________-->
<DataArchiver>
   <filebase>explodeRFull.uda</filebase>
   <outputTimestepInterval> 20 </outputTimestepInterval>
   <save label = "rho_CC"/>
   <save label = "press_CC"/>
   <save label = "temp_CC"/>
   <save label = "vol_frac_CC"/>
   <save label = "vel_CC"/>
   <save label = "g.mass"/>
   <save label = "p.x"/>
   <save label = "p.mass"/>
   <save label = "p.temperature"/>
   <save label = "p.porosity"/>
   <save label = "p.particleID"/>
   <save label = "p.velocity"/>
   <save label = "p.stress"/>
   <save label = "p.damage" material = "0"/>
   <save label = "p.plasticStrain" material = "0"/>
   <save label = "p.strainRate" material = "0"/>
   <save label = "g.stressFS"/>
   <save label = "delP_Dilatate"/>
   <save label = "delP_MassX"/>
   <save label = "p.localized"/>
   <checkpoint cycle = "2" timestepInterval = "20"/>
</DataArchiver>

<Debug>
</Debug>
  <!--__________________________________________________________________-->
  <!--    I  C  E     P  A  R  A  M  E  T  E  R  S                      -->
  <!--__________________________________________________________________-->
  <CFD>
     <cfl>0.5</cfl>
     <CanAddICEMaterial>true</CanAddICEMaterial>
     <ICE>
```

```
    <advection type = "SecondOrder"/>
    <ClampSpecificVolume>true</ClampSpecificVolume>
  </ICE>
</CFD>

<!--__________________________________________________________________-->
<!--     P H Y S I C A L     C O N S T A N T S                        -->
<!--__________________________________________________________________-->
<PhysicalConstants>
   <gravity>            [0,0,0]   </gravity>
   <reference_pressure> 101325.0  </reference_pressure>
</PhysicalConstants>

<MPM>
  <time_integrator>                explicit   </time_integrator>
  <nodes8or27>                     27         </nodes8or27>
  <minimum_particle_mass>          3.e-12     </minimum_particle_mass>
  <maximum_particle_velocity>      1.e3       </maximum_particle_velocity>
  <artificial_damping_coeff>       0.0        </artificial_damping_coeff>
  <artificial_viscosity>           true       </artificial_viscosity>
  <artificial_viscosity_coeff1>    0.07       </artificial_viscosity_coeff1
     >
  <artificial_viscosity_coeff2>    1.6        </artificial_viscosity_coeff2
     >
  <turn_on_adiabatic_heating>      false      </turn_on_adiabatic_heating>
  <accumulate_strain_energy>       false      </accumulate_strain_energy>
  <use_load_curves>                false      </use_load_curves>
  <create_new_particles>           false      </create_new_particles>
  <manual_new_material>            false      </manual_new_material>
  <DoThermalExpansion>             false      </DoThermalExpansion>
  <testForNegTemps_mpm>            false      </testForNegTemps_mpm>
  <erosion algorithm = "ZeroStress"/>
</MPM>

<!--__________________________________________________________________-->
<!--    MATERIAL PROPERTIES INITIAL CONDITIONS                        -->
<!--__________________________________________________________________-->
<MaterialProperties>
   <MPM>
     <material name = "Steel Ring">
       <include href="inputs/MPM/MaterialData/MatConst4340St.xml"/>
       <constitutive_model type="elastic_plastic">
         <isothermal>                    false </isothermal>
         <tolerance>                  1.0e-10  </tolerance>
         <evolve_porosity>               true  </evolve_porosity>
         <evolve_damage>                 true  </evolve_damage>
         <compute_specific_heat>         true  </compute_specific_heat>
         <do_melting>                    true  </do_melting>
         <useModifiedEOS>                true  </useModifiedEOS>
         <check_TEPLA_failure_criterion> true  </
             check_TEPLA_failure_criterion>
         <initial_material_temperature>  600.0 </
             initial_material_temperature>
         <taylor_quinney_coeff>          0.9   </taylor_quinney_coeff>
         <check_max_stress_failure>      false </check_max_stress_failure>
         <critical_stress>              12.0e9 </critical_stress>

         <!-- Warning: you must copy link this input file into your -->
         <!-- sus directory or these paths won't work.              -->

         <include href="inputs/MPM/MaterialData/IsoElastic4340St.xml"/>
         <include href="inputs/MPM/MaterialData/MieGrunEOS4340St.xml"/>
         <include href="inputs/MPM/MaterialData/ConstantShear.xml"/>
```

```
      <include href="inputs/MPM/MaterialData/ConstantTm.xml"/>
      <include href="inputs/MPM/MaterialData/JCPlastic4340St.xml"/>
      <include href="inputs/MPM/MaterialData/VonMisesYield.xml"/>
      <include href="inputs/MPM/MaterialData/DruckerBeckerStabilityCheck
          .xml"/>
      <include href="inputs/MPM/MaterialData/JCDamage4340St.xml"/>
      <specific_heat_model type="steel_Cp"> </specific_heat_model>

      <initial_mean_porosity>          0.005 </initial_mean_porosity>
      <initial_std_porosity>           0.001 </initial_std_porosity>
      <critical_porosity>              0.3   </critical_porosity>
      <frac_nucleation>                0.1   </frac_nucleation>
      <meanstrain_nucleation>          0.3   </meanstrain_nucleation>
      <stddevstrain_nucleation>        0.1   </stddevstrain_nucleation>
      <initial_porosity_distrib>       gauss </initial_porosity_distrib>

      <initial_mean_scalar_damage>     0.005  </
          initial_mean_scalar_damage>
      <initial_std_scalar_damage>      0.001 </initial_std_scalar_damage>
      <critical_scalar_damage>         1.0   </critical_scalar_damage>
      <initial_scalar_damage_distrib> gauss </
          initial_scalar_damage_distrib>
    </constitutive_model>

        <geom_object>
           <difference>
            <cylinder label = "outer cylinder">
              <bottom>           [0.0,0.0,-.05715]   </bottom>
              <top>              [0.0,0.0, .05715]   </top>
              <radius>           0.05715            </radius>
            </cylinder>
            <cylinder label = "inner cylinder">
              <bottom>           [0.0,0.0,-.0508]   </bottom>
              <top>              [0.0,0.0, .0508]   </top>
              <radius>           0.0508             </radius>
            </cylinder>
            </difference>
          <res>                 [2,2,2]          </res>
          <velocity>            [0.0,0.0,0.0]    </velocity>
          <temperature>         600              </temperature>
        </geom_object>
</material>
<material name = "reactant">
    <include href="inputs/MPM/MaterialData/MatConstPBX9501.xml"/>
    <constitutive_model type = "visco_scram">
      <include href="inputs/MPM/MaterialData/ViscoSCRAMPBX9501.xml"/>
      <include href="inputs/MPM/MaterialData/TimeTempPBX9501.xml"/>
      <randomize_parameters>          false </randomize_parameters>
      <use_time_temperature_equation> true  </
          use_time_temperature_equation>
      <useObjectiveRate>              true  </useObjectiveRate>
      <useModifiedEOS>                true  </useModifiedEOS>
    </constitutive_model>
       <geom_object>
          <difference>
            <cylinder label = "inner cylinder"> </cylinder>
            <cylinder label = "inner hole">
              <bottom>           [0.0,0.0,-.0508]   </bottom>
              <top>              [0.0,0.0, .0508]   </top>
              <radius>           0.01               </radius>
            </cylinder>
          </difference>
          <res>                 [2,2,2]          </res>
```

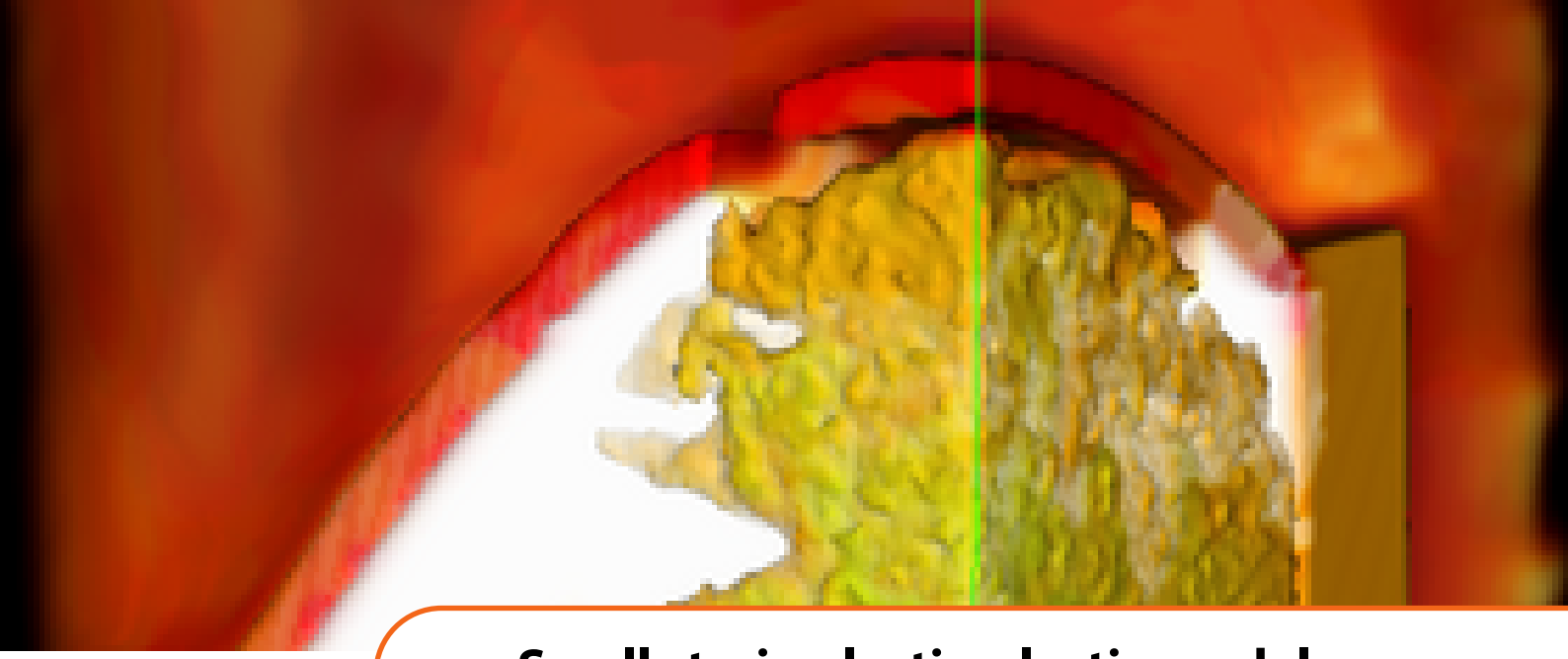
```
                <velocity>            [0.0,0.0,0.0]   </velocity>
                <temperature>         440.0           </temperature>
              </geom_object>
      </material>

         <contact>
           <type>approach</type>
           <materials>              [0,1]          </materials>
           <mu> 0.0 </mu>
         </contact>
         <thermal_contact>
         </thermal_contact>
    </MPM>

     <ICE>
       <material>
         <EOS type = "ideal_gas">
         </EOS>
         <dynamic_viscosity>          0.0             </dynamic_viscosity>
         <thermal_conductivity>       0.0             </thermal_conductivity>
         <specific_heat>              716.0           </specific_heat>
         <gamma>                      1.4             </gamma>
         <geom_object>
           <difference>
                <box>
                  <min>           [-0.254,-0.254,-0.254] </min>
                  <max>           [ 0.254, 0.254, 0.254] </max>
                </box>
               <cylinder label = "outer cylinder"> </cylinder>
             </difference>
           <cylinder label="inner hole"> </cylinder>
           <res>                        [2,2,2]         </res>
           <velocity>           [0.0,0.0,0.0]       </velocity>
           <!--
           <temperature>        300.0               </temperature>
           <density>    1.1792946927374306000e+00   </density>
           -->
           <temperature>        400.0               </temperature>
           <density>            0.884471019553073   </density>
           <pressure>           101325.0             </pressure>
         </geom_object>
       </material>
    </ICE>

    <exchange_properties>
       <exchange_coefficients>
            <momentum>  [0, 1e15, 1e15]     </momentum>
            <heat>      [0, 1e10, 1e10]     </heat>
      </exchange_coefficients>
    </exchange_properties>
  </MaterialProperties>

  <AddMaterialProperties>
    <ICE>
       <material name = "product">
         <EOS type = "ideal_gas">
         </EOS>
         <dynamic_viscosity>          0.0             </dynamic_viscosity>
         <thermal_conductivity>       0.0             </thermal_conductivity>
         <specific_heat>              716.0           </specific_heat>
         <gamma>                      1.4             </gamma>
         <geom_object>
                <box>
```

```
                    <min>           [ 1.0, 1.0, 1.0] </min>
                    <max>           [ 2.0, 2.0, 2.0] </max>
                  </box>
              <res>                       [2,2,2]        </res>
              <velocity>            [0.0,0.0,0.0]       </velocity>
              <temperature>         300.0               </temperature>
              <density>    1.17929469273743060000e+00   </density>
              <pressure>            101325.0             </pressure>
            </geom_object>
          </material>
      </ICE>

      <exchange_properties>
         <exchange_coefficients>
              <momentum>  [0, 1e15, 1e15, 1e15, 1e15, 1e15]     </momentum>
              <heat>      [0, 1e10, 1e10, 1e10, 1e10, 1e10]     </heat>
              <!--
              <heat>      [0, 1, 1, 1, 1, 1]     </heat>
              -->
        </exchange_coefficients>
      </exchange_properties>
    </AddMaterialProperties>


    <Models>
      <Model type="Simple_Burn">
        <Active>       false        </Active>
        <fromMaterial> reactant     </fromMaterial>
        <toMaterial>   product      </toMaterial>
        <ThresholdTemp>       450.0 </ThresholdTemp>
        <ThresholdPressure> 50000.0 </ThresholdPressure>
        <Enthalpy>        2000000.0 </Enthalpy>
        <BurnCoeff>            75.3 </BurnCoeff>
        <refPressure>      101325.0 </refPressure>
      </Model>
    </Models>


</Uintah_specification>
```

The PBS script used to run this test is

```
#
# ASK PBS TO SEND YOU AN EMAIL ON CERTAIN EVENTS: (a)bort (b)egin (e)nd (n)ever
#
# (User May Change)

#PBS -m abe

#
# SET THE NAME OF THE JOB:
#
# (User May Change)

#PBS -N ExplodeRing

#
# SET THE QUEUE IN WHICH TO RUN THE JOB.
#      (Note, there is currently only one queue, so you should never change
    this field.)

#PBS -q defaultq
```

```
#
# SET THE RESOURCES (# NODES, TIME) REQESTED FROM THE BATCH SCHEDULER:
#   - select: <# nodes>,ncpus=2,walltime=<time>
#     - walltime: walltime before PBS kills our job.
#                 [[hours:]minutes:]seconds[.milliseconds]
#                 Examples:
#                   walltime=60      (60 seconds)
#                   walltime=10:00   (10 minutes)
#                   walltime=5:00:00 (5 hours)
#
# (User May Change)

#PBS -l select=2:ncpus=2,walltime=24:00

#
# START UP LAM

cd $PBS_O_WORKDIR
lamboot

# [place your command here] >& ${PBS_O_WORKDIR}/output.${PBS_JOBID}
mpirun -np 4 ../sus_opt explodeRFull.ups >& output.${PBS_JOBID}

#
# REMEMBER, IF YOU ARE RUNNING TWO SERIAL JOBS, YOU NEED A:
# wait

#
# STOP LAM

lamhalt -v
exit
```



5 — Small strain elastic-plastic model

5.1 Preamble

Let $\boldsymbol{F}$ be the deformation gradient, $\boldsymbol{\sigma}$ be the Cauchy stress, and $\boldsymbol{d}$ be the rate of deformation tensor. We first decompose the deformation gradient into a stretch and a rotations using $\boldsymbol{F} = \boldsymbol{R} \cdot \boldsymbol{U}$. The rotation $\boldsymbol{R}$ is then used to rotate the stress and the rate of deformation into the material configuration to give us

$$\hat{\boldsymbol{\sigma}} = \boldsymbol{R}^T \cdot \boldsymbol{\sigma} \cdot \boldsymbol{R} \,; \quad \hat{\boldsymbol{d}} = \boldsymbol{R}^T \cdot \boldsymbol{d} \cdot \boldsymbol{R} \tag{5.1}$$

This is equivalent to using a Green-Naghdi objective stress rate. In the following all equations are with respect to the hatted quantities and we drop the hats for convenience.

5.2 Elastic relation

Let us split the Cauchy stress into a volumetric and a deviatoric part

$$\boldsymbol{\sigma} = p\,\boldsymbol{1} + \boldsymbol{s} \,; \quad p = \frac{1}{3}\,\mathrm{tr}(\boldsymbol{\sigma}) \,. \tag{5.2}$$

Taking the time derivative gives us

$$\dot{\boldsymbol{\sigma}} = \dot{p}\,\boldsymbol{1} + \dot{\boldsymbol{s}} \,. \tag{5.3}$$

We assume that the elastic response of the material is isotropic. The constitutive relation for a hypoelastic material of grade o can be expressed as

$$\dot{\boldsymbol{\sigma}} = \left[\lambda\,\mathrm{tr}(\boldsymbol{d}^e) - 3\,\kappa\,\alpha\,\frac{d}{dt}(T - T_0)\right]\boldsymbol{1} + 2\,\mu\,\boldsymbol{d}^e \,; \quad \boldsymbol{d} = \boldsymbol{d}^e + \boldsymbol{d}^p \tag{5.4}$$

where $\boldsymbol{d}^e$, $\boldsymbol{d}^p$ are the elastic and plastic parts of the rate of deformation tensor, λ, μ are the Lame constants, κ is the bulk modulus, α is the coefficient of thermal expansion, T_0 is the reference temperature, and T is the current temperature. If we split $\boldsymbol{d}^e$ into volumetric and deviatoric parts as

$$\boldsymbol{d}^e = \frac{1}{3}\,\mathrm{tr}(\boldsymbol{d}^e)\,\boldsymbol{1} + \boldsymbol{\eta}^e \tag{5.5}$$

we can write

$$\dot{\boldsymbol{\sigma}} = \left[\left(\lambda + \frac{2}{3}\,\mu\right)\,\mathrm{tr}(\boldsymbol{d}^e) - 3\,\kappa\,\alpha\,\frac{d}{dt}(T - T_0)\right]\boldsymbol{1} + 2\,\mu\,\boldsymbol{\eta}^e = \kappa\left[\mathrm{tr}(\boldsymbol{d}^e) - 3\,\alpha\,\frac{d}{dt}(T - T_0)\right]\boldsymbol{1} + 2\,\mu\,\boldsymbol{\eta}^e \tag{5.6}$$

Therefore, we have

$$\dot{\boldsymbol{s}} = 2\,\mu\,\boldsymbol{\eta}^e\ . \tag{5.7}$$

and

$$\dot{p} = \kappa\left[\mathrm{tr}(\boldsymbol{d}^e) - 3\,\alpha\,\frac{d}{dt}(T - T_o)\right]\ . \tag{5.8}$$

We will use a standard elastic-plastic stress update algorithm to integrate the rate equation for the deviatoric stress. However, we will assume that the volumetric part of the Cauchy stress can be computed using an equation of state. Then the final Cauchy stress will be given by

$$\boldsymbol{\sigma} = \left[p(J) - 3\,J\,\frac{dp(J)}{dJ}\,\alpha\,(T - T_o)\right]\boldsymbol{1} + \boldsymbol{s}\,;\quad J = \det(\boldsymbol{F})\ . \tag{5.9}$$

(Note that we assume that the plastic part of the deformation is volume preserving. This is not true for Gurson type models and will lead to a small error in the computed value of $\boldsymbol{\sigma}$.)

5.3 Flow rule

We assume that the flow rule is given by

$$\boldsymbol{d}^p = \dot{\gamma}\,\boldsymbol{r} \tag{5.10}$$

We can split $\boldsymbol{d}^p$ into a trace part and a trace free part, i.e.,

$$\boldsymbol{d}^p = \frac{1}{3}\,\mathrm{tr}(\boldsymbol{d}^p)\,\boldsymbol{1} + \boldsymbol{\eta}^p \tag{5.11}$$

Then, using the flow rule, we have

$$\boldsymbol{d}^p = \frac{1}{3}\,\dot{\gamma}\,\mathrm{tr}(\boldsymbol{r})\,\boldsymbol{1} + \boldsymbol{\eta}^p\ . \tag{5.12}$$

Therefore we can write the flow rule as

$$\boldsymbol{\eta}^p = \dot{\gamma}\,\left(-\frac{1}{3}\,\mathrm{tr}(\boldsymbol{r})\,\boldsymbol{1} + \boldsymbol{r}\right)\ . \tag{5.13}$$

Note that

$$\boldsymbol{d} = \boldsymbol{d}^e + \boldsymbol{d}^p \quad\Longrightarrow\quad \mathrm{tr}(\boldsymbol{d}) = \mathrm{tr}(\boldsymbol{d}^e) + \mathrm{tr}(\boldsymbol{d}^p)\ . \tag{5.14}$$

Also,

$$\boldsymbol{d} = \boldsymbol{d}^e + \boldsymbol{d}^p \quad\Longrightarrow\quad \frac{1}{3}\,\mathrm{tr}(\boldsymbol{d})\,\boldsymbol{1} + \boldsymbol{\eta} = \frac{1}{3}\,\mathrm{tr}(\boldsymbol{d}^e)\,\boldsymbol{1} + \boldsymbol{\eta}^e + \frac{1}{3}\,\mathrm{tr}(\boldsymbol{d}^p)\,\boldsymbol{1} + \boldsymbol{\eta}^p\ . \tag{5.15}$$

Therefore,

$$\boldsymbol{\eta} = \boldsymbol{\eta}^e + \boldsymbol{\eta}^p\ . \tag{5.16}$$

5.4 Isotropic and Kinematic hardening and porosity evolution rules

We assume that the strain rate, temperature, and porosity can be fixed at the beginning of a timestep and consider only the evolution of plastic strain and the back stress while calculating the current stress.

We assume that the plastic strain evolves according to the relation

$$\dot{\varepsilon^p} = \dot{\gamma}\, h^{\alpha} \tag{5.17}$$

We also assume that the back stress evolves according to the relation

$$\dot{\hat{\boldsymbol{\beta}}} = \dot{\gamma}\, \boldsymbol{h}^{\beta} \tag{5.18}$$

where $\hat{\boldsymbol{\beta}}$ is the back stress. If $\boldsymbol{\beta}$ is the deviatoric part of $\hat{\boldsymbol{\beta}}$, then we can write

$$\dot{\boldsymbol{\beta}} = \dot{\gamma}\, \mathrm{dev}(\boldsymbol{h}^{\beta})\,. \tag{5.19}$$

The porosity ϕ is assumed to evolve according to the relation

$$\dot{\phi} = \dot{\gamma}\, h^{\phi}\,. \tag{5.20}$$

5.5 Yield condition

The yield condition is assumed to be of the form

$$f(\boldsymbol{s}, \boldsymbol{\beta}, \varepsilon^p, \phi, \dot{\varepsilon}, T, \dots) = f(\boldsymbol{\xi}, \varepsilon^p, \phi, \dot{\varepsilon}, T, \dots) = 0 \tag{5.21}$$

where $\boldsymbol{\xi} = \boldsymbol{s} - \boldsymbol{\beta}$ and $\boldsymbol{\beta}$ is the deviatoric part of $\hat{\boldsymbol{\beta}}$. The Kuhn-Tucker loading-unloading conditions are

$$\dot{\gamma} \geq 0\,;\quad f \leq 0\,;\quad \dot{\gamma}\, f = 0 \tag{5.22}$$

and the consistency condition is $\dot{f} = 0$.

5.6 Temperature increase due to plastic dissipation

The temperature increase due to plastic dissipation is assume to be given by the rate equation

$$\dot{T} = \frac{\chi}{\rho\, C_p}\, \sigma_y\, \dot{\varepsilon^p}\,. \tag{5.23}$$

The temperature is updated using

$$T_{n+1} = T_n + \frac{\chi_{n+1}\, \Delta t}{\rho_{n+1}\, C_p}\, \sigma_y^{n+1}\, \dot{\varepsilon^p}_{n+1}\,. \tag{5.24}$$

5.7 Continuum elastic-plastic tangent modulus

To determine whether the material has undergone a loss of stability we need to compute the acosutic tensor which needs the computation of the continuum elastic-plastic tangent modulus.

To do that recall that

$$\boldsymbol{\sigma} = p\, \boldsymbol{1} + \boldsymbol{s} \quad \Longrightarrow \quad \dot{\boldsymbol{\sigma}} = \dot{p}\, \boldsymbol{1}\,. \tag{5.25}$$

We assume that

$$\dot{p} = J\,\frac{\partial p}{\partial J}\,\mathrm{tr}(\boldsymbol{d}) \qquad \text{and} \quad \dot{\boldsymbol{s}} = 2\,\mu\,\boldsymbol{\eta}^e\,. \tag{5.26}$$

Now, the consistency condition requires that

$$\dot{f}(\boldsymbol{s}, \boldsymbol{\beta}, \varepsilon^p, \phi, \dot{\varepsilon}, T, \dots) = 0\,. \tag{5.27}$$

Keeping $\dot{\varepsilon}$ and T fixed over the time interval, we can use the chain rule to get

$$\dot{f} = \frac{\partial f}{\partial \boldsymbol{s}} : \dot{\boldsymbol{s}} + \frac{\partial f}{\partial \boldsymbol{\beta}} : \dot{\boldsymbol{\beta}} + \frac{\partial f}{\partial \varepsilon^p}\,\dot{\varepsilon^p} + \frac{\partial f}{\partial \phi}\,\dot{\phi} = 0\,. \tag{5.28}$$

The needed rate equations are

$$\begin{aligned}
\dot{\boldsymbol{s}} &= 2\,\mu\,\boldsymbol{\eta}^e = 2\,\mu\,(\boldsymbol{\eta} - \boldsymbol{\eta}^p) = 2\,\mu\,[\boldsymbol{\eta} - \dot{\gamma}\,\mathrm{dev}(\boldsymbol{r})] \\
\dot{\boldsymbol{\beta}} &= \dot{\gamma}\,\mathrm{dev}(\boldsymbol{h}^\beta) \\
\dot{\varepsilon^p} &= \dot{\gamma}\,h^\alpha \\
\dot{\phi} &= \dot{\gamma}\,h^\phi
\end{aligned} \tag{5.29}$$

Plugging these into the expression for $\dot{f}$ gives

$$2\,\mu\,\frac{\partial f}{\partial \boldsymbol{s}} : [\boldsymbol{\eta} - \dot{\gamma}\,\mathrm{dev}(\boldsymbol{r})] + \dot{\gamma}\,\frac{\partial f}{\partial \boldsymbol{\beta}} : \mathrm{dev}(\boldsymbol{h}^\beta) + \dot{\gamma}\,\frac{\partial f}{\partial \varepsilon^p}\,h^\alpha + \dot{\gamma}\,\frac{\partial f}{\partial \phi}\,h^\phi = 0 \tag{5.30}$$

or,

$$\dot{\gamma} = \frac{2\,\mu\,\dfrac{\partial f}{\partial \boldsymbol{s}} : \boldsymbol{\eta}}{2\,\mu\,\dfrac{\partial f}{\partial \boldsymbol{s}} : \mathrm{dev}(\boldsymbol{r}) - \dfrac{\partial f}{\partial \boldsymbol{\beta}} : \mathrm{dev}(\boldsymbol{h}^\beta) - \dfrac{\partial f}{\partial \varepsilon^p}\,h^\alpha - \dfrac{\partial f}{\partial \phi}\,h^\phi}\,. \tag{5.31}$$

Plugging this expression for $\dot{\gamma}$ into the equation for $\dot{\boldsymbol{s}}$, we get

$$\dot{\boldsymbol{s}} = 2\,\mu\left[\boldsymbol{\eta} - \left(\frac{2\,\mu\,\dfrac{\partial f}{\partial \boldsymbol{s}} : \boldsymbol{\eta}}{2\,\mu\,\dfrac{\partial f}{\partial \boldsymbol{s}} : \mathrm{dev}(\boldsymbol{r}) - \dfrac{\partial f}{\partial \boldsymbol{\beta}} : \mathrm{dev}(\boldsymbol{h}^\beta) - \dfrac{\partial f}{\partial \varepsilon^p}\,h^\alpha - \dfrac{\partial f}{\partial \phi}\,h^\phi}\right)\mathrm{dev}(\boldsymbol{r})\right]. \tag{5.32}$$

At this stage, note that a symmetric $\boldsymbol{\sigma}$ implies a symmetric $\boldsymbol{s}$ and hence a symmetric $\boldsymbol{\eta}$. Also we assume that $\boldsymbol{r}$ is symmetric (and hence $\mathrm{dev}(\boldsymbol{r})$), which is true if the flow rule is associated. Then we can write

$$\boldsymbol{\eta} = \mathsf{I}^{4s} : \boldsymbol{\eta} \qquad \text{and} \qquad \mathrm{dev}(\boldsymbol{r}) = \mathsf{I}^{4s} : \mathrm{dev}(\boldsymbol{r}) \tag{5.33}$$

where I^{4s} is the fourth-order symmetric identity tensor. Also note that if $\boldsymbol{A}$, $\boldsymbol{C}$, $\boldsymbol{D}$ are second order tensors and B is a fourth order tensor, then

$$(\boldsymbol{A} : \mathsf{B} : \boldsymbol{C})\,(\mathsf{B} : \boldsymbol{D}) \equiv A_{ij}\,B_{ijkl}\,C_{kl}\,B_{mnpq}\,D_{pq} = (B_{mnpq}\,D_{pq})\,(A_{ij}\,B_{ijkl})\,C_{kl} \equiv [(\mathsf{B} : \boldsymbol{D}) \otimes (\boldsymbol{A} : \mathsf{B})] : \boldsymbol{C}\,. \tag{5.34}$$

Therefore we have

$$\dot{\boldsymbol{s}} = 2\,\mu\left[\mathsf{I}^{4s} : \boldsymbol{\eta} - \left(\frac{2\,\mu\,[\mathsf{I}^{4s} : \mathrm{dev}(\boldsymbol{r})] \otimes [\dfrac{\partial f}{\partial \boldsymbol{s}} : \mathsf{I}^{4s}]}{2\,\mu\,\dfrac{\partial f}{\partial \boldsymbol{s}} : \mathrm{dev}(\boldsymbol{r}) - \dfrac{\partial f}{\partial \boldsymbol{\beta}} : \mathrm{dev}(\boldsymbol{h}^\beta) - \dfrac{\partial f}{\partial \varepsilon^p}\,h^\alpha - \dfrac{\partial f}{\partial \phi}\,h^\phi}\right) : \boldsymbol{\eta}\right]. \tag{5.35}$$

Also,

$$\mathbf{I}^{4s} : \text{dev}(\boldsymbol{r}) = \text{dev}(\boldsymbol{r}) \qquad \text{and} \qquad \frac{\partial f}{\partial \boldsymbol{s}} : \mathbf{I}^{4s} = \frac{\partial f}{\partial \boldsymbol{s}} \,. \tag{5.36}$$

Hence we can write

$$\dot{\boldsymbol{s}} = 2\,\mu \left[\mathbf{I}^{4s} - \left(\frac{2\,\mu\,\text{dev}(\boldsymbol{r}) \otimes \frac{\partial f}{\partial \boldsymbol{s}}}{2\,\mu\,\frac{\partial f}{\partial \boldsymbol{s}} : \text{dev}(\boldsymbol{r}) - \frac{\partial f}{\partial \boldsymbol{\beta}} : \text{dev}(\boldsymbol{h}^{\beta}) - \frac{\partial f}{\partial \varepsilon^{p}}\, h^{\alpha} - \frac{\partial f}{\partial \phi}\, h^{\phi}} \right) \right] : \boldsymbol{\eta} \tag{5.37}$$

or,

$$\dot{\boldsymbol{s}} = \mathbf{B}^{ep} : \boldsymbol{\eta} = \mathbf{B}^{ep} : \left[\boldsymbol{d} - \frac{1}{3}\,\text{tr}(\boldsymbol{d})\,\boldsymbol{1} \right] \tag{5.38}$$

where

$$\mathbf{B}^{ep} := 2\,\mu \left[\mathbf{I}^{4s} - \left(\frac{2\,\mu\,\text{dev}(\boldsymbol{r}) \otimes \frac{\partial f}{\partial \boldsymbol{s}}}{2\,\mu\,\frac{\partial f}{\partial \boldsymbol{s}} : \text{dev}(\boldsymbol{r}) - \frac{\partial f}{\partial \boldsymbol{\beta}} : \text{dev}(\boldsymbol{h}^{\beta}) - \frac{\partial f}{\partial \varepsilon^{p}}\, h^{\alpha} - \frac{\partial f}{\partial \phi}\, h^{\phi}} \right) \right] . \tag{5.39}$$

Adding in the volumetric component gives

$$\begin{aligned} \dot{\boldsymbol{\sigma}} &= \dot{p}\,\boldsymbol{1} + \dot{\boldsymbol{s}} \\ &= J\,\frac{\partial p}{\partial J}\,\text{tr}(\boldsymbol{d})\,\boldsymbol{1} + \mathbf{B}^{ep} : \left[\boldsymbol{d} - \frac{1}{3}\,\text{tr}(\boldsymbol{d})\,\boldsymbol{1} \right] \\ &= \left[3\,J\,\frac{\partial p}{\partial J}\,\boldsymbol{1} - \mathbf{B}^{ep} : \boldsymbol{1} \right] \frac{\boldsymbol{d} : \boldsymbol{1}}{3} + \mathbf{B}^{ep} : \boldsymbol{d} \\ &= J\,\frac{\partial p}{\partial J}\,(\boldsymbol{1} \otimes \boldsymbol{1}) : \boldsymbol{d} - \frac{1}{3}\left[\mathbf{B}^{ep} : (\boldsymbol{1} \otimes \boldsymbol{1}) \right] : \boldsymbol{d} + \mathbf{B}^{ep} : \boldsymbol{d} \,. \end{aligned} \tag{5.40}$$

Therefore,

$$\dot{\boldsymbol{\sigma}} = \left[J\,\frac{\partial p}{\partial J}\,(\boldsymbol{1} \otimes \boldsymbol{1}) - \frac{1}{3}\left[\mathbf{B}^{ep} : (\boldsymbol{1} \otimes \boldsymbol{1}) \right] + \mathbf{B}^{ep} \right] : \boldsymbol{d} = \mathbf{C}^{ep} : \boldsymbol{d} \,. \tag{5.41}$$

The quantity $\mathbf{C}^{ep}$ is the continuum elastic-plastic tangent modulus. We also use the continuum elastic-plastic tangent modulus in the implicit version of the code. However, for improved accuracy and faster convergence, an algorithmically consistent tangent modulus should be used instead. That tangent modulus can be calculated in the usual manner and is left for development and implementation as an additional feature in the future.

5.8 Stress update

A standard return algorithm is used to compute the updated Cauchy stress. Recall that the rate equation for the deviatoric stress is given by

$$\dot{\boldsymbol{s}} = 2\,\mu\,\boldsymbol{\eta}^{e} \,. \tag{5.42}$$

Integrating the rate equation using a Backward Euler scheme gives

$$\boldsymbol{s}_{n+1} - \boldsymbol{s}_{n} = 2\,\mu\,\Delta\,t\,\boldsymbol{\eta}^{e}_{n+1} = 2\,\mu\,\Delta\,t\,(\boldsymbol{\eta}_{n+1} - \boldsymbol{\eta}^{p}_{n+1}) \tag{5.43}$$

Now, from the flow rule, we have

$$\boldsymbol{\eta}^p = \dot{\gamma} \left(\boldsymbol{r} - \frac{1}{3} \operatorname{tr}(\boldsymbol{r}) \, \boldsymbol{1} \right) . \tag{5.44}$$

Define the deviatoric part of $\boldsymbol{r}$ as

$$\operatorname{dev}(\boldsymbol{r}) := \boldsymbol{r} - \frac{1}{3} \operatorname{tr}(\boldsymbol{r}) \, \boldsymbol{1} . \tag{5.45}$$

Therefore,

$$\boldsymbol{s}_{n+1} - \boldsymbol{s}_n = 2 \, \mu \, \Delta \, t \, \boldsymbol{\eta}_{n+1} - 2 \, \mu \, \Delta\gamma_{n+1} \operatorname{dev}(\boldsymbol{r}_{n+1}) . \tag{5.46}$$

where $\Delta\gamma := \dot{\gamma} \, \Delta t$. Define the trial stress

$$\boldsymbol{s}^{\text{trial}} := \boldsymbol{s}_n + 2 \, \mu \, \Delta \, t \, \boldsymbol{\eta}_{n+1} . \tag{5.47}$$

Then

$$\boldsymbol{s}_{n+1} = \boldsymbol{s}^{\text{trial}} - 2 \, \mu \, \Delta\gamma_{n+1} \operatorname{dev}(\boldsymbol{r}_{n+1}) . \tag{5.48}$$

Also recall that the back stress is given by

$$\dot{\boldsymbol{\beta}} = \dot{\gamma} \operatorname{dev} \boldsymbol{h}^{\beta} \tag{5.49}$$

The evolution equation for the back stress can be integrated to get

$$\boldsymbol{\beta}_{n+1} - \boldsymbol{\beta}_n = \Delta\gamma_{n+1} \operatorname{dev}(\boldsymbol{h})^{\beta}_{n+1} . \tag{5.50}$$

Now,

$$\boldsymbol{\xi}_{n+1} = \boldsymbol{s}_{n+1} - \boldsymbol{\beta}_{n+1} . \tag{5.51}$$

Plugging in the expressions for $\boldsymbol{s}_{n+1}$ and $\boldsymbol{\beta}_{n+1}$, we get

$$\boldsymbol{\xi}_{n+1} = \boldsymbol{s}^{\text{trial}} - 2 \, \mu \, \Delta\gamma_{n+1} \operatorname{dev}(\boldsymbol{r}_{n+1}) - \boldsymbol{\beta}_n - \Delta\gamma_{n+1} \operatorname{dev}(\boldsymbol{h})^{\beta}_{n+1} . \tag{5.52}$$

Define

$$\boldsymbol{\xi}^{\text{trial}} := \boldsymbol{s}^{\text{trial}} - \boldsymbol{\beta}_n . \tag{5.53}$$

Then

$$\boldsymbol{\xi}_{n+1} = \boldsymbol{\xi}^{\text{trial}} - \Delta\gamma_{n+1} (2 \, \mu \operatorname{dev}(\boldsymbol{r}_{n+1}) + \operatorname{dev}(\boldsymbol{h})^{\beta}_{n+1}) . \tag{5.54}$$

Similarly, the evolution of the plastic strain is given by

$$\varepsilon^p_{n+1} = \varepsilon^p_n + \Delta\gamma_{n+1} \, h^{\alpha}_{n+1} \tag{5.55}$$

and the porosity evolves as

$$\phi_{n+1} = \phi_n + \Delta\gamma_{n+1} \, h^{\phi}_{n+1} . \tag{5.56}$$

The yield condition is discretized as

$$f(\boldsymbol{s}_{n+1}, \boldsymbol{\beta}_{n+1}, \varepsilon^p_{n+1}, \phi_{n+1}, \dot{\varepsilon}_{n+1}, T_{n+1}, \dots) = f(\boldsymbol{\xi}_{n+1}, \varepsilon^p_{n+1}, \phi_{n+1}, \dot{\varepsilon}_{n+1}, T_{n+1}, \dots) = 0 . \tag{5.57}$$

Important: We assume that the derivatives with respect to $\dot{\varepsilon}$ and T are small enough to be neglected.

5.8.1 Newton iterations

We now have the following equations that have to be solved for $\Delta\gamma_{n+1}$:

$$\begin{aligned}
\boldsymbol{\xi}_{n+1} &= \boldsymbol{\xi}^{\text{trial}} - \Delta\gamma_{n+1}(2\,\mu\,\text{dev}(\boldsymbol{r}_{n+1}) + \text{dev}(\boldsymbol{h})^{\beta}_{n+1}) \\
\varepsilon^{p}_{n+1} &= \varepsilon^{p}_{n} + \Delta\gamma_{n+1}\,h^{\alpha}_{n+1} \\
\phi_{n+1} &= \phi_{n} + \Delta\gamma_{n+1}\,h^{\phi}_{n+1} \\
f(\boldsymbol{\xi}_{n+1}, \varepsilon^{p}_{n+1}, \phi_{n+1}, \dot{\varepsilon}_{n+1}, T_{n+1}, \dots) &= 0\,.
\end{aligned} \tag{5.58}$$

Recall that if $g(\Delta\gamma) = 0$ is a nonlinear equation that we have to solve for $\Delta\gamma$, an iterative Newton method can be expressed as

$$\Delta\gamma^{(k+1)} = \Delta\gamma^{(k)} - \left[\frac{dg}{d\Delta\gamma}\right]^{-1}_{(k)} g^{(k)}\,. \tag{5.59}$$

Define

$$\delta\gamma := \Delta\gamma^{(k+1)} - \Delta\gamma^{(k)}\,. \tag{5.60}$$

Then, the iterative scheme can be written as

$$g^{(k)} + \left[\frac{dg}{d\Delta\gamma}\right]^{(k)} \delta\gamma = 0\,. \tag{5.61}$$

In our case we have

$$\begin{aligned}
\mathbf{a}(\Delta\gamma) &= 0 = -\boldsymbol{\xi} + \boldsymbol{\xi}^{\text{trial}} - \Delta\gamma(2\,\mu\,\text{dev}(\boldsymbol{r}) + \text{dev}(\boldsymbol{h})^{\beta}) \\
b(\Delta\gamma) &= 0 = -\varepsilon^{p} + \varepsilon^{p}_{n} + \Delta\gamma\,h^{\alpha} \\
c(\Delta\gamma) &= 0 = -\phi + \phi_{n} + \Delta\gamma\,h^{\phi} \\
f(\Delta\gamma) &= 0 = f(\boldsymbol{\xi}, \varepsilon^{p}, \phi, \dot{\varepsilon}, T, \dots)
\end{aligned} \tag{5.62}$$

Therefore,

$$\begin{aligned}
\frac{d\mathbf{a}}{d\Delta\gamma} &= -\frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma} - (2\,\mu\,\text{dev}(\boldsymbol{r}) + \text{dev}(\boldsymbol{h})^{\beta}) - \Delta\gamma\left(2\,\mu\,\frac{\partial\text{dev}(\boldsymbol{r})}{\partial\Delta\gamma} + \frac{\partial\text{dev}(\boldsymbol{h})^{\beta}}{\partial\Delta\gamma}\right) \\
&= -\frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma} - (2\,\mu\,\text{dev}(\boldsymbol{r}) + \text{dev}(\boldsymbol{h})^{\beta}) - \Delta\gamma\left(2\,\mu\,\frac{\partial\text{dev}(\boldsymbol{r})}{\partial\boldsymbol{\xi}} : \frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma} + 2\,\mu\,\frac{\partial\text{dev}(\boldsymbol{r})}{\partial\varepsilon^{p}}\,\frac{\partial\varepsilon^{p}}{\partial\Delta\gamma} + 2\,\mu\,\frac{\partial\text{dev}(\boldsymbol{r})}{\partial\phi}\,\frac{\partial\phi}{\partial\Delta\gamma} + \right. \\
&\qquad \left. \frac{\partial\text{dev}(\boldsymbol{h})^{\beta}}{\partial\boldsymbol{\xi}} : \frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma} + \frac{\partial\text{dev}(\boldsymbol{h})^{\beta}}{\partial\varepsilon^{p}}\,\frac{\partial\varepsilon^{p}}{\partial\Delta\gamma} + \frac{\partial\text{dev}(\boldsymbol{h})^{\beta}}{\partial\phi}\,\frac{\partial\phi}{\partial\Delta\gamma}\right) \\
\frac{db}{d\Delta\gamma} &= -\frac{\partial\varepsilon^{p}}{\partial\Delta\gamma} + h^{\alpha} + \Delta\gamma\left(\frac{\partial h^{\alpha}}{\partial\boldsymbol{\xi}} : \frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma} + \frac{\partial h^{\alpha}}{\partial\varepsilon^{p}}\,\frac{\partial\varepsilon^{p}}{\partial\Delta\gamma} + \frac{\partial h^{\alpha}}{\partial\phi}\,\frac{\partial\phi}{\partial\Delta\gamma}\right) \\
\frac{dc}{d\Delta\gamma} &= -\frac{\partial\phi}{\partial\Delta\gamma} + h^{\phi} + \Delta\gamma\left(\frac{\partial h^{\phi}}{\partial\boldsymbol{\xi}} : \frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma} + \frac{\partial h^{\phi}}{\partial\varepsilon^{p}}\,\frac{\partial\varepsilon^{p}}{\partial\Delta\gamma} + \frac{\partial h^{\phi}}{\partial\phi}\,\frac{\partial\phi}{\partial\Delta\gamma}\right) \\
\frac{df}{d\Delta\gamma} &= \frac{\partial f}{\partial\boldsymbol{\xi}} : \frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma} + \frac{\partial f}{\partial\varepsilon^{p}}\,\frac{\partial\varepsilon^{p}}{\partial\Delta\gamma} + \frac{\partial f}{\partial\phi}\,\frac{\partial\phi}{\partial\Delta\gamma}\,.
\end{aligned} \tag{5.63}$$

Now, define

$$\Delta\boldsymbol{\xi} := \frac{\partial\boldsymbol{\xi}}{\partial\Delta\gamma}\,\delta\gamma\,;\quad \Delta\varepsilon^{p} := \frac{\partial\varepsilon^{p}}{\partial\Delta\gamma}\,\delta\gamma\,;\quad \Delta\phi := \frac{\partial\phi}{\partial\Delta\gamma}\,\delta\gamma\,. \tag{5.64}$$

Then

$$\begin{aligned}
&\mathbf{a}^{(k)} - \Delta\boldsymbol{\xi} - \left[2\,\mu\,\mathrm{dev}(\boldsymbol{r}^{(k)}) + \mathrm{dev}(\boldsymbol{h})^{\beta(k)}\right]\delta\gamma \\
&\qquad - 2\,\mu\,\Delta\gamma\left(\frac{\partial \mathrm{dev}(\boldsymbol{r}^{(k)})}{\partial\boldsymbol{\xi}} : \Delta\boldsymbol{\xi} + \frac{\partial \mathrm{dev}(\boldsymbol{r}^{(k)})}{\partial\varepsilon^p}\,\Delta\varepsilon^p + \frac{\partial \mathrm{dev}(\boldsymbol{r}^{(k)})}{\partial\phi}\,\Delta\phi\right) \\
&\qquad - \Delta\gamma\left(\frac{\partial \mathrm{dev}(\boldsymbol{h})^{\beta(k)}}{\partial\boldsymbol{\xi}} : \Delta\boldsymbol{\xi} + \frac{\partial \mathrm{dev}(\boldsymbol{h})^{\beta(k)}}{\partial\varepsilon^p}\,\Delta\varepsilon^p + \frac{\partial \mathrm{dev}(\boldsymbol{h})^{\beta(k)}}{\partial\phi}\,\Delta\phi\right) = 0 \\
&b^{(k)} - \Delta\varepsilon^p + h^\alpha\,\delta\gamma + \Delta\gamma\left(\frac{\partial h^{\alpha(k)}}{\partial\boldsymbol{\xi}} : \Delta\boldsymbol{\xi} + \frac{\partial h^{\alpha(k)}}{\partial\varepsilon^p}\,\Delta\varepsilon^p + \frac{\partial h^{\alpha(k)}}{\partial\phi}\,\Delta\phi\right) = 0 \\
&c^{(k)} - \Delta\phi + h^\phi\,\delta\gamma + \Delta\gamma\left(\frac{\partial h^{\phi(k)}}{\partial\boldsymbol{\xi}} : \Delta\boldsymbol{\xi} + \frac{\partial h^{\phi(k)}}{\partial\varepsilon^p}\,\Delta\varepsilon^p + \frac{\partial h^{\phi(k)}}{\partial\phi}\,\Delta\phi\right) = 0 \\
&f^{(k)} + \frac{\partial f^{(k)}}{\partial\boldsymbol{\xi}} : \Delta\boldsymbol{\xi} + \frac{\partial f^{(k)}}{\partial\varepsilon^p}\,\Delta\varepsilon^p + \frac{\partial f^{(k)}}{\partial\phi}\,\Delta\phi = 0
\end{aligned} \tag{5.65}$$

Because the derivatives of $\boldsymbol{r}^{(k)}, \boldsymbol{h}^{\alpha(k)}, \boldsymbol{h}^{\beta(k)}, \boldsymbol{h}^{\phi(k)}$ with respect to $\boldsymbol{\xi}, \varepsilon^p, \phi$ may be difficult to calculate, we instead use a semi-implicit scheme in our implementation where the quantities $\boldsymbol{r}$, h^α, $\boldsymbol{h}^\beta$, and h^ϕ are evaluated at t_n. Then the problematic derivatives disappear and we are left with

$$\begin{aligned}
\mathbf{a}^{(k)} - \Delta\boldsymbol{\xi} - \left[2\,\mu\,\mathrm{dev}(\boldsymbol{r}_n) + \mathrm{dev}(\boldsymbol{h})^\beta_n\right]\delta\gamma &= 0 \\
b^{(k)} - \Delta\varepsilon^p + h^\alpha_n\,\delta\gamma &= 0 \\
c^{(k)} - \Delta\phi + h^\phi_n\,\delta\gamma &= 0 \\
f^{(k)} + \frac{\partial f^{(k)}}{\partial\boldsymbol{\xi}} : \Delta\boldsymbol{\xi} + \frac{\partial f^{(k)}}{\partial\varepsilon^p}\,\Delta\varepsilon^p + \frac{\partial f^{(k)}}{\partial\phi}\,\Delta\phi &= 0
\end{aligned} \tag{5.66}$$

We now force $\mathbf{a}^{(k)}$, $b^{(k)}$, and $c^{(k)}$ to be zero at all times, leading to the expressions

$$\begin{aligned}
\Delta\boldsymbol{\xi} &= -\left[2\,\mu\,\mathrm{dev}(\boldsymbol{r}_n) + \mathrm{dev}(\boldsymbol{h})^\beta_n\right]\delta\gamma \\
\Delta\varepsilon^p &= h^\alpha_n\,\delta\gamma \\
\Delta\phi &= h^\phi_n\,\delta\gamma \\
f^{(k)} + \frac{\partial f^{(k)}}{\partial\boldsymbol{\xi}} : \Delta\boldsymbol{\xi} + \frac{\partial f^{(k)}}{\partial\varepsilon^p}\,\Delta\varepsilon^p + \frac{\partial f^{(k)}}{\partial\phi}\,\Delta\phi &= 0
\end{aligned} \tag{5.67}$$

Plugging the expressions for $\Delta\boldsymbol{\xi}$, $\Delta\varepsilon^p$, $\Delta\phi$ from the first three equations into the fourth gives us

$$f^{(k)} - \frac{\partial f^{(k)}}{\partial\boldsymbol{\xi}} : \left[2\,\mu\,\mathrm{dev}(\boldsymbol{r}_n) + \mathrm{dev}(\boldsymbol{h})^\beta_n\right]\delta\gamma + h^\alpha_n\,\frac{\partial f^{(k)}}{\partial\varepsilon^p}\,\delta\gamma + h^\phi_n\,\frac{\partial f^{(k)}}{\partial\phi}\,\delta\gamma = 0 \tag{5.68}$$

or

$$\Delta\gamma^{(k+1)} - \Delta\gamma^{(k)} = \delta\gamma = \frac{f^{(k)}}{\dfrac{\partial f^{(k)}}{\partial\boldsymbol{\xi}} : \left[2\,\mu\,\mathrm{dev}(\boldsymbol{r}_n) + \mathrm{dev}(\boldsymbol{h}^\beta_n)\right] - h^\alpha_n\,\dfrac{\partial f^{(k)}}{\partial\varepsilon^p} - h^\phi_n\,\dfrac{\partial f^{(k)}}{\partial\phi}}\,. \tag{5.69}$$

5.8.2 Algorithm

The following stress update algorithm is used for each (plastic) time step:

1. Initialize:

$$k = 0\,;\quad (\varepsilon^p)^{(k)} = \varepsilon^p_n\,;\quad \phi^{(k)} = \phi_n\,;\quad \boldsymbol{\beta}^{(k)} = \boldsymbol{\beta}_n\,;\quad \Delta\gamma^{(k)} = 0\,;\quad \boldsymbol{\xi}^{(k)} = \boldsymbol{\xi}^{\mathrm{trial}}\,. \tag{5.70}$$

2. Check yield condition:

$$f^{(k)} := f(\boldsymbol{\xi}^{(k)}, (\varepsilon^p)^{(k)}, \phi^{(k)}, \dot{\varepsilon}_n, T_n, \dots) \tag{5.71}$$

If $f^{(k)} <$ tolerance then go to step 5 else go to step 3.

3. Compute updated $\delta\gamma^{(k)}$ using

$$\delta\gamma^{(k)} = \frac{f^{(k)}}{\dfrac{\partial f^{(k)}}{\partial \boldsymbol{\xi}} : [2\,\mu\,\mathrm{dev}(\boldsymbol{r}_n) + \mathrm{dev}(\boldsymbol{h}_n^{\beta})] - h_n^{\alpha}\,\dfrac{\partial f^{(k)}}{\partial \varepsilon^p} - h_n^{\phi}\,\dfrac{\partial f^{(k)}}{\partial \phi}}\,. \tag{5.72}$$

Compute

$$\begin{aligned} \Delta\boldsymbol{\xi}^{(k)} &= -[2\,\mu\,\mathrm{dev}(\boldsymbol{r}_n) + \mathrm{dev}(\boldsymbol{h}_n^{\beta})]\,\delta\gamma^{(k)} \\ (\Delta\varepsilon^p)^{(k)} &= h_n^{\alpha}\,\delta\gamma^{(k)} \\ \Delta\phi^{(k)} &= h_n^{\phi}\,\delta\gamma^{(k)} \end{aligned} \tag{5.73}$$

4. Update variables:

$$\begin{aligned} (\varepsilon^p)^{(k+1)} &= (\varepsilon^p)^{(k)} + (\Delta\varepsilon^p)^{(k)} \\ \phi^{(k+1)} &= \phi^{(k)} + \Delta\phi^{(k)} \\ \boldsymbol{\xi}^{(k+1)} &= \boldsymbol{\xi}^{(k)} + \Delta\boldsymbol{\xi}^{(k)} \\ \Delta\gamma^{(k+1)} &= \Delta\gamma^{(k)} + \delta\gamma^{(k)} \end{aligned} \tag{5.74}$$

Set $k \leftarrow k+1$ and go to step 2.

5. Update and calculate back stress and the deviatoric part of Cauchy stress:

$$\varepsilon^p_{n+1} = (\varepsilon^p)^{(k)}\;;\quad \phi_{n+1} = \phi^{(k)}\;;\quad \boldsymbol{\xi}_{n+1} = \boldsymbol{\xi}^{(k)}\;;\quad \Delta\gamma_{n+1} = \Delta\gamma^{(k)} \tag{5.75}$$

and

$$\begin{aligned} \hat{\boldsymbol{\beta}}_{n+1} &= \hat{\boldsymbol{\beta}}_n + \Delta\gamma_{n+1}\,\boldsymbol{h}^{\beta}(\boldsymbol{\xi}_{n+1}, \varepsilon^p_{n+1}, \phi_{n+1}) \\ \boldsymbol{\beta}_{n+1} &= \hat{\boldsymbol{\beta}}_{n+1} - \frac{1}{3}\,\mathrm{tr}(\hat{\boldsymbol{\beta}}_{n+1})\,\boldsymbol{1} \\ \boldsymbol{s}_{n+1} &= \boldsymbol{\xi}_{n+1} + \boldsymbol{\beta}_{n+1} \end{aligned} \tag{5.76}$$

6. Update the temperature and the Cauchy stress

$$\begin{aligned} T_{n+1} &= T_n + \frac{\chi_{n+1}\,\Delta t}{\rho_{n+1}\,C_p}\,\sigma_y^{n+1}\,\dot{\varepsilon^p}_{n+1} = T_n + \frac{\chi_{n+1}\,\Delta\gamma_{n+1}}{\rho_{n+1}\,C_p}\,\sigma_y^{n+1}\,h^{\alpha}_{n+1} \\ p_{n+1} &= p(J_{n+1}) \\ \kappa_{n+1} &= J_{n+1}\left[\frac{dp(J)}{dJ}\right]_{n+1} \\ \boldsymbol{\sigma}_{n+1} &= [p_{n+1} - 3\,\kappa_{n+1}\,\alpha\,(T_{n+1} - T_0)]\,\boldsymbol{1} + \boldsymbol{s}_{n+1} \end{aligned} \tag{5.77}$$

5.9 Examples

Let us now look at a few examples.

5.9.1 Example 1

Consider the case of J_2 plasticity with the yield condition

$$f := \sqrt{\frac{3}{2}}\,\|\boldsymbol{s}-\boldsymbol{\beta}\| - \sigma_y(\varepsilon^p, \dot{\varepsilon}, T, \dots) = \sqrt{\frac{3}{2}}\,\|\boldsymbol{\xi}\| - \sigma_y(\varepsilon^p, \dot{\varepsilon}, T, \dots) \le 0 \tag{5.78}$$

where $\|\boldsymbol{\xi}\| = \sqrt{\boldsymbol{\xi} : \boldsymbol{\xi}}$. Assume the associated flow rule

$$\boldsymbol{d}^p = \dot{\gamma}\,\boldsymbol{r} = \dot{\gamma}\,\frac{\partial f}{\partial \boldsymbol{\sigma}} = \dot{\gamma}\,\frac{\partial f}{\partial \boldsymbol{\xi}}\,. \tag{5.79}$$

Then

$$\boldsymbol{r} = \frac{\partial f}{\partial \boldsymbol{\xi}} = \sqrt{\frac{3}{2}}\,\frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} \tag{5.80}$$

and

$$\boldsymbol{d}^p = \sqrt{\frac{3}{2}}\,\dot{\gamma}\,\frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|}\,;\quad \|\boldsymbol{d}^p\| = \sqrt{\frac{3}{2}}\,\dot{\gamma}\,. \tag{5.81}$$

The evolution of the equivalent plastic strain is given by

$$\dot{\varepsilon^p} = \dot{\gamma}\,h^\alpha = \sqrt{\frac{2}{3}}\,\|\boldsymbol{d}^p\| = \dot{\gamma}\,. \tag{5.82}$$

This definition is consistent with the definition of equivalent plastic strain

$$\varepsilon^p = \int_0^t \dot{\varepsilon}^p\,d\tau = \int_0^t \sqrt{\frac{2}{3}}\,\|\boldsymbol{d}^p\|\;d\tau\,. \tag{5.83}$$

The evolution of porosity is given by (there is no evolution of porosity)

$$\dot{\phi} = \dot{\gamma}\,h^\phi = 0 \tag{5.84}$$

The evolution of the back stress is given by the Prager kinematic hardening rule

$$\dot{\hat{\boldsymbol{\beta}}} = \dot{\gamma}\,\boldsymbol{h}^\beta = \frac{2}{3}\,H'\,\boldsymbol{d}^p \tag{5.85}$$

where $\hat{\boldsymbol{\beta}}$ is the back stress and H' is a constant hardening modulus. Also, the trace of $\boldsymbol{d}^p$ is

$$\operatorname{tr}(\boldsymbol{d}^p) = \sqrt{\frac{3}{2}}\,\dot{\gamma}\,\frac{\operatorname{tr}(\boldsymbol{\xi})}{\|\boldsymbol{\xi}\|}\,. \tag{5.86}$$

Since $\boldsymbol{\xi}$ is deviatoric, $\operatorname{tr}(\boldsymbol{\xi}) = 0$ and hence $\boldsymbol{d}^p = \boldsymbol{\eta}^p$. Hence, $\dot{\hat{\boldsymbol{\beta}}} = \dot{\boldsymbol{\beta}}$ (where $\boldsymbol{\beta}$ is the deviatoric part of $\hat{\boldsymbol{\beta}}$), and

$$\dot{\boldsymbol{\beta}} = \sqrt{\frac{2}{3}}\,H'\,\dot{\gamma}\,\frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|}\,. \tag{5.87}$$

These relation imply that

$$\boxed{\begin{aligned} \boldsymbol{r} &= \sqrt{\frac{3}{2}}\,\frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} \\ h^\alpha &= 1 \\ h^\phi &= 0 \\ \boldsymbol{h}^\beta &= \sqrt{\frac{2}{3}}\,H'\,\frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|}\,. \end{aligned}} \tag{5.88}$$

We also need some derivatives of the yield function. These are

$$\begin{aligned}\frac{\partial f}{\partial \boldsymbol{\xi}} &= \boldsymbol{r}\\ \frac{\partial f}{\partial \varepsilon^p} &= -\frac{\partial \sigma_y}{\partial \varepsilon^p}\\ \frac{\partial f}{\partial \phi} &= 0\,.\end{aligned} \tag{5.89}$$

Let us change the kinematic hardening model and use the Armstrong-Frederick model instead, i.e.,

$$\dot{\boldsymbol{\beta}} = \dot\gamma\, \boldsymbol{h}^\beta = \frac{2}{3}\, H_1\, \boldsymbol{d}^p - H_2\, \boldsymbol{\beta}\, \left\|\boldsymbol{d}^p\right\|\,. \tag{5.90}$$

Since

$$\boldsymbol{d}^p = \sqrt{\frac{3}{2}}\, \dot\gamma\, \frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} \tag{5.91}$$

we have

$$\left\|\boldsymbol{d}^p\right\| = \sqrt{\frac{3}{2}}\, \dot\gamma\, \frac{\|\boldsymbol{\xi}\|}{\|\boldsymbol{\xi}\|} = \sqrt{\frac{3}{2}}\, \dot\gamma\,. \tag{5.92}$$

Therefore,

$$\dot{\boldsymbol{\beta}} = \sqrt{\frac{2}{3}}\, H_1\, \dot\gamma\, \frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} - \sqrt{\frac{3}{2}}\, H_2\, \dot\gamma\, \boldsymbol{\beta}\,. \tag{5.93}$$

Hence we have

$$\boxed{\boldsymbol{h}^\beta = \sqrt{\frac{2}{3}}\, H_1\, \frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} - \sqrt{\frac{3}{2}}\, H_2\, \boldsymbol{\beta}\,.} \tag{5.94}$$

5.9.2 Example 2

Let us now consider a Gurson type yield condition with kinematic hardening. In this case the yield condition can be written as

$$f := \frac{3\,\boldsymbol{\xi} : \boldsymbol{\xi}}{2\,\sigma_y^2} + 2\,q_1\,\phi^*\,\cosh\left(\frac{q_2\,\mathrm{tr}(\boldsymbol{\sigma})}{2\,\sigma_y}\right) - \left[1 + q_3\,(\phi^*)^2\right] \tag{5.95}$$

where ϕ is the porosity and

$$\phi^* = \begin{cases} \phi & \text{for } \phi \le \phi_c\\ \phi_c - \dfrac{\phi_u^* - \phi_c}{\phi_f - \phi_c}\,(\phi - \phi_c) & \text{for } \phi > \phi_c \end{cases} \tag{5.96}$$

Final fracture occurs for $\phi = \phi_f$ or when $\phi_u^* = 1/q_1$.

Let us use an associated flow rule

$$\boldsymbol{d}^p = \dot\gamma\, \boldsymbol{r} = \dot\gamma\, \frac{\partial f}{\partial \boldsymbol{\sigma}}\,. \tag{5.97}$$

Then

$$\boldsymbol{r} = \frac{\partial f}{\partial \boldsymbol{\sigma}} = \frac{3\,\boldsymbol{\xi}}{\sigma_y^2} + \frac{q_1\,q_2\,\phi^*}{\sigma_y}\,\sinh\left(\frac{q_2\,\mathrm{tr}(\boldsymbol{\sigma})}{2\,\sigma_y}\right)\boldsymbol{1}\,. \tag{5.98}$$

In this case

$$\operatorname{tr}(\boldsymbol{r}) = \frac{3\, q_1\, q_2\, \phi^*}{\sigma_y}\, \sinh\left(\frac{q_2\, \operatorname{tr}(\boldsymbol{\sigma})}{2\, \sigma_y}\right) \neq 0 \tag{5.99}$$

Therefore,

$$\boldsymbol{d}^p \neq \boldsymbol{\eta}^p\ . \tag{5.100}$$

For the evolution equation for the plastic strain we use

$$(\boldsymbol{\sigma} - \hat{\boldsymbol{\beta}}) : \boldsymbol{d}^p = (1-\phi)\, \sigma_y\, \dot{\varepsilon}^p \tag{5.101}$$

where $\dot{\varepsilon}^p$ is the effective plastic strain rate in the matrix material. Hence,

$$\dot{\varepsilon}^p = \dot{\gamma}\, h^\alpha = \dot{\gamma}\, \frac{(\boldsymbol{\sigma} - \hat{\boldsymbol{\beta}}) : \boldsymbol{r}}{(1-\phi)\, \sigma_y}\,. \tag{5.102}$$

The evolution equation for the porosity is given by

$$\dot{\phi} = (1-\phi)\, \operatorname{tr}(\boldsymbol{d}^p) + A\, \dot{\varepsilon^p} \tag{5.103}$$

where

$$A = \frac{f_n}{s_n\sqrt{2\pi}} \exp[-1/2(\varepsilon^p - \varepsilon_n)^2/s_n^2] \tag{5.104}$$

and f_n is the volume fraction of void nucleating particles, ε_n is the mean of the normal distribution of nucleation strains, and s_n is the standard deviation of the distribution.

Therefore,

$$\dot{\phi} = \dot{\gamma}\, h^\phi = \dot{\gamma}\, \left[(1-\phi)\, \operatorname{tr}(\boldsymbol{r}) + A\, \frac{(\boldsymbol{\sigma} - \hat{\boldsymbol{\beta}}) : \boldsymbol{r}}{(1-\phi)\, \sigma_y}\right]\,. \tag{5.105}$$

If the evolution of the back stress is given by the Prager kinematic hardening rule

$$\dot{\hat{\boldsymbol{\beta}}} = \dot{\gamma}\, \boldsymbol{h}^\beta = \frac{2}{3}\, H'\, \boldsymbol{d}^p \tag{5.106}$$

where $\hat{\boldsymbol{\beta}}$ is the back stress, then

$$\dot{\hat{\boldsymbol{\beta}}} = \frac{2}{3}\, H'\, \dot{\gamma}\, \boldsymbol{r}\,. \tag{5.107}$$

Alternatively, if we use the Armstrong-Frederick model, then

$$\dot{\hat{\boldsymbol{\beta}}} = \dot{\gamma}\, \boldsymbol{h}^\beta = \frac{2}{3}\, H_1\, \boldsymbol{d}^p - H_2\, \hat{\boldsymbol{\beta}}\, \left\|\boldsymbol{d}^p\right\|\ . \tag{5.108}$$

Plugging in the expression for $\boldsymbol{d}^p$, we have

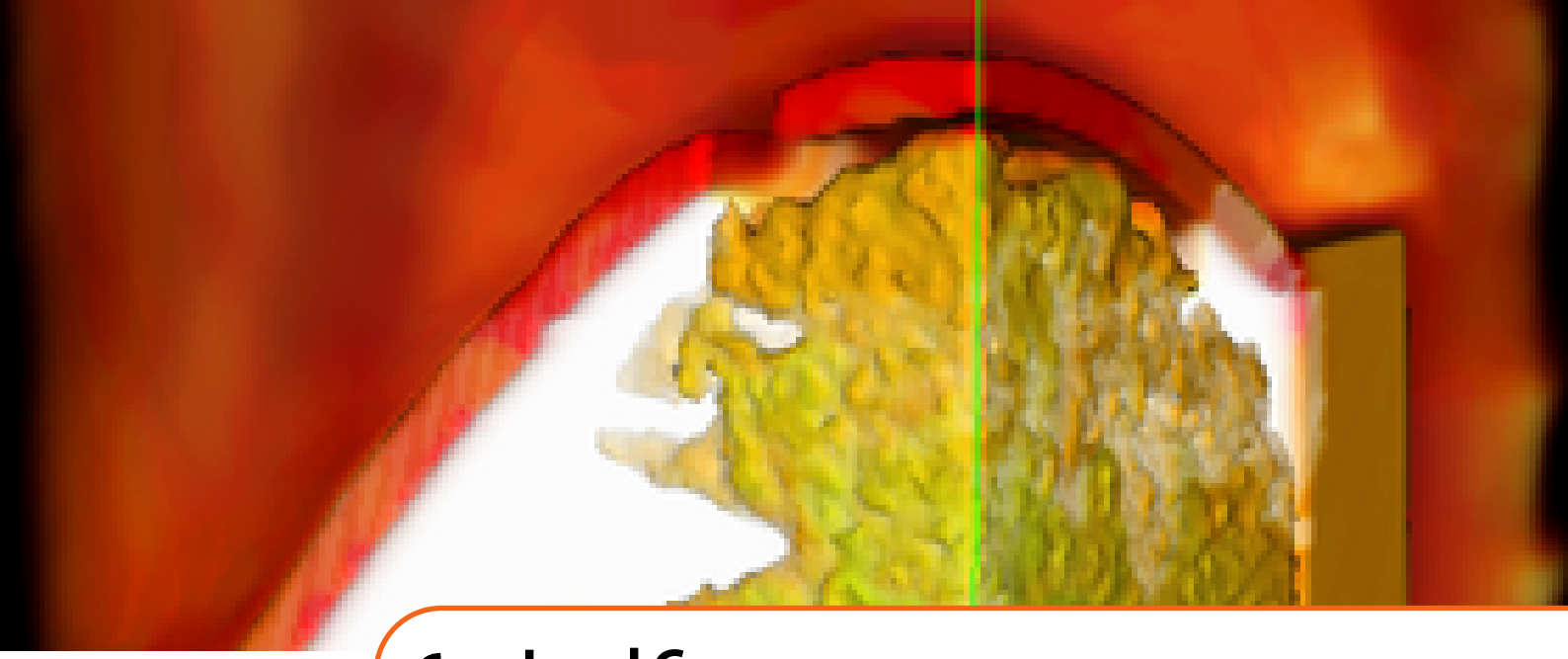
$$\dot{\hat{\boldsymbol{\beta}}} = \dot{\gamma}\, \left[\frac{2}{3}\, H_1\, \boldsymbol{r} - H_2\, \hat{\boldsymbol{\beta}}\, \|\boldsymbol{r}\|\right]\,. \tag{5.109}$$

Therefore, for this model,

$$\boxed{\begin{aligned} \boldsymbol{r} &= \frac{3\,\boldsymbol{\xi}}{\sigma_y^2} + \frac{q_1\,q_2\,\phi^*}{\sigma_y}\,\sinh\left(\frac{q_2\,\mathrm{tr}(\boldsymbol{\sigma})}{2\,\sigma_y}\right)\boldsymbol{1} \\ h^{\alpha} &= \frac{(\boldsymbol{\sigma}-\boldsymbol{\beta}):\boldsymbol{r}}{(1-\phi)\,\sigma_y} \\ h^{\phi} &= (1-\phi)\,\mathrm{tr}(\boldsymbol{r}) + A\,\frac{(\boldsymbol{\sigma}-\hat{\boldsymbol{\beta}}):\boldsymbol{r}}{(1-\phi)\,\sigma_y} \\ \boldsymbol{h}^{\beta} &= \frac{2}{3}\,H_1\,\boldsymbol{r} - H_2\,\hat{\boldsymbol{\beta}}\,\|\boldsymbol{r}\| \end{aligned}} \tag{5.110}$$

The other derivatives of the yield function that we need are

$$\begin{aligned} \frac{\partial f}{\partial \boldsymbol{\xi}} &= \frac{3\,\boldsymbol{\xi}}{\sigma_y^2} \\ \frac{\partial f}{\partial \varepsilon^p} &= \frac{\partial f}{\partial \sigma_y}\,\frac{\partial \sigma_y}{\partial \varepsilon^p} = -\left[\frac{3\,\boldsymbol{\xi}:\boldsymbol{\xi}}{\sigma_y^3} + \frac{q_1\,q_2\,\phi^*\,\mathrm{tr}(\boldsymbol{\sigma})}{\sigma_y^2}\,\sinh\left(\frac{q_2\,\mathrm{tr}(\boldsymbol{\sigma})}{2\,\sigma_y}\right)\right]\frac{\partial \sigma_y}{\partial \varepsilon^p} \\ \frac{\partial f}{\partial \phi} &= 2\,q_1\,\frac{d\phi^*}{d\phi}\,\cosh\left(\frac{q_2\,\mathrm{tr}(\sigma)}{2\,\sigma_y}\right) - 2\,q_3\,\phi^*\,\frac{d\phi^*}{d\phi}\,. \end{aligned} \tag{5.111}$$



6 — Load Curves

It is often more convenient to apply a specified load at the MPM particles. The load may be a function of time. Such a load versus time curve is called a **load curve**.

In Uintah, the load curve infrastructure is available for general use (and not only for particles). However, it has been implemented only for a special case of pressure loading.

We invoke the load curve in the <MPM> section of the input file using <use_load_curves> true </use_load_curves The default value is <use_load_curves> false </use_load_curves>.

In Uintah, a load curve infrastructure is implemented in the file .../MPM/PhysicalBC/LoadCurve.h. This file is essentially a templated structure that has the following private data

```
// Load curve information
std::vector<double> d_time;
std::vector<T> d_load;
int d_id;
```

The variable d_id is the load curve ID, d_time is the time, and d_load is the load. Note that the load can have any form - scalar, vector, matrix, etc.

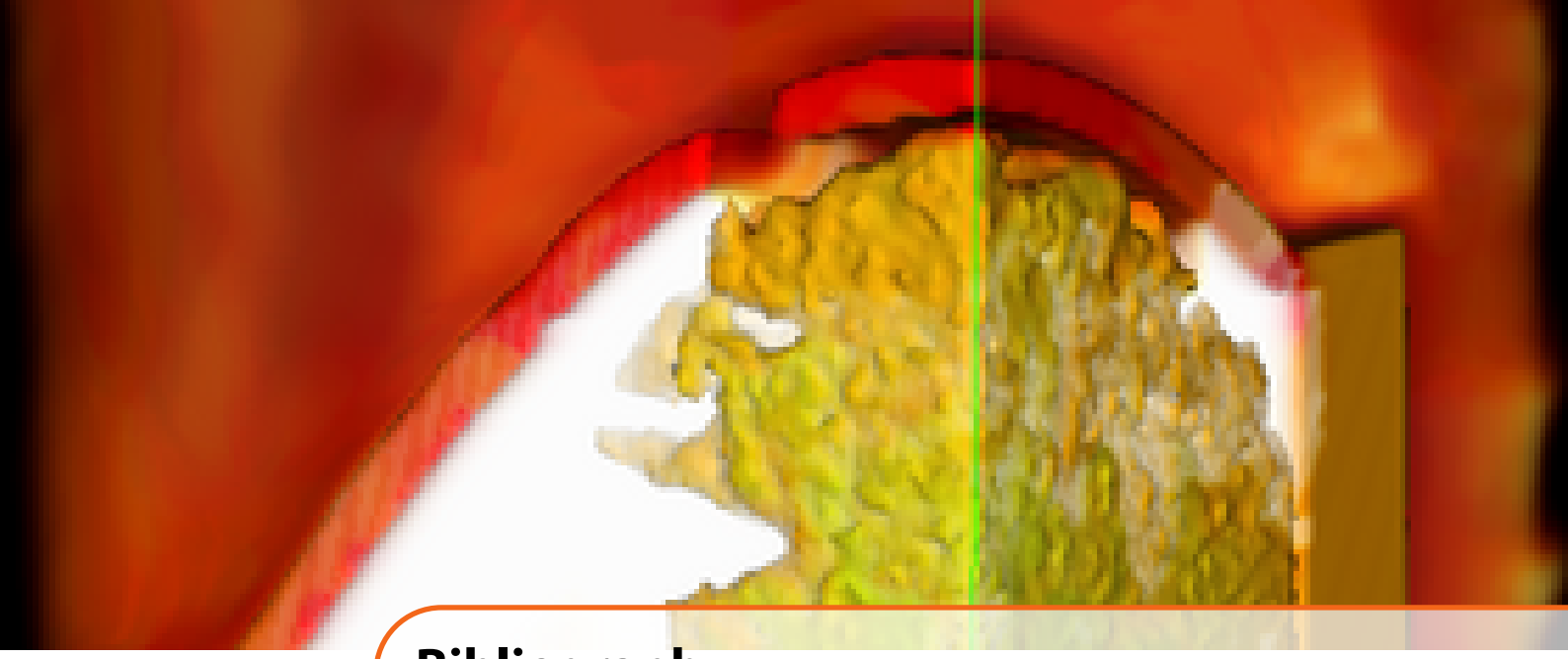
In our current implementation, the actual specification of the load curve information is in the <PhysicalBC> section of the input file. The implementation is limited in that it applies only to pressure boundary conditions for some special geometries (the implementation is in .../MPM/PhysicalBC/PressureBC.cc). However, the load curve template can be used in other, more general, contexts.

A sample input file specification of a pressure load curve is shown below. In this case, a pressure is applied to the inside and outside of a cylinder. The pressure is ramped up from 0 to 1 GPa on the inside and from 0 to 0.1 MPa on the outside over a time of 10 microsecs.

```
<PhysicalBC>
  <MPM>
    <pressure>
      <geom_object>
        <cylinder label = "inner cylinder">
          <bottom>          [0.0,0.0,0.0]    </bottom>
          <top>             [0.0,0.0,.02]    </top>
          <radius>          0.5              </radius>
        </cylinder>
      </geom_object>
      <load_curve>
        <id>1</id>
```

```
          <time_point>
            <time> 0 </time>
            <load> 0 </load>
          </time_point>
          <time_point>
            <time> 1.0e-5 </time>
            <load> 1.0e9 </load>
          </time_point>
        </load_curve>
      </pressure>
      <pressure>
        <geom_object>
          <cylinder label = "outer cylinder">
            <bottom>           [0.0,0.0,0.0]   </bottom>
            <top>              [0.0,0.0,.02]   </top>
            <radius>           1.0             </radius>
          </cylinder>
        </geom_object>
        <load_curve>
          <id>2</id>
          <time_point>
            <time> 0 </time>
            <load> 0 </load>
          </time_point>
          <time_point>
            <time> 1.0e-5 </time>
            <load> 101325.0 </load>
          </time_point>
        </load_curve>
      </pressure>
    </MPM>
  </PhysicalBC>
```

The complete input file can be found in `Uintah/StandAlone/inputs/MPM/thickCylinderMPM.ups`.



Bibliography

[AV05] F. H. Abed and G. Z. Voyiadjis. "A consistent modified Zerilli-Armstrong flow stress model for bcc and fcc metals for elevated temperatures". In: *Acta Mechanica* 175 (2005), pages 1–18 (cited on page 24).

[Ban04a] B. Banerjee. "Material Point Method simulations of fragmenting cylinders". In: *Proc. 17th ASCE Engineering Mechanics Conference (EM2004)*. Newark, Delaware, 2004 (cited on page 28).

[Ban04b] B. Banerjee. *MPM Validation: Sphere-Cylinder Impact: Low Resolution Simulations*. Technical report C-SAFE-CD-IR-04-002. University of Utah, USA: Center for the Simulation of Accidental Fires and Explosions, 2004. URL: http://www.csafe.utah.edu/documents/C-SAFE-CD-IR-04-002.pdf (cited on page 28).

[Ban05] B. Banerjee. "Simulation of impact and fragmentation with the material point method". In: *Proc. 11th International Conference on Fracture*. Turin, Italy, 2005 (cited on page 28).

[BBS00] S.G. Bardenhagen, J.U. Brackbill, and D. Sulsky. "The material-point method for granular materials". In: *Comput. Methods Appl. Mech. Engrg.* 187 (2000), pages 529–541 (cited on page 6).

[BB85] Z. P. Bazant and T. Belytschko. "Wave propagation in a strain-softening bar: Exact solution". In: *ASCE J. Engg. Mech* 111.3 (1985), pages 381–389 (cited on page 32).

[Bec02] R. Becker. "Ring fragmentation predictions using the Gurson model with material stability conditions as failure criteria". In: *Int. J. Solids Struct.* 39 (2002), pages 3555–3580 (cited on page 31).

[BPS00] L. Burakovsky, D. L. Preston, and R. R. Silbar. "Analysis of dislocation mechanism for melting of elements: pressure dependence". In: *J. Appl. Phys.* 88.11 (2000), pages 6294–6301 (cited on page 21).

[CG96] S. R. Chen and G. T. Gray. "Constitutive behavior of tantalum and tantalum-tungsten alloys". In: *Metall. Mater. Trans. A* 27A (1996), pages 2994–3006 (cited on page 22).

[CN80] C. C. Chu and A. Needleman. "Void nucleation effects in biaxially stretched sheets". In: *ASME J. Engg. Mater. Tech.* 102 (1980), pages 249–256 (cited on page 30).

[Cop92] J. O. Coplien. *Advanced C++ Programming Styles and Idioms*. Reading, MA: Addison-Wesley, 1992 (cited on page 28).

[Dru59] D. C. Drucker. "A definition of stable inelastic material". In: *J. Appl. Mech.* 26 (1959), pages 101–106 (cited on page 31).

[FK88] P. S. Follansbee and U. F. Kocks. "A Constitutive Description of the Deformation of copper Based on the Use of the Mechanical Threshold Stress as an Internal State Variable". In: *Acta Metall.* 36 (1988), pages 82–93 (cited on page 25).

[Got+00] D. M. Goto et al. "Anisotropy-corrected MTS constitutive strength modeling in HY-100 steel". In: *Scripta Mater.* 42 (2000), pages 1125–1131 (cited on page 25).

[Gur77] A. L. Gurson. "Continuum theory of ductile rupture by void nucleation and growth: Part 1. Yield criteria and flow rules for porous ductile media". In: *ASME J. Engg. Mater. Tech.* 99 (1977), pages 2–15 (cited on page 28).

[HLQ00] S. Hao, W. K. Liu, and D. Qian. "Localization-induced band and cohesive model". In: *J. Appl. Mech.* 67 (2000), pages 803–812 (cited on page 32).

[HH75] R. Hill and J. W. Hutchinson. "Bifurcation phenomena in the plane tension test". In: *J. Mech. Phys. Solids* 23 (1975), pages 239–264 (cited on page 32).

[HM77] K. G. Hoge and A. K. Mukherjee. "The temperature and strain rate dependence of the flow stress of tantalum". In: *J. Mater. Sci.* 12 (1977), pages 1666–1672 (cited on page 24).

[JC83] G. R. Johnson and W. H. Cook. "A constitutive model and data for metals subjected to large strains, high strain rates and high temperatures". In: *Proc. 7th International Symposium on Ballistics*. 1983, pages 541–547 (cited on page 23).

[JC85] G. R. Johnson and W. H. Cook. "Fracture characteristics of three metals subjected to various strains, strain rates, temperatures and pressures". In: *Int. J. Eng. Fract. Mech.* 21 (1985), pages 31–48 (cited on page 30).

[JA88] J. N. Johnson and F. L. Addessio. "Tensile plasticity and ductile fracture". In: *J. Appl. Phys.* 64.12 (1988), pages 6699–6712 (cited on page 31).

[Koc01] U. F. Kocks. "Realistic constitutive relations for metal plasticity". In: *Materials Science and Engrg.* A317 (2001), pages 181–187 (cited on page 25).

[LSS74] F. L. Lederman, M. B. Salamon, and L. W. Shacklette. "Experimental verification of scaling and test of the universality hypothesis from specific heat data". In: *Phys. Rev. B* 9.7 (1974), pages 2981–2988 (cited on page 27).

[MS96] P. J. Maudlin and S. K. Schiferl. "Computational anisotropic plasticity for high-rate forming applications". In: *Comput. Methods Appl. Mech. Engrg.* 131 (1996), pages 1–30 (cited on page 16).

[NL03] M.-H. Nadal and P. Le Poac. "Continuous model for the shear modulus as a function of pressure and temperature up to the melting point: analysis and ultrasonic validation". In: *J. Appl. Phys.* 93.5 (2003), pages 2472–2480 (cited on page 22).

[Nem91] S. Nemat-Nasser. "Rate-independent finite-deformation elastoplasticity: a new explicit constitutive algorithm". In: *Mech. Mater.* 11 (1991), pages 235–249 (cited on page 16).

[NC92] S. Nemat-Nasser and D. -T. Chung. "An explicit constitutive algorithm for large-strain, large-strain-rate elastic-viscoplasticity". In: *Comput. Meth. Appl. Mech. Engrg* 95.2 (1992), pages 205–219 (cited on pages 16, 18).

[Per98] P. Perzyna. "Constitutive modelling of dissipative solids for localization and fracture". In: *Localization and Fracture Phenomena in Inelastic Solids: CISM Courses and Lectures No. 386*. Edited by Perzyna P. New York: SpringerWien, 1998, pages 99–241 (cited on page 31).

[PTW03] D. L. Preston, D. L. Tonks, and D. C. Wallace. "Model of plastic deformation for extreme loading conditions". In: *J. Appl. Phys.* 93.1 (2003), pages 211–220 (cited on page 26).

[RA98a] S. Ramaswamy and N. Aravas. "Finite element implementation of gradient plasticity models Part I: Gradient-dependent yield functions". In: *Comput. Methods Appl. Mech. Engrg.* 163 (1998), pages 11–32 (cited on page 32).

[RA98b] S. Ramaswamy and N. Aravas. "Finite element implementation of gradient plasticity models Part II: Gradient-dependent evolution equations". In: *Comput. Methods Appl. Mech. Engrg.* 163 (1998), pages 33–53 (cited on page 30).

[Rav+01] G. Ravichandran et al. "On the conversion of plastic work into heat during high-strain-rate deformation". In: *Proc. , 12th APS Topical Conference on Shock Compression of Condensed Matter*. American Physical Society. 2001, pages 557–562 (cited on page 27).

[RR75] J. W. Rudnicki and J. R. Rice. "Conditions for the localization of deformation in pressure-sensitive dilatant materials". In: *J. Mech. Phys. Solids* 23 (1975), pages 371–394 (cited on page 31).

[SH98] J. C. Simo and T. J. R. Hughes. *Computational Inelasticity*. New York: Springer-Verlag, 1998 (cited on page 18).

[SCG80] D. J. Steinberg, S. G. Cochran, and M. W. Guinan. "A constitutive model for metals applicable at high-strain rate". In: *J. Appl. Phys.* 51.3 (1980), pages 1498–1504 (cited on pages 20, 22, 23).

[SL89] D. J. Steinberg and C. M. Lund. "A constitutive model for strain rates from 10^{-4} to 10^{6} s^{-1}". In: *J. Appl. Phys.* 65.4 (1989), pages 1528–1533 (cited on page 23).

[SCS94] D. Sulsky, Z. Chen, and H.L. Schreyer. "A particle method for history dependent materials". In: *Comput. Methods Appl. Mech. Engrg.* 118 (1994), pages 179–196 (cited on page 5).

[SZS95] D. Sulsky, S. Zhou, and H.L. Schreyer. "Application of a particle-in-cell method to solid mechanics". In: *Computer Physics Communications* 87 (1995), pages 236–252 (cited on pages 5, 6).

[TN84] V. Tvergaard and A. Needleman. "Analysis of the cup-cone fracture in a round tensile bar". In: *Acta Metall.* 32.1 (1984), pages 157–169 (cited on page 28).

[TN90] V. Tvergaard and A. Needleman. "Ductile failure modes in dynamically loaded notched bars". In: *Damage Mechanics in Engineering Materials: AMD 109/MD 24*. Edited by J. W. Ju, D. Krajcinovic, and H. L. Schreyer. New York, NY: American Society of Mechanical Engineers, 1990, pages 117–128 (cited on page 32).

[Var70] Y. P. Varshni. "Temperature dependence of the elastic constants". In: *Physical Rev. B* 2.10 (1970), pages 3952–3958 (cited on page 22).

[WA94] L. H. Wang and S. N. Atluri. "An analysis of an explicit algorithm and the radial return algorithm, and a proposed modification, in finite elasticity". In: *Computational Mechanics* 13 (1994), pages 380–389 (cited on page 16).

[Wil99] M. L. Wilkins. *Computer Simulation of Dynamic Phenomena*. Berlin: Springer-Verlag, 1999 (cited on page 20).

[Zer04] F. J. Zerilli. "Dislocation mechanics-based constitutive equations". In: *Metall. Mater. Trans. A* 35A (2004), pages 2547–2555 (cited on page 24).

[ZA87] F. J. Zerilli and R. W. Armstrong. "Dislocation-mechanics-based constitutive relations for material dynamics calculations". In: *J. Appl. Phys.* 61.5 (1987), pages 1816–1825 (cited on page 24).

[ZA93] F. J. Zerilli and R. W. Armstrong. "Constitutive relations for the plastic deformation of metals". In: *High-Pressure Science and Technology - 1993*. American Institute of Physics. Colorado Springs, Colorado, 1993, pages 989–992 (cited on page 24).

[Zoc+00] M. A. Zocher et al. "An evaluation of several hardening models using Taylor cylinder impact data". In: *Proc. , European Congress on Computational Methods in Applied Sciences and Engineering*. ECCOMAS. Barcelona, Spain, 2000 (cited on pages 16, 20, 22).