Material Models and the Uintah Computational Framework

December 28, 2007

Contents

1 Introduction

This document deals with 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. Two stress update algorithms (hypoelastic-plastic and hyperelastic-plastic) are discussed and the manner in

which plasticity flow rules, damage models and equations of state fit into the stress update algorithms are shown.

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.

2.1 Hypoelastic-plastic Stress Update

This section describes the current implementation of the hypoelastic- plastic model.

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.

The evolution variables specific to the hypoelastic-plastic model are the left stretch ($\mathbf{V}$) and the rotation ($\mathbf{R}$). 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). The stress update formula implemented is one based on the approach taken by Zocher et al. (2000) [1].

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.

2.2 Hyperelastic-plastic Stress Update

The stress update used for the hyperelastic-plastic material is a return mapping plasticity algorithm based on a multiplicative decomposition of the deformation gradient and the intermediate configuration concept. The algorithms has been taken from Simo and Hughes (1998) [2].

The variables that are evolved locally in the Hyperelastic-plastic stress update algorithm are the deviatoric part of the left Cauchy-Green tensor and a scalar damage variable.

The plasticity, damage and equation of state models are initialized and called in exactly the same way as in the hypoelastic-plastic model. Therefore, the stress update algorithm and the plasticity, damage and equation of state models are encapsulated from each other and no modifications of the latter are required.

3 Plasticity Model Framework

Currently implemented plasticity models are isotropic hardening, Johnson-Cook and MTS. Addition of a new model requires the following steps :

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.

4 Damage Models

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.

5 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.

5.1 Default hypoelastic equation of state

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

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

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

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

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

$$\dot{p} = 3 \, \kappa \, \text{tr}(\boldsymbol{d}^e) \, . \tag{3}$$

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

$$\dot{p} = 3 \, \kappa \, \text{tr}(\boldsymbol{d}) \, . \tag{4}$$

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

$$\boxed{p_{n+1} = p_n + 3 \, \kappa \, \text{tr}(\boldsymbol{d}_{n+1}) \, \Delta t \, .} \tag{5}$$

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

$$\dot{p} = \frac{\partial p}{\partial J}\,\dot{J} = \frac{\partial p}{\partial J}\,J\,\mathrm{tr}(\boldsymbol{d})\,. \tag{6}$$

Therefore,

$$\boxed{\frac{\partial p}{\partial J} = \frac{3\,\kappa}{J}\,.} \tag{7}$$

5.2 Mie-Gruneisen equation of state

The pressure (p) is calculated using a Mie-Grüneisen equation of state of the form ([3, 1])

$$\boxed{p_{n+1} = -\frac{\rho_0\,C_0^2\,(1-J^e_{n+1})[1-\Gamma_0(1-J^e_{n+1})]}{[1-S_\alpha(1-J^e_{n+1})]^2} - 2\,\Gamma_0\,e_{n+1}\;;\quad J^e := \det\boldsymbol{F}^e} \tag{8}$$

where C_0 is the bulk speed of sound, ρ_0 is the initial mass density, $2\,\Gamma_0$ 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,

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

Also,

$$\boxed{\frac{\partial p}{\partial J^e} = \frac{\rho_0\,C_0^2\,[1+(S_\alpha-2\,\Gamma_0)\,(1-J^e)]}{[1-S_\alpha\,(1-J^e)]^3} - 2\,\Gamma_0\,\frac{\partial e}{\partial J^e}.} \tag{9}$$

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

6 Example Input Files

6.1 Hypoelastic-plastic Stress Update

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 exists).

6.2 Hyperelastic-plastic Stress Update

An example of the portion of an input file that specifies a copper body with a hyperelastic stress update, Mechanical Threshold Stress 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="hyperelastic_plastic">
    <tolerance>5.0e-10</tolerance>
    <include href="inputs/MPM/MaterialData/IsotropicElasticAnnCopper.xml"/>
    <include href="inputs/MPM/MaterialData/MTSPlasticAnnCopper.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 data files are the same as those for the hypoelastic-plastic case, except for the plasticity model. The constants for the MTS plasticity model are specified in the file `MTSPlasticAnnCopper.xml`. The contents of this file are show below.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<Uintah_Include>
  <plasticity_model type="mts_model">
    <s_a>40.0e6</s_a>
    <koverbcubed>0.823e6</koverbcubed>
    <edot0>1.0e7</edot0>
    <g0>1.6</g0>
    <q>1.0</q>
    <p>0.666667</p>
```

```
      <alpha>2</alpha>
      <edot_s0>1.0e7</edot_s0>
      <A>0.2625</A>
      <s_s0>770.0e6</s_s0>
      <a0>2390.0e6</a0>
      <a1>12.0e6</a1>
      <a2>1.696e6</a2>
      <b1>45.78e9</b1>
      <b2>3.0e9</b2>
      <b3>180</b3>
      <mu_0>47.7e9</mu_0>
    </plasticity_model>
  </Uintah_Include>
```

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.

References

[1] M. A. Zocher, P. J. Maudlin, S. R. Chen, and E. C. Flower-Maudlin. An evaluation of several hardening models using Taylor cylinder impact data. In *Proc. , European Congress on Computational Methods in Applied Sciences and Engineering*, Barcelona, Spain, 2000. ECCOMAS.

[2] JC Simo and TJR Hughes. *Computational Inelasticity*. Springer-Verlag, New York, 1998.

[3] M. L. Wilkins. *Computer Simulation of Dynamic Phenomena*. Springer-Verlag, Berlin, 1999.