Validation of VUEOS in Abaqus/Explicit and Application to Noble-Abel and Mie-Grüneisen Equations of State

A. Colletti, P. Carlucci

U.S. Army – ARDEC, Picatinny, NJ 07806

N. Dan, J. Hurtado

Dassault Systemes Simulia Corp, Providence, RI 02909

Abstract: In the release of Abaqus 6.13-1, SIMULIA is adding a new user subroutine capability in Explicit termed VUEOS. This capability allows the user to define a hydrodynamic material model in which the material's volumetric response is determined by a user-defined equation of state (EOS). The introduction of this capability gives the user the ability define an EOS based on a constitutive relationship involving pressure, energy, and density, where the pressure may have parameter fits or empirical data. In order to validate this new user subroutine framework, the U.S. Army – ARDEC developed three user EOS's for comparison with analytical and published data. The validation is completed in two parts: First, a comparison to the analytical results of the constitutive equations; and second, application to other equations of state. The Noble-Abel EOS, a modified Van der Waals EOS, is implemented. This EOS is used for high pressure gases such as those found in projectile launch generated from burning propellant. A Mie-Grüneisen equation of state is also defined and determines the pressure in shock compressed solid materials. This implementation uses a third order fit to the shock vs. particle velocity and a volume correction term. Validation results are presented, indicating a successful implementation of the new VUEOS subroutine interface.

Keywords: VUEOS, Ideal Gas, Noble-Abel, Mie-Grüneisen, Equation of State, Us-Up, Constitutive Model, Shock Compression, user subroutine, CEL, Abaqus/Explicit.

1. Introduction

In developing weapon systems for the warfighter, the US Army uses modeling and simulation tools to support the design, test and manufacturing of these systems. One of these tools is Abaqus/Explicit, including the Coupled Eulerian-Lagrangian capability (CEL). The problems that are generally solved with Abaqus/Explicit CEL are typically high rate dynamic events with large material deformations. Under these types of loading, the behavior of many materials can be captured using a hydrodynamic Equation of State (EOS) which defines the volumetric response of the material. Abaqus/Explicit includes several different equations of state as part of the built-in material models, including linear Mie-Grüneisen u_s-u_p, JWL, and ideal gas. But in many cases, it

is desired to use an equation of state which is not included. While it is possible to develop a full VUMAT (user defined material model), this approach is burdensome if only the volumetric response is of interest.

The introduction of the user defined equation of state (VUEOS) allows the user to forego the burden of using the VUMAT interface, and gives a simple and straightforward interface. The VUEOS framework is used to define a hydrodynamic material where the volumetric response of the material is directly specified. The validation is in two parts: First, a comparison to the analytical results of the constitutive equations; and second, application of the model to a class of problems of interest to the U.S. Army.

2. VUEOS Validation Cases

2.1 Ideal Gas Equation of State

The first material model considered is the ideal gas equation of state. Since this material model is already implemented in Abaqus, it is considered a good test candidate as it can be compared to the built-in implementation. The subroutine requires the definition of three terms: P, $\partial P/\partial \rho$, and $\partial P/\partial E_m$. For an ideal gas, in terms of energy (with $E_m = C_v T$), they are the following relations:

$$P = \frac{\rho R E_m}{C_v} \tag{1}$$

$$\frac{\partial P}{\partial \rho} = \frac{R\, E_m}{C_v} \tag{2}$$

$$\frac{\partial P}{\partial E_m} = \frac{\rho R}{C_v} \tag{3}$$

Where ρ is density, R is the gas constant for a specific substance, E_m is the energy, C_v is the specific heat at constant volume, and T is temperature. It is preferable to have the equations in terms of energy in the case of a variable specific heat. The resulting temperature is compared against the analytical solution in Figure 1 with excellent correlation, with the maximum error being less than 0.2%. Figure 2 shows a comparison for a piston cylinder arrangement which is discussed further in the following section (2.2).

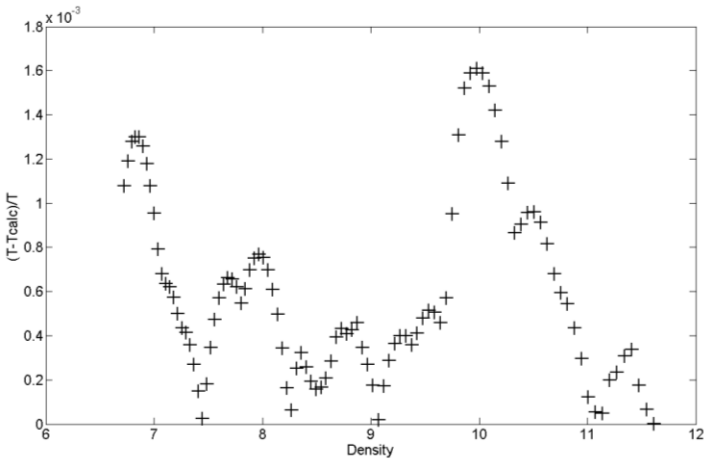

Figure 1. Relative Temperature Error Vs. Density.

2.2 Noble-Abel Equation of State

In high-pressure/high-temperature regimes, gases are not accurately modeled using the ideal gas assumptions. The Noble-Abel equation of state is often used; it is a modified Van der Waals equation of state that adds a co-volume term, b, to the constitutive relationship. The co-volume has no strict physical meaning but takes into account inter-molecular forces and molecule sizes (Carlucci, 2008). It can either be measured indirectly through closed bomb experimentation or estimated with relations (Gheorghian, 2007). The equation takes the following form:

$$P\left(1/\rho - b\right) = RT \tag{4}$$

Where in terms of energy:

$$P = \frac{\rho R E_m}{C_v(1-\rho b)} \tag{5}$$

$$\frac{\partial P}{\partial \rho} = \frac{R E_m}{C_v(1-\rho b)^2} \tag{6}$$

$$\frac{\partial P}{\partial E_m} = \frac{\rho R}{C_v(1-\rho b)} \tag{7}$$

Similar to Figure 1 for an ideal gas EOS, the Noble-Abel VUEOS temperature is compared against the analytical solution with excellent correlation with errors less than 0.2%. The data is for a hydrostatic compression of a single element. These results are not surprising at it indicates that the VUEOS framework has been implemented correctly into Abaqus/Explicit.

The Noble-Abel equation of state will be typically used in problems with burning propellant creating high pressures gases. This is the case for gases pushing on a projectile during the gun launch event. The validation problem chosen for the ideal gas and Noble-Abel equations of state is a piston-cylinder arrangement where the piston is driven by an initial pocket of high pressure gas. The simulation uses an Eulerian gas and Lagrangian piston. The resulting position-time curves are shown in Figure 2 and are compared to a lumped-parameter, analytical solution. The small differences are attributed to two factors: First, the lumped parameter solution ignores pressure transients while they are included in the finite element model and Second, a slight variation in initial conditions. Since temperature is a nodal quantity, it is applied to a slighter larger domain than where initial pressure is prescribed to. This results in slightly more energy in the finite element model than the analytical model.

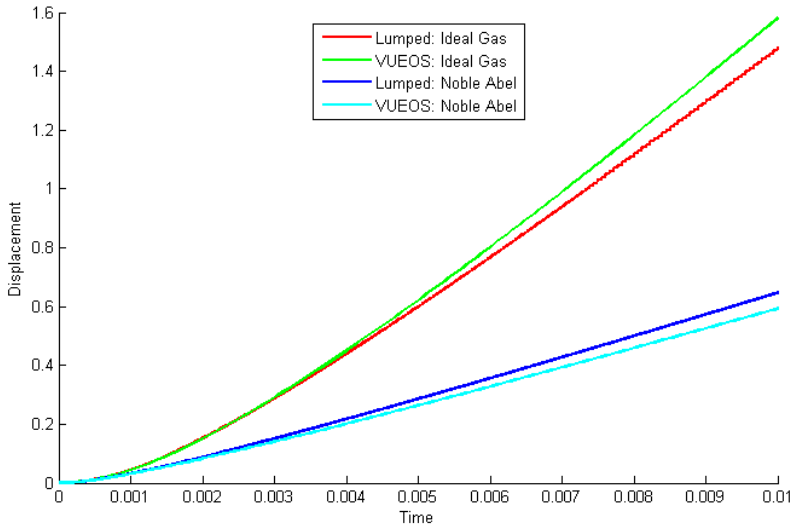

Figure 2. Displacement vs. Time for a piston cylinder with various gas models.

2.3 Mie-Grüneisen Equation of State

Mie-Grüneisen equation of state is used to determine the pressure in shock compressed solid materials. It is derived to keep the internal energy on the Hugoniot and uses a curve fit for material data of shock velocity vs. particle velocity (u_s vs. u_p). Often this is a linear fit of the form:

$$u_s = c_0 + s u_p \tag{8}$$

Where c_0 is the intercept (often considered the bulk speed of sound in the material at ambient conditions) and s is fit to the data. This is the form for the Abaqus/Explicit built-in "u_s - u_p" equation of state. For some materials a higher order fit is more appropriate, and is of the form:

$$u_s = c_0 + S_1 u_p + S_2 \left(\frac{u_p}{u_s}\right) u_p + S_3 \left(\frac{u_s}{u_p}\right)^2 u_p \tag{9}$$

Coefficients S_1, S_2, and S_3 are fit to the experimentally measured curve. Pressure is in the form:

$$P = \frac{\rho_0 c^2 \mu \left(1 + \left(1 - \frac{\gamma_0}{2}\right)\mu + \frac{a}{2}\mu^2\right)}{\left(1 - (S_1 - 1)\mu - \frac{S_2 \mu^2}{\mu + 1} - \frac{S_3 \mu^3}{(\mu+1)^2}\right)^2} + (\gamma_0 + a\mu)E \tag{10}$$

Where $\mu = \frac{\rho}{\rho_0} - 1$, ρ is the density, γ_0 is the Grüneisen parameter, and the subscript 0 refers to the reference state at ambient conditions. This also incorporates a, a volume correction term. In order to use the VUEOS subroutine, the partial derivatives with respect to density and energy are needed:

$$\frac{\partial P}{\partial \rho} = \frac{Ea}{\rho_0} - \frac{c^2\left(\frac{a\mu^2}{2} + \mu\left(\frac{\gamma}{2} - 1\right) - 1\right)}{\left((S_1 - 1)\mu + \frac{S_2 \rho_0 \mu^2}{\rho} + \frac{S_3 \rho_0^2 \mu^3}{\rho^2} - 1\right)^2} - \frac{c^2 \rho \mu \left(\frac{\left(\frac{\gamma}{2} - 1\right)}{\rho_0} + \frac{a\mu}{\rho_0}\right)}{\left((S_1 - 1)\mu + \frac{S_2 \rho_0 \mu^2}{\rho} + \frac{S_3 \rho_0^2 \mu^3}{\rho^2} - 1\right)^2}$$
$$+ \frac{2c^2 \rho \mu \left(\frac{a\mu^2}{2} + \mu\left(\frac{\gamma_0}{2} - 1\right) - 1\right)\left(\frac{S_1 - 1}{\rho_0} + \frac{2S_2 \mu}{\rho} - \frac{3S_3 \rho_0 \mu^2}{\rho^2} - \frac{2s_3 \rho_0^2 \mu^3}{\rho^3}\right)}{\left((S_1 - 1)\mu + \frac{S_2 \rho_0 \mu^2}{\rho} + \frac{S_3 \rho_0^2 \mu^3}{\rho^2} - 1\right)^3} \tag{11}$$

$$\frac{\partial P}{\partial E} = \gamma_0 + a\mu \tag{12}$$

To expand the material definition to a tensile state pressure is often used in the form (LSTC 1999):

$$P = \rho_0 c^2 \mu + (\gamma_0 + a\mu)E \tag{13}$$

With the derivatives being:

$$\frac{\partial P}{\partial \rho} = \frac{2\rho}{\rho_0}\left(c^2 + \frac{aE}{\rho_0}\right) \tag{14}$$

$$\frac{\partial P}{\partial E} = \gamma_0 + a\mu \tag{15}$$

When the S_2, S_3, and a terms are set to zero in this form of the Mie-Grüneisen equation of state it reduces to the linear form shown in Equation 8. The potential advantages of this higher order fit are shown for Teflon whose u_s vs. u_p data is shown in Figure 3, parameters in Table 1, and results in Figure 4 (Marsh, 1980). The linear u_s vs. u_p fit results in an R^2 value of 0.682 and the polynomial 0.697. While this is only a slight improvement in this case, in many cases a linear fit is will not work for the measured data and the higher order fit is required for good simulation results.

Also, the higher order implementation includes the volume correction term which is missing from the linear u_s vs. u_p form.

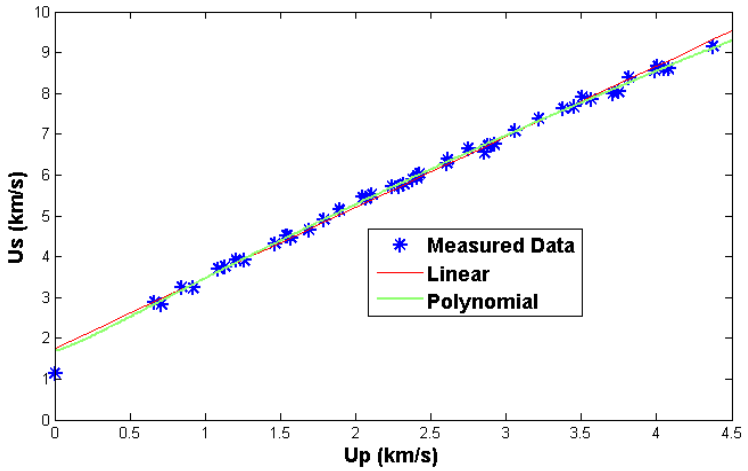

Figure 3. Teflon u_s vs. u_p experimental data and corresponding fits.

Table 1. Teflon Mie-Grüneisen model parameters.

	ρ_0 (g/cm^3)	c_0 (cm/μs)	S_1	S_2	S_3	γ_0	a
Linear fit	2.15	0.175	1.73	--	--	0.59	--
3rd order fit	2.15	0.168	1.123	3.983	-5.797	0.59	0.0

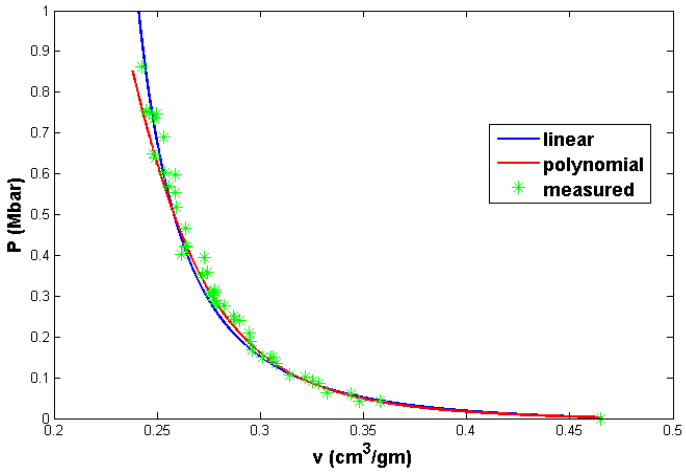

Figure 4. Teflon Pressure vs. Specific Volume, experimental and model results.

3. Conclusion

The inclusion of VUEOS in the latest release of Abaqus successfully alleviates the tediousness of writing a full user-defined material model (VUMAT). It is confirmed reproduce the results of the ideal gas law and Noble-Abel equation of state. The VUEOS construct has expanded to Mie-Grüneisen equation of state for third order curve fits to the u_s vs. u_p data with a volume correction term. While these validation cases are not exhaustive, they are more than satisfactory to give confidence in using the VUEOS interface to develop various equations of state. It allows easier access to a broad spectrum of material models that are well suited for use Abaqus/Explicit and CEL.

4. Appendix

4.1 Ideal Gas Equation of State – VUEOS Source Code

```
      subroutine vueos(
C Read only -
     *     nblock,
     *     jElem, kIntPt, kLayer, kSecPt,
     *     stepTime, totalTime, dt, cmname,
     *     nstatev, nfieldv,
     *     nprops, props,
     *     tempOld, tempNew, fieldOld, fieldNew,
     *     stateOld, charLength,
     *     coordMp,
     *     densityMean,
     *     refDensity,
     *     density, dkk,
     *     Emt,
C Write only -
     *     press, dpdrho,dpdem,
     *     stateNew )

      include 'vaba_param.inc'

      dimension props(nprops),
     *  tempOld(nblock),
     *  fieldOld(nblock,nfieldv),
     *  stateOld(nblock,nstatev),
     *  tempNew(nblock),
     *  fieldNew(nblock,nfieldv),
     *  press(nblock), density(nblock),
     *  refDensity(nblock), Emt(nblock),
     *  energy(nblock), energyBulk(nblock),
     *  coordMp(nblock,*),dkk(nblock),
     *  dpdrho(nblock), densityMean(nblock),
     *  dpdem(nblock), charLength(nblock),
     *  qOld(nblock), qNew(nblock),
     *  stateNew(nblock,nstatev)

      character*80 cmname

      double precision :: xR_Specific, xPress_Amb, xCv
```

```
xR_Specific = props(1)
xPress_Amb  = props(2)
xCv         = props(3)

do k=1, nblock
  press(k) = density(k)*xR_Specific*emt(k)/xCv -xPress_Amb
  dPdRho(k)= xR_Specific*emt(k)/xCv
  dPdEm(k) = density(k)*xR_Specific/xCv
end do
return
end
```

4.2 Noble-Abel Equation of State – VUEOS Source Code (equations only)

```
double precision :: xR_Specific, xCv,xB, xPress_Amb
double precision :: one, two
parameter (one = 1.d0, two = 2.d0 )

xR_Specific = props(1)
xPress_Amb  = props(2)
xCv         = props(3)
xB          = props(4)

do k=1, nblock

  press(k) = density(k)*xR_specific*Emt(k)/
*            (xCv*(one-density(k)*xB)) - xPress_Amb
  dPdRho(k)= xR_specific*Emt(k)/
*            (xCv*(density(k)*xB-one))**two
  dPdEm(k) = density(k)*xR_Specific
*            / (xCv*(one-density(k)*xB))
end do
```

4.3 Mie-Grüneisen Equation of State – VUEOS Source Code (equations only)

```
double precision :: xRho_0, xC, xS1, xS2, xS3, xG_0, xA
!write only variables
double precision :: xMu, xTop, xBottom, xTerm2, xTerm3, xTerm4,
*                   xRho, xE, xGo2m1
!constants
double precision,parameter :: one = 1.d0, two = 2.d0, three = 3.d0

xRho_0 = props(1)
xC     = props(2)
xS1    = props(3)
xS2    = props(4)
xS3    = props(5)
xG_0   = props(6)
xA     = props(7)

do k = 1 , nblock
  xRho = density(k)
```

```
      xE   = emt(k)*xRho_0
      xMu  = xRho/xRho_0-one

     if (xMu .GE. 0.d0) then !compressive state
      !commonly appearing variable grouping
      xGo2m1 = xG_0/two-one

      !Pressure term
      xTop = one+(one-xG_0/two)*xMu-xA/two*xMu**two
      xBottom = (one-(xS1-one)*xMu-xS2*xMu**two/(xMu+one)
     *     -xS3*xMu**three/(xMu+one)**two)**two
      press(k) = xRho_0*xC**two*xMu*xTop/xBottom
     *               + (xG_0+xA*xMu)*xE

      !Deriv of pressure wrt density
      xTop = xC**two* (xA*xMu**two/two + xMu*xGo2m1 - one)
      xBottom = ( (xS1-one)*xMu
     *           + xS2*xRho_0*xMu**two/xRho
     *           + xS3*xRho_0*two*xMu**three/xRho**two
     *           - one)**two
      xTerm2 = xTop/xBottom
      xTop = xC**two*xRho_0*xMu*
     *         (xGo2m1/xRho_0 + xA*xMu/xRho_0)
      xBottom = ( (xS1-one)*xMu
     *           + xS2*xRho_0*xMu**two/xRho
     *           + xS3*xRho_0**two*xMu**three/xRho**two
     *           - one)**two
      xTerm3 = xTop/xBottom
      xTop = two*xC**two*xRho_0*xMu
     *        * (xA*xMu**two/two + xMu*xGo2m1 - one)
     *        * ( (xS1-one)/xRho_0
     *           + two*xS2*xMu/xRho
     *           - xS2*xRho_0*xMu**two/xRho**two
     *           + three*xS3*xRho_0*xMu**two/xRho**two
     *           - two*xS3*xRho_0**two*xMu**three/xRho**three)
      xBottom = ( (xS1-one)*xMu
     *             + xS2*xRho_0*xMu**two/xRho
     *             + xS3*xRho_0**two*xMu**three/xRho**two
     *             - one )**three
      xTerm4 = xTop/xBottom
      dPdRho(k) = xE*xA/xRho_0 - xTerm2 - xTerm3 + xTerm4

      !Deriv of pressure wrt energy
      dPdEm(k) = (xG_0+xA*xMu)*xRho_0

      else !tensile state
      press(k) = xRho_0*xC**two*xMu + (xG_0+xA*xMu)*xE
      dPdRho(k) = two*xRho/xRho_0*(xC**two+xA*xE/xRho_0)
      dPdEm(k) = (xG_0+xA*xMu)*xRho_0
      end if
    end do
```

5. References

1. Carlucci, D., and Jacobson, S. "Ballistics: Theory and Design of Guns and Ammunition," CRC Press, 2008.

2. Gheorghian, S., and T. Chereches, "Estimation of the Force and Gas Covolume of a Propellant from Closed Bomb Experiments," Military Technical Academy, Romania, 2007.

3. Marsh, S. P, "LASL Shock Hugoniot Data," Vol. 5. University of California Press, 1980.

4. LSTC, "LS-DYNA Keyword User's Manual," Volume 1. Livermore Software Technology Corporation (LSTC), May 1999.

5. Abaqus Users Manual, Version 6.12-1, Dassault Systémes Simulia Corp., Providence, RI.