An efficient and robust VUMAT implementation of elastoplastic constitutive laws in Abaqus/Explicit finite element code

Lu Ming, Olivier Pantalé

An efficient and robust VUMAT implementation of elastoplastic constitutive laws in Abaqus/Explicit finite element code

Lu Ming, Olivier Pantalé*

Laboratoire Génie de Production, Université de Toulouse, INP-ENIT, Toulouse, France

Abstract

This paper describes the development of an efficient and robust numerical algorithm for the implementation of elastoplastic constitutive laws in the commercial nonlinear finite element software Abaqus/Explicit through a VUMAT FORTRAN subroutine. In the present paper, while the Abaqus/Explicit uses an explicit time integration scheme, the implicit radial return mapping algorithm is used to compute the plastic strain, the plastic strain rate and the temperature at the end of each increment instead of the widely used forward Euler approach. This more complex process allows us to obtain more precise results with only a slight increase of the total computational time. Corrector term of the radial return scheme is obtained through the implementation of a safe and robust Newton-Raphson algorithm able to converge even when the piecewise defined hardening curve is not derivable everywhere. The complete method of how to implement a user-defined elastoplastic material model using the radial return mapping integration scheme is presented in details with the application to the widely used Johnson-Cook constitutive law. Five benchmark tests including one element tests, necking of a circular bar and 2D and 3D Taylor impact tests show the efficiency and robustness of the proposed algorithm and confirm the improved efficiency in terms of precision, stability and solution CPU time. Finally, three alternative constitutive laws (the TANH, modified TANH and Bäker laws) are presented, implemented through our VUMAT routine and tested.

Keywords: Radial return algorithms, Elastoplasticity, Finite element, Strain-dependent, Thermomechanical

1 Introduction

Over the last past years, numerous studies have been concentrated on dynamic mechanical properties of materials. Constitutive laws, the relation between internal forces and deformations, have also received much attention. Under large deformations and high deformation rates such as forming, machining processes and impact analyses, elastoplastic constitutive laws play a significant role in predicting the mechanical behavior of materials.

Although many kinds of constitutive laws have been implemented into the commercial nonlinear finite element software Abaqus [1], in some conditions they cannot fit well some complex

*Corresponding author
Email address: Olivier.Pantale@enit.fr (Olivier Pantalé)
URL: http://www.enit.fr (Olivier Pantalé)

behaviors of materials such as the deformation process involving large strain, high strain rates, temperature softening, etc... Therefore, Abaqus provides the ability for users to implement only the constitutive flow, or the complete constitutive behavior law by themselves via FORTRAN user subroutines: UHARD or VUHARD for the hardening flow law and UMAT or VUMAT for the complete behavior law. While the user hardening implementation is quite simple and easy, the user material implementation needs more expertise. The major difference from VUMAT to UMAT is that VUMAT subroutines do not need the evaluation of the so-called consistent Jacobian matrix, but only the evaluation of the stress tensor $\boldsymbol{\sigma}_1$ at the end of the increment. The usual approach used to write a VUMAT subroutine, because the time increment is very small due to the explicit time integration scheme, is to use a straightforward evaluation of the equivalent plastic strain increment $\Delta\bar{\varepsilon}^p$ without requiring any local iterative algorithm [2, 3].

In this paper a different approach has been chosen and consist in evaluating, in the VUMAT subroutine, the plastic strain increment $\Delta\bar{\varepsilon}^p$ using the implicit radial return algorithm presented in section 2. In order to illustrate the proposed approach and to be able to validate it, the Johnson-Cook constitutive flow law [4], presented in section 3, has been selected to be implemented into the Abaqus/Explicit code through both a VUHARD and a VUMAT subroutines. The complete implementation of the Johnson-Cook elastoplastic constitutive law through a VUMAT subroutine is presented in details in section 4. Numerical efficiency and precision of the proposed implementation are finally investigated, in section 5, through some dynamic benchmarks tests and have shown the efficiency and robustness of the proposed integration scheme. Section 5 also present some alternative constitutive laws results for the Taylor impact test.

2 Time integration algorithm of J_2 plasticity

Time integration algorithms based on finite difference methods are widely used in finite element solutions of mechanics. In these algorithms, time is discretized on a finite grid, and the distance between consecutive points on the grid is defined as the time step Δt. With the positions and some of the time derivatives at time t, the integration scheme calculates the same quantities at a later time $t+\Delta t$. Through the iteration of the procedure, the time evolution of those quantities can be traced.

2.1 J_2 plasticity model

The elastic stress-strain equation is usually written in the following incremental form:

$$\overset{\nabla}{\boldsymbol{\sigma}} = \mathbb{H} : \mathsf{D}^e \tag{1}$$

where $\overset{\nabla}{\boldsymbol{\sigma}}$ is an objective stress rate in order to take into account the objectivity in large deformations, : is the double dot product, $\mathbb{H}$ is the linear isotropic fourth order elastic tensor given by:

$$\mathbb{H} = K\mathsf{1}\otimes\mathsf{1} - 2G(\mathbb{I} - \frac{1}{3}\mathsf{1}\otimes\mathsf{1}) \tag{2}$$

with $\mathsf{1}$ the second order identity tensor, $\mathbb{I}$ the fourth order identity tensor, $\otimes$ the Dyadic product, G the shear modulus and K the Bulk modulus linked to the Young's modulus E and Poisson's ratio ν by the following:

$$G = \frac{E}{2(1+\nu)} \quad \text{and} \quad K = \frac{E}{3(1-2\nu)} \tag{3}$$

and D^e is the so-called elastic part of the rate of deformation tensor D generally assumed [5] for hypo-elastic material to conform to the following additive decomposition equation:

$$\mathsf{D} = \mathsf{D}^e + \mathsf{D}^p \tag{4}$$

where D^p is the plastic part of the rate of deformation tensor D. According to the literature, J_2 plasticity, also called von Mises yield criterion, was proposed by Huber in 1904 and von Mises in 1913 [6]. It is widely applied in mechanical engineering and metal forming. It assumes the existence of a scalar yield function f given by:

$$f = \overline{\sigma} - \sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T) = 0 \tag{5}$$

where $\overline{\sigma}$ is the von Mises equivalent stress, or effective stress, defined from the deviatoric stress $\mathsf{S} = \boldsymbol{\sigma} - \frac{1}{3}\,\mathrm{tr}\,[\boldsymbol{\sigma}]\mathsf{1}$ as:

$$\overline{\sigma} = \sqrt{\frac{3}{2}\mathsf{S} : \mathsf{S}} \tag{6}$$

and σ^y is the so-called current yield stress of the material, depending in our kind of applications on the the equivalent plastic strain $\overline{\varepsilon}^p$, the equivalent plastic strain rate $\dot{\overline{\varepsilon}}^p$ and the temperature T given by:

$$\overline{\varepsilon}^p = \int_0^t \dot{\overline{\varepsilon}}^p \, dt = \int_0^t \sqrt{\frac{2}{3}\mathsf{D}^p : \mathsf{D}^p}\, dt \tag{7}$$

$$\dot{T} = \frac{\eta}{\rho C_p}\boldsymbol{\sigma} : \mathsf{D}^p \tag{8}$$

where η is the Taylor-Quinney [7] coefficient defining the amount of plastic work converted into heat energy, C_p is the specific heat coefficient and ρ is the density of the material. The so-called yield function f defines the yield surface (when $f = 0$), the elastic domain (when $f < 0$) and the no-access domain (when $f > 0$). Assuming an associative plastic flow rule, the plastic strain rate D^p can be expressed with the following equation:

$$\mathsf{D}^p = \gamma \mathsf{n} \tag{9}$$

where γ is a scalar representing the flow intensity and n is a second order tensor (the unit normal to the flow stress determined exclusively in terms of the trial elastic stress [8]) representing the flow direction given by:

$$\mathsf{n} = \frac{\mathsf{S}}{\sqrt{\mathsf{S} : \mathsf{S}}} \tag{10}$$

From equation (7) one can easily obtain:

$$\dot{\overline{\varepsilon}}^p = \sqrt{\frac{2}{3}}\gamma \quad \text{and} \quad \overline{\varepsilon}^p = \sqrt{\frac{2}{3}}\int_0^t \gamma dt = \sqrt{\frac{2}{3}}\Gamma \tag{11}$$

The plasticity model described here above has to be integrated in time with respect to an incremental objective algorithm. Different expressions of the time derivative of the Cauchy stress tensor $\overset{\nabla}{\boldsymbol{\sigma}}$ used in equation (1) can be used, leading to some noticeable differences in the final results. Abaqus/Explicit for example in case of solid elements uses a Jaumann stress rate for all Built-In constitutive models and a Green-Naghdi stress rate in case of VUMAT user subroutines [1]. This can lead to some difficulties when one wants to compare VUMAT results with Built-In models as we will see further.

2.2 Radial return mapping algorithm

A widely used method for the time integration of J_2 plasticity with isotropic hardening is the return mapping algorithm introduced by Wilkins [9] and Maenchen and Sack [10]. This algorithm is now extensively used and detailed in numerous books such as [6, 11] or papers like Simo et *al.* [8] or Ponthot [12]. We assume that, thanks to the Finite Element formulation, the strain increment between time t_0 (the last equilibrated configuration) and t_1 (the new equilibrated configuration with $t_1 = t_0 + \Delta t$) is $\Delta\mathsf{E}$ and the deviatoric stress tensor at time t_0 is S_0. The trial deviatoric part of the stress tensor S_{tr} and the final hydrostatic pressure p_1 are calculated from the strain increment $\Delta\mathsf{E}$ assuming that the whole step is fully elastic in a first time, so that:

$$\mathsf{S}_{tr} = \mathsf{S}_0 + 2G\,\mathsf{dev}\left[\Delta\mathsf{E}\right] \tag{12}$$

and:

$$p_1 = p_0 + K\,\mathsf{tr}\left[\Delta\mathsf{E}\right] \tag{13}$$

Conforming to equation (5), at time t_1, the yield function f is therefore:

$$f = \overline{\sigma}_{tr} - \sigma_0^y(\overline{\varepsilon}_0^p, \dot{\overline{\varepsilon}}_0^p, T_0) \tag{14}$$

with:

$$\overline{\sigma}_{tr} = \sqrt{\frac{3}{2}\mathsf{S}_{tr} : \mathsf{S}_{tr}} \tag{15}$$

and σ_0^y the yield stress of the last equilibrated configuration at the beginning of the current increment ($t = t_0$). In order to know if the predicted trial stress S_{tr} is admissible or not, a test has to be performed on the sign of the yield function f leading to the two options here after:

- If $f \leq 0$, the whole step is fully elastic, which means that the predicted stress is admissible and we conclude that $\mathsf{S}_1 = \mathsf{S}_{tr}$.
- If $f > 0$, the trial stress is not admissible and a plastic correction has to be performed in order to compute the final value of S_1.

The plastic correction is computed enforcing the (discrete) generalized consistency parameter $f(\Gamma) = 0$ at the end of the current increment (so that time $t = t_1$):

$$f(\Gamma) = \sqrt{\frac{3}{2}\mathsf{S}_1 : \mathsf{S}_1} - \sigma_1^y(\overline{\varepsilon}_1^p, \dot{\overline{\varepsilon}}_1^p, T_1) = 0 \tag{16}$$

with:

$$\begin{cases} \mathsf{S}_1 = \mathsf{S}_{tr} - 2G\Gamma\mathsf{n} \\ \overline{\varepsilon}_1^p = \overline{\varepsilon}_0^p + \sqrt{\frac{2}{3}}\Gamma \\ \dot{\overline{\varepsilon}}_1^p = \frac{1}{\Delta t}\sqrt{\frac{2}{3}}\Gamma \\ T_1 = T_0 + \frac{\eta}{2\rho C_p}\sqrt{\frac{2}{3}}(\sigma_0^y + \sigma_1^y)\Gamma \end{cases} \tag{17}$$

Combination of equations (16) and (17) leads to the following final form of the consistency parameter $f(\Gamma)$:

$$f(\Gamma) = \overline{\sigma}_{tr} - \sqrt{6}G\Gamma - \sigma_1^y(\Gamma) = 0 \tag{18}$$

2.3 Root-finding of the non-linear equation

where the only unknown value is the scalar Γ, already defined in equation (11). If one assume here a linear hardening of the material during the time-step, as usually done thanks to the explicit integration scheme used in Abaqus/Explicit, the following analytical solution to equation (16) can be written, as proposed for example by Gao et *al.* [2] or the Abaqus manual [3]:

$$\Gamma = \sqrt{\frac{3}{2}} \frac{\overline{\sigma}_{tr} - \sigma_0^y}{3G + h} \tag{19}$$

with $h = d\sigma_0^y / d\bar{\varepsilon}_0^p$, the slope of the hardening law at the current point, assumed to be constant during the time increment Δt. This approach is usually adopted in VUMAT implementations because of its simplicity as this later doesn't require any iteration to solve the proposed problem. Unfortunately, as it will be presented further, this lead to many instabilities because of the approximation proposed by this approach when the non-linear terms of the constitutive flow law becomes important. In our approach, we have therefore chosen to solve equation (18) using an approach similar to the one presented in Zaera et *al.* [13], the so called root finding methods used to solve equation (11) will be presented in section 2.3.

Once the correct final value of the plastic corrector Γ has been obtained, the final values of the deviatoric part of the stress tensor $\mathbf{S}_1$ and the updated internal variables are computed thanks to equations (17). At the end of the proposed algorithm, the final stress at the end of the increment is computed using the following equation:

$$\boldsymbol{\sigma}_1 = \mathbf{S}_1 + p_1 \mathbf{1} \tag{20}$$

2.3 Root-finding of the non-linear equation

Several root-finding methods can be used to find the numerical solution of equation (18). In our kind of applications, after a correct selection of the bounds of the searching interval for the solution, we know that the requested root is bracketed in a given interval. Once we know that the proposed interval contains a root, several classical procedures are available to refine it. Among the proposed methods, some of them are known to have fast convergence rate, precision and/or robustness [14, 15]. Unfortunately, the methods that are guaranteed to converge are known to be the slower ones, while those that exhibit a fast convergence rate can also dash rapidly to infinity without warning if measures are not taken to avoid such behavior. Among the large list of available methods (Chord, Bisection, Regula-Falsi, Ridders', Newton-Raphson), only two of them have finally been selected for their efficiency and robustness and are presented here after: the bisection method, a slow and sure method and the Newton-Raphson method a fast but sometimes failing method.

2.3.1 The bisection method

The first root-finding method presented in this paper is the so-called bisection method. This method is one that cannot fail, but with a slow rate of convergence. It's an iterative root-finding method where the predicted interval of the root is successively halved until it becomes sufficiently small, which is also called interval halving method. This process is repeated until the interval is small enough, which satisfies:

$$|x_{i+1} - x_i| \leq \boldsymbol{\varepsilon}_{bis} \tag{21}$$

where $\boldsymbol{\varepsilon}_{bis}$ is the error tolerance of the bisection method. Assuming the root to be found is initially bracketed in an interval $[x_0,\ x_1]$ satisfying $f(x_0)\, f(x_1) < 0$, the method converges linearly, which is comparatively slow, but the main advantage of this method is that it only requires the

evaluation of the function $f(x)$ itself and not its derivative $f'(x)$ as in other methods such as the Newton-Raphson. Therefore, this method can become competitive when the computation of the derivative of the function $f(x)$ becomes long to compute; i.e. when the expression of $f(x)$ is complex.

2.3.2 The safe version of Newton-Raphson method

The Newton-Raphson method [14, 15] is the best known method of finding roots because of its simplicity and efficiency (very high rate of convergence). The only apparent drawback of this method is that it requires the evaluation of the derivative $f'(x)$ of the function $f(x)$, so it's only usable in problems where $f'(x)$ can be readily computed, or numerically evaluated as we will see further. From Taylor series expansion of $f(x)$ about x, we obtain the following expression:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{22}$$

The Newton-Raphson is an iterative process starting with a first guess x_0 for a root of the function $f(x)$ and the process is repeated until the convergence criterion given by:

$$|x_{i+1} - x_i| \leq \boldsymbol{\varepsilon}_{NR} \tag{23}$$

where $\boldsymbol{\varepsilon}_{NR}$ is the error tolerance of the Newton-Raphson method, is reached.

In some situations, the Newton-Raphson method appears to have poor global convergence, because the tangent line is not always an acceptable approximation of the function. But more important, when $f(x)$ is a piecewise function defined by two different expressions on the left and right side of a given point x_s, $f(x)$ will not be differentiable at x_s, leading to numerical difficulties and divergence of the Newton-Raphson algorithm when the seeking solution x_i crosses the point x_s. As a result, the so-called safe version of Newton-Raphson, described in Figure 1, resulting from a combination of the Newton-Raphson and the bisection methods is proposed here after.

If there is a root in the interval $[x_0,\ x_1]$ satisfying $f(x_0)\, f(x_1) < 0$, the safe version of Newton-Raphson method regards the midpoint of $[x_0,\ x_1]$ as the first guess of the root and the Newton-Raphson iteration starts. After each iteration, the interval is updated by replacing one of the two boundaries with the new solution. If the iteration is out of the interval, it is disregarded and replaced with a bisection step to re-position the initial guess. The Newton-Raphson algorithm requires to calculate the derivative $f'(\Gamma)$ of the yield function $f(\Gamma)$ with respect to the Γ parameter when solving equation (18), so that:

$$f'(\Gamma) = -\sqrt{6}G - \frac{d\sigma^y(\Gamma)}{d\Gamma} \tag{24}$$

with the following definition for the derivative of the yield function:

$$\begin{aligned} \frac{d\sigma^y(\Gamma)}{d\Gamma} &= \frac{\partial\sigma^y}{\partial\overline{\varepsilon}^p}\frac{d\overline{\varepsilon}^p}{d\Gamma} + \frac{\partial\sigma^y}{\partial\,\dot{\overline{\varepsilon}}^p}\frac{d\,\dot{\overline{\varepsilon}}^p}{d\Gamma} + \frac{\partial\sigma^y}{\partial T}\frac{dT}{d\Gamma} \\ &= \sqrt{\frac{2}{3}}\left(\frac{\partial\sigma^y}{\partial\overline{\varepsilon}^p} + \frac{1}{\Delta t}\frac{\partial\sigma^y}{\partial\,\dot{\overline{\varepsilon}}^p} + \frac{\eta\sigma^y}{\rho C_p}\frac{\partial\sigma^y}{\partial T}\right) \end{aligned} \tag{25}$$

A common method to calculate the derivatives of the yield function σ^y with respect to $\overline{\varepsilon}^p$, $\dot{\overline{\varepsilon}}^p$ and T is to use an analytical method, which is to calculate the analytical expression for every

2.3 Root-finding of the non-linear equation

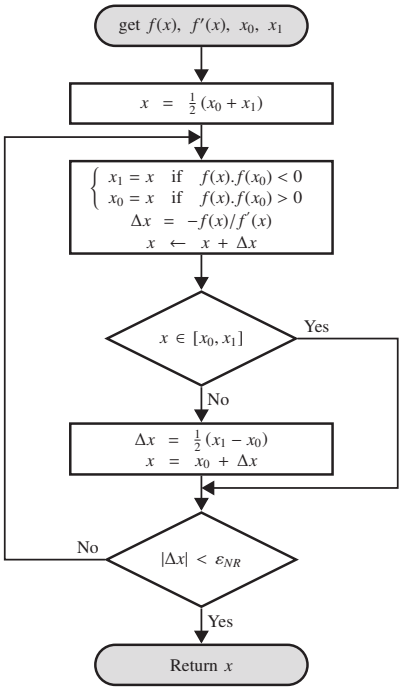


Figure 1: Flowchart of the Safe-NR algorithm

partial derivative based on the hardening flow law of the material. However, for most yield functions, it may be difficult to obtain derivatives using the analytical method. Therefore, in such cases, we propose here after a numerical solution as an alternative. This numerical solution is obtained by adding a small increment to the equivalent plastic strain, plastic strain rate and temperature respectively in order to calculate the three partial derivatives $\frac{\partial \sigma^y}{\partial \overline{\varepsilon}^p}$, $\frac{\partial \sigma^y}{\partial \dot{\overline{\varepsilon}}^p}$ and $\frac{\partial \sigma^y}{\partial T}$ in equation (25):

$$\frac{\partial \sigma^y}{\partial \overline{\varepsilon}^p} = \frac{\sigma^y(\overline{\varepsilon}^p + \Delta\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T) - \sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)}{\Delta\overline{\varepsilon}^p} \tag{26}$$

$$\frac{\partial \sigma^y}{\partial \dot{\overline{\varepsilon}}^p} = \frac{\sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p + \Delta\dot{\overline{\varepsilon}}^p, T) - \sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)}{\Delta\dot{\overline{\varepsilon}}^p} \tag{27}$$

$$\frac{\partial \sigma^y}{\partial T} = \frac{\sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T + \Delta T) - \sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)}{\Delta T} \tag{28}$$

Of course, accurate results depends on a correct choice for the three increments $\Delta\overline{\varepsilon}^p$, $\Delta\dot{\overline{\varepsilon}}^p$ and ΔT. In all following numerical tests, the three increments have been arbitrary fixed to the same value $\Delta x = 10^{-6}$.

3 Hardening flow law

3.1 The Johnson-Cook hardening flow law

In this paper, the Johnson-Cook hardening flow law [4, 16] has been selected to be implemented in Abaqus/Explicit through VUMAT and VUHARD subroutines. Although the proposed approach can be applied to implement other elastoplastic constitutive laws, the Johnson-Cook law is the best choice for validating the proposed approach, because it has already been implemented in Abaqus/Explicit code, which means the benchmark tests can be conducted between the Abaqus Built-In constitutive law and our FORTRAN implementations.

The Johnson-Cook hardening flow law is probably the most widely used flow law for the simulation of high strain rate deformation processes taking into account plastic strain, plastic strain rate and temperature effects. Since a lot of efforts have been made in the past to identify the constitutive flow law parameters for many materials, it is implemented in numerous Finite Element codes such as Abaqus [1]. The general formulation $\sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)$ is given by the following equation:

$$\sigma^y = \left(A + B\overline{\varepsilon}^{p^n}\right)\left[1 + C\ln\left(\frac{\dot{\overline{\varepsilon}}^p}{\dot{\overline{\varepsilon}}_0}\right)\right]\left[1 - \left(\frac{T - T_0}{T_m - T_0}\right)^m\right] \tag{29}$$

where $\dot{\overline{\varepsilon}}_0$ is the reference strain rate, T_0 and T_m are the reference temperature and the melting temperature of the material respectively and A, B, C, n and m are the five constitutive flow law parameters. The multiplicative formulation of this flow law allows for the following form:

$$\sigma^y = \left(A + B\overline{\varepsilon}^{p^n}\right)\sigma^y_{\dot{\varepsilon}}(\dot{\overline{\varepsilon}}^p)\sigma^y_T(T) \tag{30}$$

where, according to [4, 13, 17], the dependence on the plastic strain rate $\sigma^y_{\dot{\varepsilon}}(\dot{\overline{\varepsilon}}^p)$ is only taken into account if $\dot{\overline{\varepsilon}}^p \geq \dot{\overline{\varepsilon}}_0$:

$$\begin{cases} \sigma^y_{\dot{\varepsilon}}(\dot{\overline{\varepsilon}}^p) = 1 + C\ln\left(\dfrac{\dot{\overline{\varepsilon}}^p}{\dot{\overline{\varepsilon}}_0}\right) & \text{if} \quad \dot{\overline{\varepsilon}}^p \geq \dot{\overline{\varepsilon}}_0 \\ \sigma^y_{\dot{\varepsilon}}(\dot{\overline{\varepsilon}}^p) = 1 & \text{if} \quad \dot{\overline{\varepsilon}}^p < \dot{\overline{\varepsilon}}_0 \end{cases} \tag{31}$$

and the dependence on temperature $\sigma^y_T(T)$ is defined so that, if $T < T_0$ there is no temperature dependence of the yield stress and if $T \geq T_m$ the material is assumed to behave like liquid:

$$\begin{cases} \sigma^y_T(T) = 1 - \left(\frac{T - T_0}{T_m - T_0}\right)^m & \text{if} \quad T_0 \leq T \leq T_m \\ \sigma^y_T(T) = 1 & \text{if} \quad T < T_0 \\ \sigma^y_T(T) = 0 & \text{if} \quad T \geq T_m \end{cases} \tag{32}$$

Those two conditions defined by equations (31) and (32) lead to some discontinuities in the hardening relation $\sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)$, its derivative $h(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)$ and the yield function itself resulting in numerical difficulties in the iterative solving procedure. The yield function is therefore not differentiable at $\dot{\overline{\varepsilon}}_0$ and T_0. Nevertheless, the robust Newton-Raphson procedure proposed in section 2.3.2 has been found sufficiently robust to overcome those problems. Analytical forms

3.2 VUHARD implementation in Abaqus/Explicit

for the derivatives of the Johnson-Cook flow law σ^y with respect to $\overline{\varepsilon}^p$, $\dot{\overline{\varepsilon}}^p$ and T are given by the three following equations:

$$\frac{\partial \sigma^y}{\partial \overline{\varepsilon}^p} = nB\overline{\varepsilon}^{p^{n-1}} \sigma^y_{\dot{\varepsilon}}(\dot{\overline{\varepsilon}}^p)\sigma^y_T(T) \tag{33}$$

$$\begin{cases} \frac{\partial \sigma^y}{\partial \dot{\overline{\varepsilon}}^p} = \frac{C}{\dot{\overline{\varepsilon}}^p}\left(A + B\overline{\varepsilon}^{p^n}\right)\sigma^y_T(T) & \text{if} \quad \dot{\overline{\varepsilon}}^p \geq \dot{\overline{\varepsilon}}_0 \\ \frac{\partial \sigma^y}{\partial \dot{\overline{\varepsilon}}^p} = 0 & \text{if} \quad \dot{\overline{\varepsilon}}^p < \dot{\overline{\varepsilon}}_0 \end{cases} \tag{34}$$

$$\begin{cases} \frac{\partial \sigma^y}{\partial T} = \frac{-m\left(A+B\overline{\varepsilon}^{p^n}\right)}{T-T_0}\sigma^y_{\dot{\varepsilon}}(\dot{\overline{\varepsilon}}^p)\left(\frac{T-T_0}{T_m-T_0}\right)^m & \text{if} \quad T \in [T_0, T_m] \\ \frac{\partial \sigma^y}{\partial T} = 0 & \text{if} \quad T \notin [T_0, T_m] \end{cases} \tag{35}$$

The last problem usually encountered is that, because of the term $nB\overline{\varepsilon}^{p^{n-1}}$ in equation (33), h tends to infinity when the first plastic increment occurs (i.e. when $\overline{\varepsilon}^p = 0$). Therefore, we will have to apply a special treatment in the hardening parameter computation on the first plastic step. This will be presented further in section 4.3.

3.2 VUHARD implementation in Abaqus/Explicit

The VUHARD subroutine is a straightforward approach to implement a new constitutive flow law in Abaqus/Explicit by just implementing a FORTRAN subroutine to compute the yield stress of the material $\sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)$ and its derivatives with respect to $\overline{\varepsilon}^p$, $\dot{\overline{\varepsilon}}^p$ and T. The main part of the Built-In constitutive law is used for time integration of the stress, for a given time increment, and the provided user subroutine is used to compute the hardening flow law. Very few details are given about this implementation in the Abaqus documentation, but some useful informations are available in Jansen van Rensburg et *al.* [18].

The numerical implementation of the VUHARD subroutine for the Johnson-Cook flow has been done through a FORTRAN program defining the hardening flow law $\sigma^y(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T)$ according to equation (29) and the three analytical derivatives of σ^y with respect to $\overline{\varepsilon}^p$, $\dot{\overline{\varepsilon}}^p$ and T defined by equations (33-35). This part of the code is exactly the same as the one that will be used in the VUMAT subroutine presented in the next section of this paper.

4 VUMAT implementation in Abaqus/Explicit

All details concerning the implementation of the radial return mapping algorithm using a Fortran VUMAT subroutine for the Johnson-Cook flow law are detailed in this section. The detailed flowchart algorithm is illustrated in Figure 2. The first block of the proposed algorithm "Start of VUMAT" is used to get the material properties defined as user material constants. Abaqus/Explicit provides the user subroutine VUMAT the quantities defined here after:

- The strain increment $\Delta\mathsf{E}$ for the current time step,
- The stress tensor $\boldsymbol{\sigma}_0$ and the temperature T_0 at the beginning of the current increment,
- The time increment Δt corresponding to the current time step,
- A table of solution dependent state variables (SDVs) used to store important data such as $\overline{\varepsilon}^p$, $\dot{\overline{\varepsilon}}^p$, Γ and transfer them from one increment to the other.

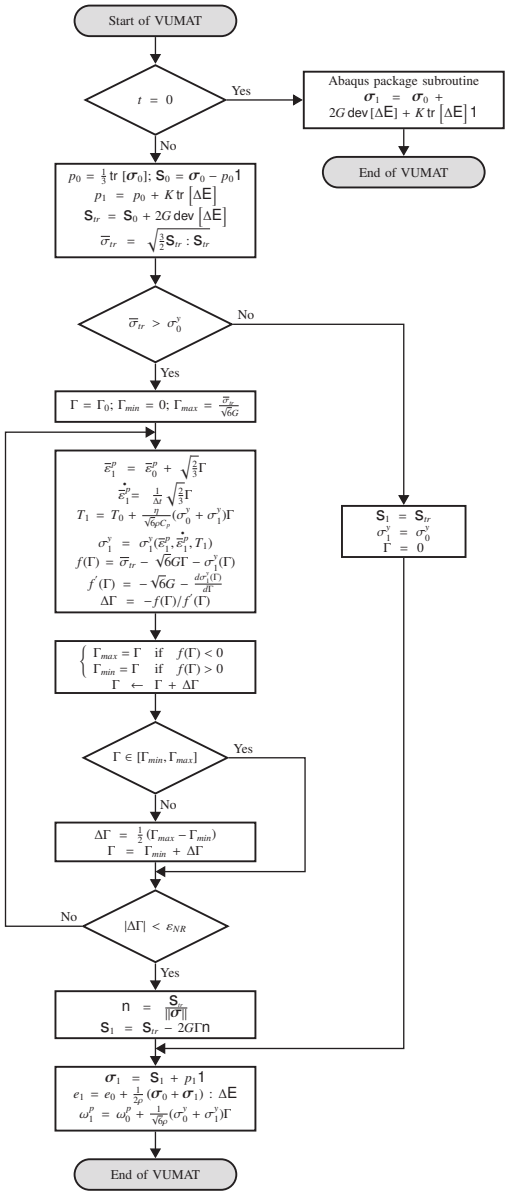


Figure 2: Flowchart of VUMAT implementation

4.1 Abaqus package subroutine

The VUMAT subroutine must compute and return the value of the stress $\boldsymbol{\sigma}_1$ and the SDVs variables at the end of the increment for each integration point. The internal and dissipated energies have also to be evaluated in order to compute the temperatures in the model.

4.1 Abaqus package subroutine

The package part in the Abaqus software is a mandatory step used to compute the starting values (compute the value of the time increment Δt for example by taking into account the material behaviour) for a reference time $t = 0$. For this we need to compute the elastic stress due to a virtual strain increment $\Delta\mathsf{E}$ provided by Abaqus using the following expression:

$$\boldsymbol{\sigma}_1 = \boldsymbol{\sigma}_0 + 2G\,\mathsf{dev}\left[\Delta\mathsf{E}\right] + K\,\mathsf{tr}\left[\Delta\mathsf{E}\right]\mathsf{1} \tag{36}$$

The computed stress $\boldsymbol{\sigma}_1$ will be discarded after the end of the package step.

4.2 Elastic predictor

The aim of this subsection is to calculate the deviatoric part of the predicted elastic stress S_{tr} and the von Mises equivalent stress $\overline{\sigma}_{tr}$ and compare it with the yield stress at the beginning of the increment σ_0^y in order to test if the current step is fully elastic or partly plastic. Therefore, we need firstly to decompose the initial stress $\boldsymbol{\sigma}_0$ into its hydrostatic pressure (p_0) and its deviatoric (S_0) parts:

$$\left\{\begin{array}{l} p_0 = \frac{1}{3}\,\mathsf{tr}\,[\boldsymbol{\sigma}_0] \\ \mathsf{S}_0 = \boldsymbol{\sigma}_0 - p_0\mathsf{1} \end{array}\right. \tag{37}$$

Then, we compute the new pressure (p_1) and the deviatoric part of the trial stress tensor (S_{tr}) using:

$$\left\{\begin{array}{l} p_1 = p_0 + K\,\mathsf{tr}\left[\Delta\mathsf{E}\right] \\ \mathsf{S}_{tr} = \mathsf{S}_0 + 2G\,\mathsf{dev}\left[\Delta\mathsf{E}\right] \end{array}\right. \tag{38}$$

and compare the yield stress at the beginning of the increment $\sigma_0^y(\overline{\varepsilon}_0^p, \dot{\overline{\varepsilon}}_0^p, T_0)$ to the von Mises equivalent trial stress $\overline{\sigma}_{tr}$ (see equation (15) for this later). Depending on this test, we will then correct or not the final stress.

- If $\overline{\sigma}_{tr} \leq \sigma_0^y$, the plastic corrector is zero, so the plastic correction steps can be skipped. We assume that the predicted stress is the final one $\mathsf{S}_1 = \mathsf{S}_{tr}$, the plastic corrector $\Gamma = 0$, the yield stress remains unchanged $\sigma_1^y = \sigma_0^y$ and we can go directly to the final computations in section 4.4.

- If $\overline{\sigma}_{tr} > \sigma_0^y$, the step is at least partly plastic and the plastic corrector described in section 4.3 has to be computed in order to draw back the predicted stress onto the yield surface of the material at the end of the current increment.

4.3 Plastic corrector

The safe version of the Newton-Raphson algorithm is used to recover the stress in accordance with the elastoplastic constitutive behavior law. This Newton-Raphson method is used to compute the Γ parameter defining the correction due to the increase of the strain. In order to enhance the computations efficiency, we initialize the value of the Γ parameter to its value at the end of the previous increment ($\Gamma = \Gamma_0$). If the current increment is the first plastic increment (i.e. if

4.4 Final computations

$\overline{\varepsilon}_0^p = 0$), as we cannot compute the value of $h(\Gamma)$ when the plastic strain is zero, we initialize Γ to an initial value different of zero ($\Gamma = 10^{-8}$). The interval for the bisection part is initialized so that $\Gamma \in \left[0, \overline{\sigma}_{tr}/\sqrt{6}G\right]$. The predicted equivalent plastic strain $\overline{\varepsilon}_1^p$, plastic strain rate $\dot{\overline{\varepsilon}}_1^p$ and temperature T_1 at the end of the increment are computed thanks to the system of equations (17). The yield stress σ_1^y, the yield function $f(\Gamma)$ and its derivative $f'(\Gamma)$ are then computed. Then, we test the convergence of the Newton-Raphson algorithm by computing the increment $\Delta\Gamma$ of the Γ parameter:

$$\Delta\Gamma = -\frac{f(\Gamma)}{f'(\Gamma)} \tag{39}$$

and comparing it to the Newton-Raphson precision $\boldsymbol{\varepsilon}_{NR}$ defined by the user:

- If $\Delta\Gamma > \boldsymbol{\varepsilon}_{NR}$ we need to iterate to compute the correction of the Γ value using the following relation:
$$\Gamma \leftarrow \Gamma + \Delta\Gamma \tag{40}$$
and re-evaluate the values of $f(\Gamma)$ and $f'(\Gamma)$.
- If $\Delta\Gamma \leq \boldsymbol{\varepsilon}_{NR}$ we have obtained the final value of the Γ parameter.

The final deviatoric part of the stress tensor $\mathbf{S}_1$ is computed from the predicted value $\mathbf{S}_{tr}$ and the correction term Γ using:

$$\mathbf{S}_1 = \mathbf{S}_{tr} - 2G\Gamma\mathbf{n} \tag{41}$$

with $\mathbf{n}$, within the framework of the radial return algorithm, defined by:

$$\mathbf{n} = \frac{\mathbf{S}_{tr}}{\sqrt{\mathbf{S}_{tr} : \mathbf{S}_{tr}}} \tag{42}$$

4.4 Final computations

The main work of the final computations is to update the state variables (SDVs) and the energies. The final stress tensor at the end of the increment $\boldsymbol{\sigma}_1$ is computed from the hydrostatic pressure p_1 and the deviatoric part of the stress tensor $\mathbf{S}_1$ thanks to equation (20). The equivalent plastic strain $\overline{\varepsilon}_1^p$, equivalent plastic strain rate $\dot{\overline{\varepsilon}}_1^p$ and final temperature T_1 at the end of the increment are stored in their respective SDVs variables for subsequent re-use in the next increment. We also have to compute the new specific internal energy e_1 from:

$$e_1 = e_0 + \frac{1}{2\rho}(\boldsymbol{\sigma}_0 + \boldsymbol{\sigma}_1) : \Delta\mathbf{E} \tag{43}$$

and the dissipated inelastic energy ω_1^p from:

$$\omega_1^p = \omega_0^p + \frac{1}{\sqrt{6}\rho}(\sigma_0^y + \sigma_1^y)\Gamma \tag{44}$$

At this point, the VUMAT subroutine comes to an end. At this point of the flowchart, the final temperature is not computed, since the software uses a subsequent thermal step to evaluate the temperature raise due to the dissipated inelastic energy and the conduction part. The internal energy is modified during this thermal step and this seems to be taken into account during this extra step (out of the VUMAT subroutine) since the tests on 1 element with imposed displacement of all nodes gives the exact same results as the Build-in routine.

Table 1: Material properties of the 42CrMo4 steel

E	ν	A	B	C
(Gpa)		(MPa)	(MPa)	
206.9	0.29	806	614	0.0089
n	m	$\dot{\bar{\varepsilon}}_0$	T_0	T_m
		(s^{-1})	$(°C)$	$(°C)$
0.168	1.1	1	20	1540
ρ	λ	C_p	η	
(kg/m^3)	$(W/m°C)$	$(J/Kg°C)$		
7830	34.0	460	0.9	

5 Validation of the proposed implementations

In this section, the performance of the Johnson-Cook law programmed in the VUMAT subroutine is compared with the performance of the Abaqus/Explicit Built-In Johnson-Cook law and of the VUHARD Johnson-Cook implementation, in order to validate the proposed implementation approach. The benchmark tests consist of two different one element tests (tensile test and shear test), the necking of a circular bar and the well known Taylor impact test. The same material, a 42CrMo4 steel, has been selected for all those tests, and material properties are reported in Table 1. The proposed benchmarks are using the "Dynamic Temperature-displacement, Explicit" procedure of Abaqus explicit FEM code. The inelastic heat fraction parameters has been set to its default value of $\eta = 0.9$. This thermomechanical coupling option allows heat to be generated by plastic dissipation or viscoelastic dissipation. No other thermal boundary conditions are applied *i.e.* a globally adiabatic situation.

All benchmarks tests have been solved using Abaqus/Explicit v.6.14 on a Dell Precision T7500 computer running Ubuntu 16.04 64bits with 12Gb of Ram and two 4 core E5620 Intel Xeon Processors. All computations have been done using the double precision option of Abaqus, with one CPU and the VUMAT and VUHARD subroutines have been compiled using the Intel Fortran 64 v.14 compiler. The following models have been tested for each of the four presented benchmarks:

- **A-N-R** model: VUMAT model with Newton-Raphson procedure and an analytical computation of the derivatives using equations (33-35).
- **N-N-R** model: VUMAT model with Newton-Raphson procedure and a numerical computation of the derivatives using equations (26-28).
- **Direct** model: VUMAT model with a direct evaluation of Γ using equation (19) as presented in other papers [2].
- **VUHARD** model: Only the Johnson-Cook constitutive flow law defined by equation (29) and its three analytical derivatives (33-35) have been implemented using a FORTRAN subroutine.
- **Built-In** model: native implementation of the Built-In Johnson-Cook constitutive law, in order to compare the results with a reference solution.

5.1 One element benchmark tests

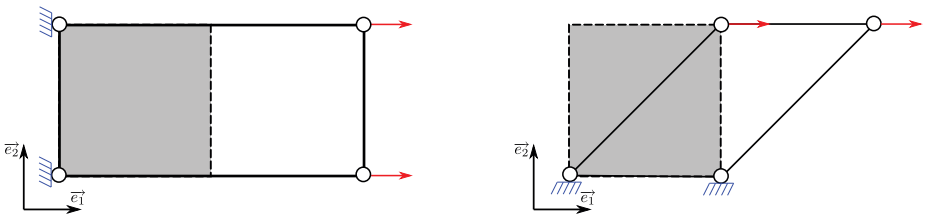


Figure 3: Numerical models for the one element tensile and shear tests

Tests have also been done using the Bisection method, but we have made the choice to omit them in this section to reduce the number of results because those results are very closed to the A-N-R model but computing time is 10% to 30% longer than the A-N-R and the N-N-R models. At this point, it has to be noted that, Abaqus/Explicit doesn't use the same objective stress rate when using the Built-In implementation and the VUMAT subroutine. As reported in documentation [1] and confirmed by our own tests based on a one element hyperelastic shear test, a Jaumann rate is used for both the Built-In formulation and the VUHARD subroutine while a Green-Naghdi rate is used for the VUMAT subroutine. Therefore, a straightforward comparison of Built-In and VUMAT results is not possible when large rotations occur.

5.1 One element benchmark tests

One element test, also called single element test, is a very simple and practical method to investigate the accuracy and sensitivity of the behavior of an element to the external loading. In this subsection, the deformation of a 4-node bilinear displacement and temperature, reduced integration with hourglass control element CPE4RT will be simulated using the proposed VUMAT subroutine and the Abaqus Built-In model. As only one under-integrated element is used, we only have one integration point located at the center of the element. In this kind of test, all nodes of the element are constrained with a prescribed displacement. As the geometry change is exactly the same in each of the two following benchmarks, it will be easy to compare the results in terms of plastic deformations, stresses and temperatures. The original size of the element is $10\,mm \times 10\,mm$.

5.1.1 One element tensile test

In this first benchmark, the two left nodes of the element are encastred and a prescribed horizontal displacement $d = 10\,mm$ is applied on the two right nodes of the same element as illustrated in Figure 3. As we are using an explicit integration scheme, the total simulation time is set to $t = 0.01\,s$. A perfect match between all five results has been found for the one element tensile test, with a final value of the plastic strain $\overline{\varepsilon}^p = 0.457$ and a final temperature $T = 164.09°\text{C}$. Differences in computational time are not appreciable in this test since the final result is obtained in less than $1\,s$.

5.1.2 One element shear test

This second benchmark is similar to the previous one, but now, the two bottom nodes of the element are encastred and a prescribed horizontal displacement $d = 10\,mm$ is applied on the two top nodes of the same element as illustrated in Figure 3. Again, a perfect match between all five results has been found, with a final value of the plastic strain $\overline{\varepsilon}^p = 0.572$ and a final temperature $T = 192.22°\text{C}$.

5.2 Necking of a circular bar

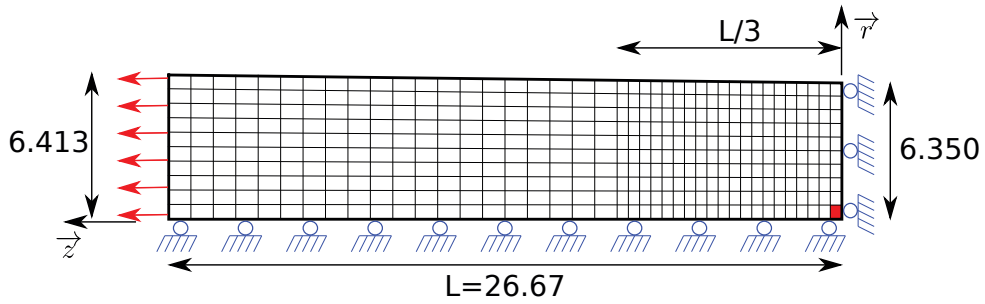


Figure 4: Numerical model for the necking of a circular bar

5.2 Necking of a circular bar

The necking of a circular bar test is useful to evaluate the performance of VUMAT subroutine for materials in presence of plasticity and large deformation [12, 8]. Because of the symmetric structure, an axisymmetric quarter model of the specimen is established. Dimensions of the specimen are reported in Figure 4. The loading is realized through an imposed total displacement of 7 *mm* along the $\vec{z}$ axis on the left side of the specimen while the radial displacement of the same edge is supposed to remain zero. On the opposite side the axial displacement is restrained while the radial displacement is free. The mesh consists of 400 CAX4RT elements with a refined zone of 200 elements on the right side on $1/3$ of the total height. Again, the total simulation time is set to $t = 0.01$ *s* because of the explicit integration scheme adopted.

Figure 5 shows the equivalent plastic strain contourplot of the deformed bar for two models: the Built-In model (left side) and the N-N-R model (right side). The maximum equivalent plastic strain $\overline{\varepsilon}^p$ is located into the center of the bar (so we have chosen to record the time-history evolutions for the red element in Figure 4) and all the models give quite the same values as reported in Table 2 for $\overline{\varepsilon}^p$, $\overline{\sigma}$ and T. Figure 6 shows the evolution of the von Mises stress $\overline{\sigma}$ with the displacement applied at the end of the specimen for the different models. As reported in this figure, the Built-In model, the VUHARD model and the two versions of the Newton-Raphson model give almost the same results. A slight difference can be seen in the Direct model, where there is a little over-estimation of the von Mises stress in the middle of the simulation.

Table 2 reports some results at the end of the computation concerning the total number of increments and the total computing time for all models. As reported in this table, the results of the two versions of the Newton-Raphson model are identical, this tends to prove that there is no difference in using numerical or analytical derivatives of the flow law. The T/I column show the ratio of the total computing time over the total number of increments, *i.e.* the computing time/increment. As reported in Table 2, the native procedure corresponds to the fastest algorithm in terms of computing time. This is easy to explain because this internal procedure is natively optimized within the Abaqus code. The simple fact of transferring all the data to a VUMAT or VUHARD subroutine increases the computing time independently of the optimization of the implemented stress evaluation algorithm. This translates in particular into the fact that the increase in CPU time due to the use of the VUHARD subroutine is about 18%, while the increase for the different versions of the VUMAT routines is about 12 to 15% in terms of T/I ratios.

It is noticeable in this case that the total number of increments needed to perform the whole simulation is lower for the Newton-Raphson models than for the three other models. This difference is illustrated in Figure 7 where the evolution of the time increment Δt during the computa-

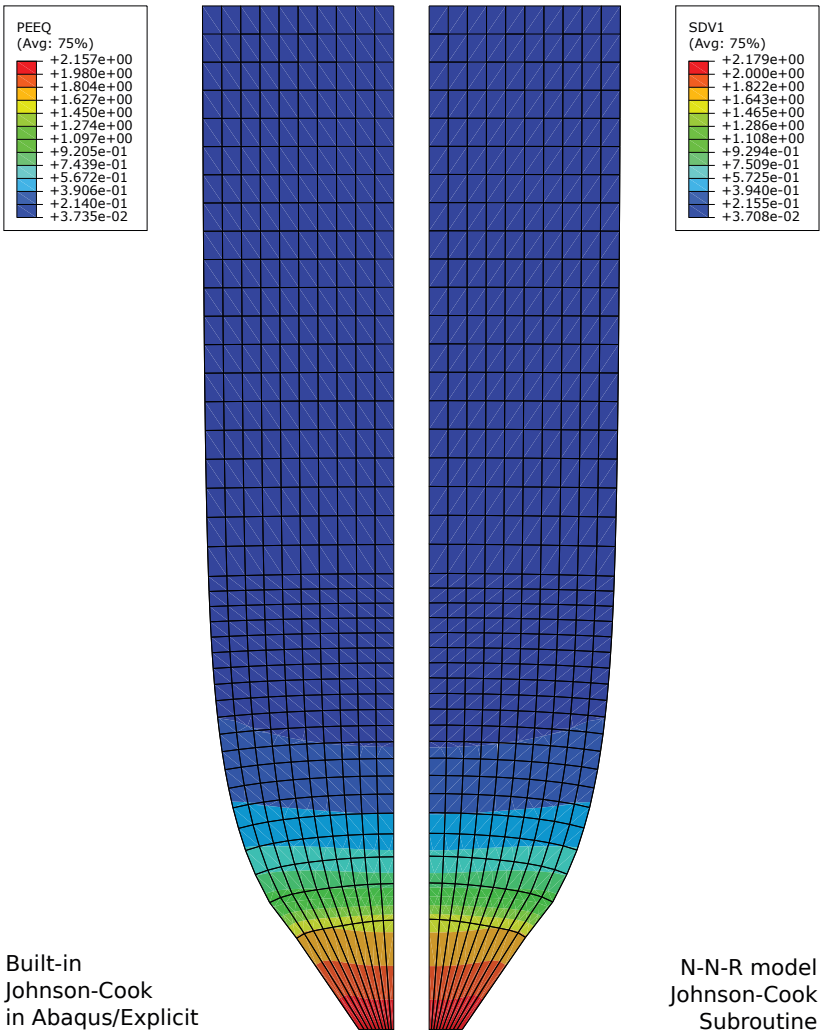


Figure 5: Equivalent plastic strain $\bar{\varepsilon}^p$ contourplot for the necking of a circular bar

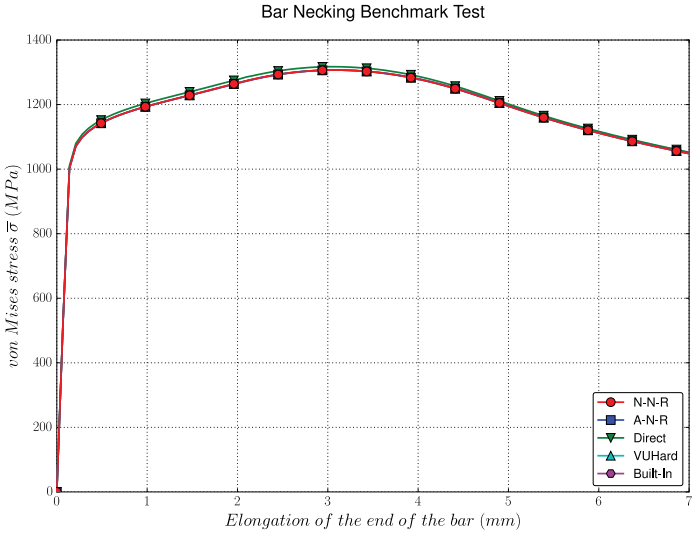


Figure 6: Von Mises stress $\overline{\sigma}$ vs. displacement for the necking of a circular bar

tion is reported. The smoother variation of the time increment, and greatest values, are obtained with both versions of Newton-Raphson, leading to the minimal number of increments to complete the simulation. Using the Built-In model and the VUHARD model, a reduction of the stable time increment from $\Delta t = 6.5\,10^{-8}\,s$ to $\Delta t = 5.2\,10^{-8}\,s$ is noticed after a displacement of $2.52\,mm$, while using the Direct method, a large reduction of the stable increment to $\Delta t = 3.4\,10^{-8}\,s$ with some residual oscillations is encountered after a displacement $2.03\,mm$. The time increment using both versions of the Newton-Raphson model is constant up-to a displacement of $3.36\,mm$ and reduces smoothly after this point because of the striction of the specimen. During the last third of the simulation time increments of the N-R models are almost equal to the one of the Built-In model and the VUHARD model.

From those results we can conclude that the integration of the constitutive equation has an influence on the stable time increment Δt. We can also note, from the results reported in Table 2 that, using the VUMAT Newton-Raphson doesn't increase a lot the total computational time (around 10% more time in this case) as the total number of increments has been reduced by a factor of 3.8%. We can also note that the VUMAT subroutine is faster than the VUHARD subroutine. Of course, computational cost of the VUMAT model can be reduced by optimizing the FORTRAN routine (removing the numerous number of tests inside the subroutine, optimizing the computations and the mathematical expressions, ...). It has also to be noted that the requested precision for the Newton-Raphson subroutine $\boldsymbol{\varepsilon}_{NR} = 10^{-8}$ has also an influence on the computational time, reducing it reduces also the computational time.

5.3 Taylor impact test

5.3.1 Axisymmetric Taylor impact test

Finally, the performance of the proposed VUMAT subroutine is validated under high deformation rate with the simulation of the Taylor impact test [19]. In the Taylor impact test, a

5.3 Taylor impact test

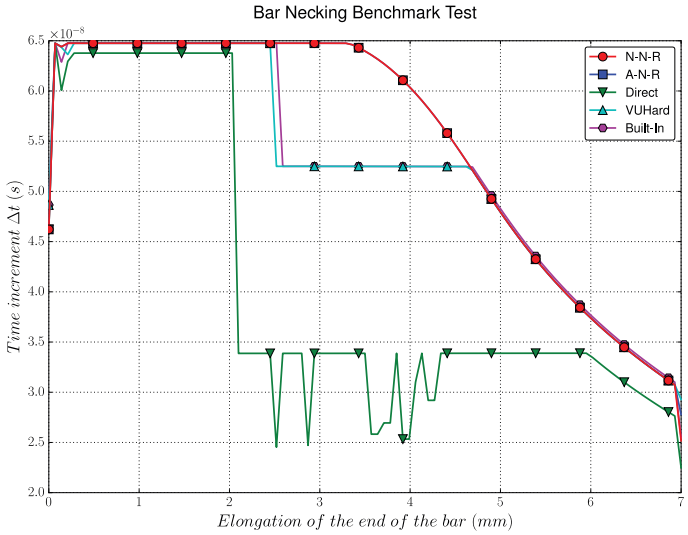


Figure 7: Time increment Δt vs. displacement for the necking of a circular bar

Table 2: Comparison of results for the necking of a circular bar benchmark

Model	Incr.	Time	T/I (μs)	$\overline{\varepsilon}^p$	$\overline{\sigma}$ (MPa)	T (°C)
A-N-R	191 655	$1m\,31s$	469	2.158	1048.15	583.64
N-N-R	191 653	$1m\,32s$	480	2.158	1048.15	583.64
Direct	267 964	$2m\,08s$	478	2.161	1052.60	588.18
VUHARD	200 182	$1m\,39s$	494	2.157	1048.14	583.60
Built-In	199 402	$1m\,23s$	416	2.137	1051.28	579.79

5.3 Taylor impact test

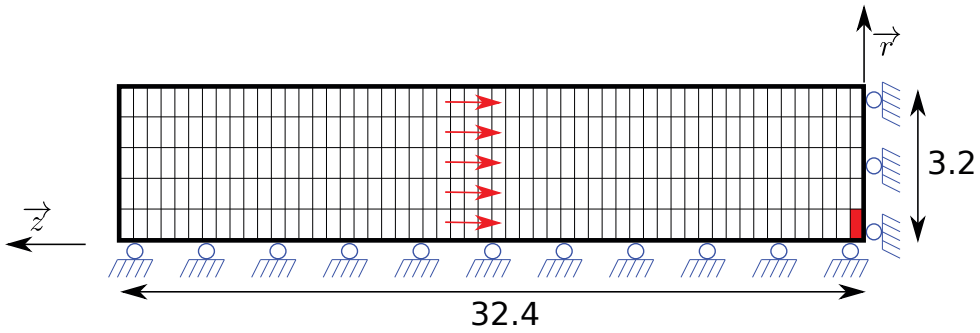


Figure 8: Model of the Taylor impact test

cylindrical specimen is launched to impact a rigid target with a prescribed initial velocity. The numerical model, reported in Figure 8 is established as axisymmetric. The height is 32.4 mm and the radius is 3.2 mm. The axial displacement is restrained on the right side of the specimen while the radial displacement is free (to figure a perfect contact without friction of the projectile onto the target). A predefined velocity of $V_c = 287\, m/s$ is imposed on the specimen. The mesh consists of 250 CAX4RT elements (5×50 elements). The total simulation time for the Taylor impact test is $t = 8.0\, 10^{-5}\, s$.

Figure 9 shows the equivalent plastic strain contourplot of the deformed rod for two models: the Built-In model (left side) and the N-N-R model (right side). The distributions of the equivalent plastic strain are almost the same for both models. The maximum equivalent plastic strain $\overline{\varepsilon}^p$ is located into the center element of the model (the red element in Figure 8) and the models give quite the same value as reported in Table 3 for $\overline{\varepsilon}^p$, T and the final dimensions of the specimen L_f (final length) and D_f (final diameter of the impacting face). Figure 10 shows the evolution of the equivalent plastic strain $\overline{\varepsilon}^p$ with time for the different models for the element at the center of the impacting face (the red element in Figure 8). As reported in this figure and according to the results presented in Table 3, all models give almost the same results.

It can be noticed from this later that the Direct model (i.e. the classic way to program a VUMAT subroutine assuming that the explicit time integration scheme allows to use a straightforward evaluation of the plastic strain) doesn't provide results in accordance with the other models concerning the internal fields and the final geometry of the specimen. This tends to prove that if one wants to perform an inverse characterization of material constitutive parameters for dynamic applications based on a Taylor impact test and a FORTRAN VUMAT subroutine for implementing an exotic constitutive law, this can be source of errors in the identification of the constitutive equation parameters.

5.3.2 3D Taylor impact test

In order to have relatively longer computational times while remaining within the framework of the numerical simulation of the Taylor impact test, the choice was made for a 3D modeling. Therefore, a 3D quarter model of the Taylor cylindrical specimen, with the same dimensions as the Taylor 2D model, as been set-up as reported in Figure 11. The new mesh consists of 37 422 C3D8RT elements which is a quite large simulation benchmark.

The results concerning the CPU times (increments, time and T/I ratio), the final values of deformations and temperatures as well as the geometrical results are shown in Table 4. We can note

5.3 *Taylor impact test*

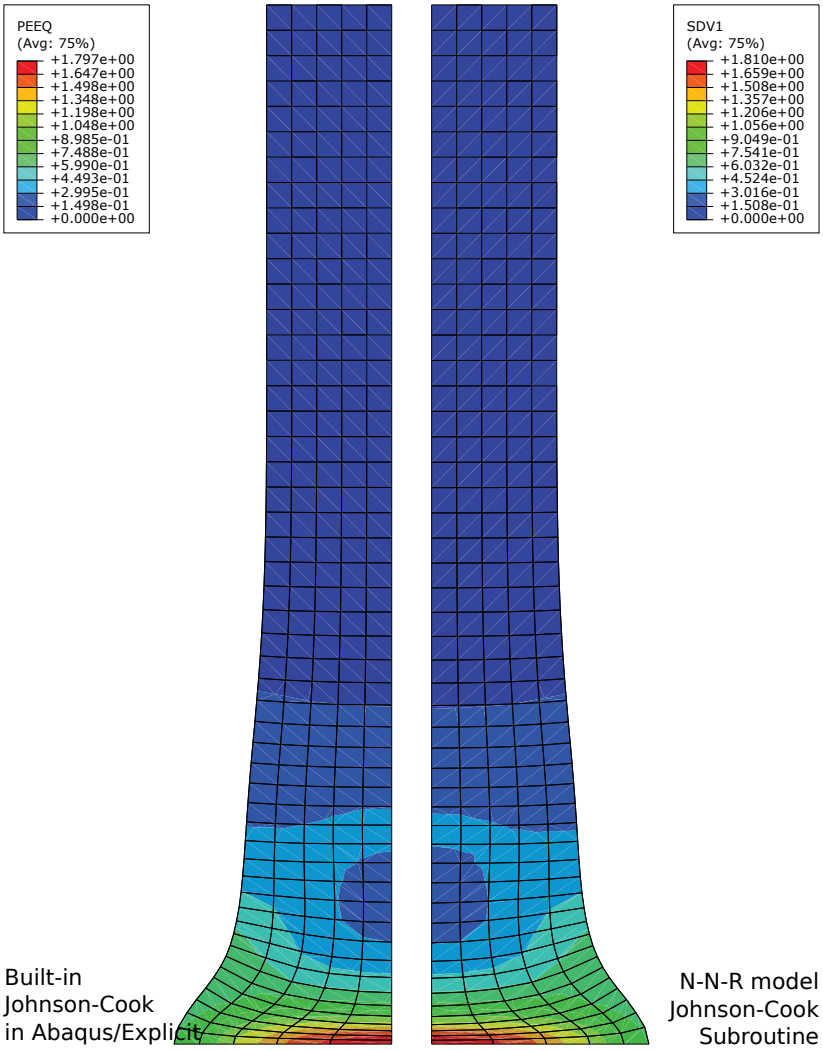


Figure 9: Equivalent plastic strain $\overline{\varepsilon}^p$ contourplot for the Taylor impact test

Table 3: Comparison of results for the Taylor impact test

Model	Incr.	$\overline{\varepsilon}^p$	T (°C)	L_f (mm)	D_f (mm)
A-N-R	3 938	1.810	561.09	26.56	11.10
N-N-R	3 939	1.810	561.09	26.56	11.10
Direct	3 967	1.802	570.44	26.62	11.04
VUHARD	3 849	1.808	561.55	25.55	11.11
Built-In	3 834	1.797	560.17	26.57	11.12

5.3 Taylor impact test

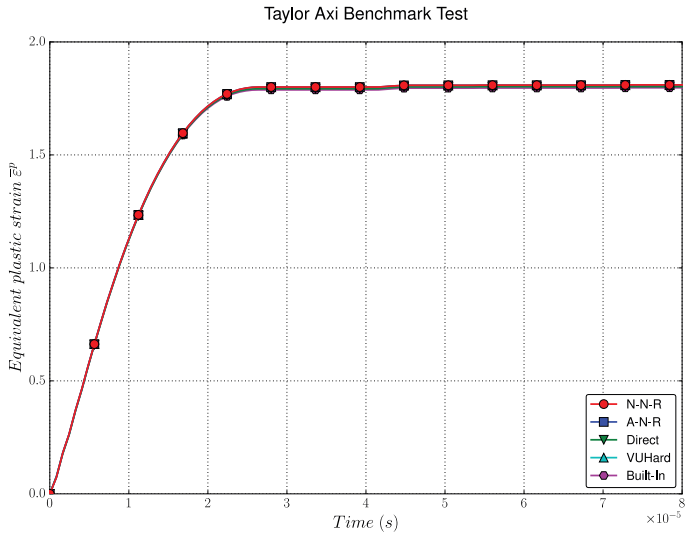


Figure 10: Equivalent plastic strain $\overline{\varepsilon}^p$ vs. time for the Taylor impact test

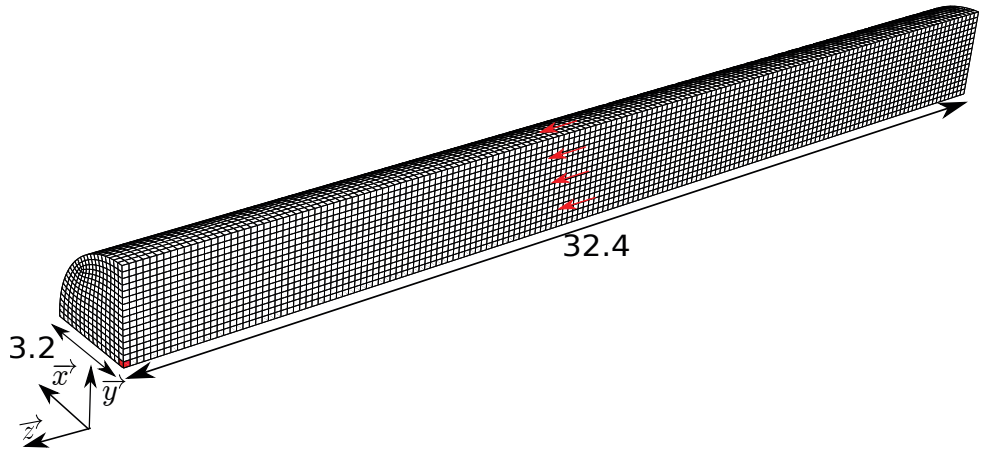


Figure 11: Model of the 3D Taylor impact test

Table 4: Comparison of results for the 3D Taylor impact test

Model	Incr.	Time	T/I (ms)	$\overline{\varepsilon}^p$	T (°C)	L_f (mm)	D_f (mm)
A-N-R	20 702	$23m\,59s$	69.5	8.99	1507	26.42	12.07
N-N-R	20 834	$24m\,38s$	70.9	9.04	1512	26.42	12.06
Direct	19 872	$23m\,23s$	70.6	8.54	1500	26.47	11.95
VUHARD	22 498	$24m\,46s$	66.1	7.53	1434	26.40	11.73
Built-In	22 671	$22m\,44s$	60.2	7.67	1451	26.41	11.80

a very good correlation of the results concerning geometric dimensions, whereas the results in terms of deformation and temperature (very localized results on a very constrained element) differ due to a better taking into account of the constitutive law by the VUMAT subroutine (this has been confirmed by refining the mesh). Similar to the results established in section 5.2 concerning the necking of a circular bar, the T/I ratio is again increased by about 15% for the different versions of the VUMAT subroutines, whereas the overall increase in CPU time is only about 5% due to the reduction of the total number of increments required for simulation by about 10%. Concerning the VUHARD approach, the increase in the T/I ratio is around 10% with approximately the same number of required increments as the Built-In method, leading to a total CPU increase of about 10% also. This proves again the efficiency of the proposed VUMAT algorithm.

5.4 VUMAT implementation of the alternative constitutive laws

In this section, the Johnson-Cook constitutive law and three other constitutive laws are implemented in the N-N-R VUMAT and the Direct VUMAT subroutines to simulate Taylor compression test, in order to validate the application of the proposed algorithm to alternative elastoplastic constitutive laws which follow J_2 plasticity and isotropic hardening and have the general form of $\sigma^y\left(\overline{\varepsilon}^p, \dot{\overline{\varepsilon}}^p, T\right)$. The main advantage of using our N-N-R approach is that it does not require to compute explicitly the derivatives of the hardening law, they are numerically computed using equations (26-28). The other constitutive laws are the TANH constitutive law, modified TANH constitutive law and Bäker constitutive law.

- Calamaz et *al.* [20] proposed the so called TANH constitutive law by adding a term modeling the strain softening to the Johnson-Cook constitutive law, given by:

$$\sigma^y = \sigma_{JC}[D + (1 - D)\tanh(\frac{1}{\overline{\varepsilon}^p + \overline{\varepsilon}_0})] \tag{45}$$

  with:

$$D = 1 - \left(\frac{p\overline{\varepsilon}^p}{1 + p\overline{\varepsilon}^p}\tanh\left(\frac{T - T_0}{T_{rec} - T_0}\right)^q\right) \tag{46}$$

  in which σ_{JC} represents the original Johnson-Cook constitutive law. The constant $\overline{\varepsilon}_0$ can modulate the strain corresponding to the peak stress, p and q are the additional constitutive law parameters, and T_{rec} is the onset temperature for the strain softening phenomenon.

- Hor et *al.* [21] proposed later a constitutive law by modifying the TANH constitutive law, given by:

$$\sigma^y = \sigma_{\overline{\varepsilon}^p}(\overline{\varepsilon}^p, T)\sigma_T(T)\sigma_{\dot{\overline{\varepsilon}}^p}(\dot{\overline{\varepsilon}}^p, T) \tag{47}$$

  where:

$$\begin{cases} \sigma_{\overline{\varepsilon}^p}(\overline{\varepsilon}^p, T) = \left(A + B\overline{\varepsilon}^{p^n}\right)[D + (1 - D)\tanh(\frac{1}{\overline{\varepsilon}^p + \overline{\varepsilon}_0})] \\ \sigma_T(T) = 1 - m_1\left(\frac{T - T_0}{T_m - T_0}\right)^{m_2} \\ \sigma_{\dot{\overline{\varepsilon}}^p}(\dot{\overline{\varepsilon}}^p, T) = 1 + C(T)\ln\left(\frac{\dot{\overline{\varepsilon}}^p}{\dot{\overline{\varepsilon}}_0}\right) \end{cases} \tag{48}$$

  with:

$$D = 1 - (\frac{p\overline{\varepsilon}^p}{1 + p\overline{\varepsilon}^p})\tanh\left(\frac{T - T_0}{T_{rec} - T_0}\right) \text{ and } C(T) = \frac{C_1 exp\left(C_2\frac{T}{T_m}\right)}{\frac{T}{T_m}} \tag{49}$$

and A, B, C_1, C_2, m_1, m_2, n and p are the constitutive law parameters. As with the TANH model, the constant $\overline{\varepsilon}_0$ can modulate the strain corresponding to the peak stress and T_{rec} is the onset temperature for the strain softening phenomenon.

- The last constitutive law we have implemented using the VUMAT subroutine is the one proposed by Bäker [22, 21] and given by:

$$\sigma^y = \left(A\overline{\varepsilon}^{p^{n_0 f(T)}}\right)\left[1 + C\ln\left(\frac{\dot{\overline{\varepsilon}}^p}{\dot{\overline{\varepsilon}}_0}\right)\right] f(T) \tag{50}$$

with:

$$f(T) = exp\left[-\left(\frac{T}{T_m}\right)^m\right] \tag{51}$$

where A, n_0 and m are the temperature-dependent material parameters, and C and $\dot{\overline{\varepsilon}}_0$ are constants.

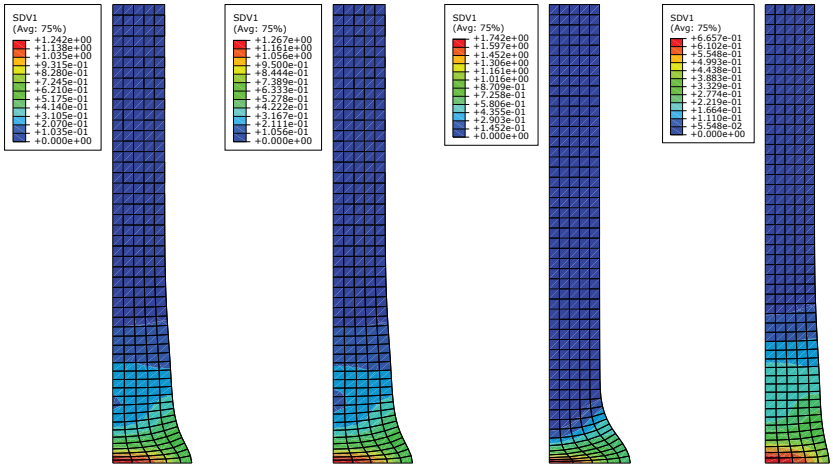
The 2D model used for the simulation is the same as the one introduced in Section 5.3.1. The material used in this case is the 42CrMo4 steel with ferrito-perlitic (referred to as 42CrMo4-FP). The parameters of the four constitutive laws for 42CrMo4-FP steel were proposed by Hor et *al.* [21]. The predefined velocity is set to $V_c = 200m/s$. The equivalent plastic strain contour-plots of the deformed rod for the four models are shown in Figure 12.

The plastic deformation is concentrated at the bottom, and maximum equivalent plastic strain is located in the center element of the specimen. More details concerning the total number of increments, total computing time, maximum equivalent plastic strain $\overline{\varepsilon}^p$, final length L_f, final radius R_f of the bottom and maximum temperature T are reported in Table 5. The results of the Johnson-Cook and TANH models are almost identical. Compared with the previous two models, the modified TANH model achieves larger equivalent plastic strain at the bottom, and correspondingly it also achieves larger final radius and higher temperature at the bottom. However, as we can see, this model has less plastic deformation in the other part. No matter in terms of the maximum equivalent plastic strain and temperature or in terms of the geometric responses, the Bäker model has the least plastic deformation. Concerning the Direct approach, it can be seen that this one usually overestimates the plastic deformation and the temperature for the tree first models. In terms of global results, the three first models give almost the same results, while the Bäker model gives quite different results with regards to the previous ones.

5.5 Discussion on the benchmarks

In all the proposed benchmark tests, it is noteworthy that the results of the VUMAT subroutines and the Abaqus Built-In model are very closed but some slight differences can be pointed. Those differences can mainly be explained by the following remarks:

- The Johnson-Cook constitutive law implemented through the VUMAT subroutine doesn't have exactly the same expression as the Abaqus/Explicit Built-In model because the dependence on the deformation rate has been modified in our implementation as discussed before in section 3. This can be seen in the Bar Necking benchmark where the VUHARD and the Built-In models doesn't provide exactly the same results.



(a) Johnson-Cook (b) TANH (c) Mod. TANH (d) Bäker

Figure 12: Equivalent plastic strain contour-plot of the Taylor specimen for the four constitutive models using the N-N-R VUMAT

Table 5: Results for the Taylor compression test

Model	Incr.	$\overline{\varepsilon}^p$	T ($^\circ C$)	L_f (mm)	D_f (mm)
JC N-N-R	2 355	1.24	307.7	28.26	9.74
JC Direct	2 346	1.23	316.5	28.38	9.63
TANH N-N-R	2 370	1.26	308.1	28.27	9.78
TANH Direct	2 427	1.26	316.4	28.36	9.68
Mod. TANH N-N-R	3 060	1.74	459.5	29.24	10.33
Mod. TANH Direct	3 528	1.76	471.1	29.33	10.25
Bäker N-N-R	1 376	0.67	281.7	29.77	8.32
Bäker Direct	1 925	0.941	263.8	28.23	9.09

- The Built-In model and the VUHARD model are integrated through an explicit central-difference time integration rule, while the radial return method, which belongs to an implicit integration algorithm, is employed in the VUMAT subroutines.
- As the root of the function is not an exact solution, but an approximation, the choice of the precision tolerance has an effect on the final results.
- The Jaumann stress rate is used for the Built-In model and the VUHARD subroutine while the VUMAT routine employs the Green-Naghdi stress rate. This can cause differences in the results in large deformations models when finite rotation of a material point is accompanied by finite shear [1].

6 Conclusions

In this paper, an approach has been proposed for the implementation of elastoplastic constitutive laws in Abaqus/Explicit through VUMAT subroutine. The proposed implementation is robust and efficient, and can be applied to simulate the behavior of various materials defined by any elastoplastic constitutive model following J_2 plasticity and isotropic hardening. In the proposed algorithm, an implicit time integration scheme based on the radial return method is employed, which is unconditionally stable and can ensure that the von Mises yield criterion is always satisfied during computation. Validated through numerical benchmarks, a robust and efficient root-finding method, the so-called safe version of Newton-Raphson method, has been implemented to compute the plastic corrector term.

This algorithm can now be re-used by only changing the expression of the flow law in order to implement alternative constitutive law, such as the revised form of the Johnson-Cook law proposed by Rule et *al.* [16], or any other one of the same family. Applications of this kind of developments mainly concerns inverse identification of constitutive law parameters based on a dynamic impact tests, or programming of alternative constitutive laws in Abaqus/Explicit. The proposed implementation has been found more precise with regards to other kind of VUMAT or VUHARD routines based on a direct evaluation of the plastic strain rate in order to provide accurate results for the identification procedure.

Acknowledgements

This work is partly supported by the scholarship from China Scholarship Council (CSC) under Grant CSC N°201406290010.

References

[1] Abaqus v. 6.14 documentation, Dassault Systemes Simulia Corporation.
[2] C. Gao, Fe realization of a thermo-visco-plastic constitutive model using vumat in abaqus/explicit program, in: Computational Mechanics, Springer, 2007, pp. 301–301.
[3] Abaqus v. 6.14 user subroutines reference guide, Dassault Systemes Simulia Corporation.
[4] G. R. Johnson, W. H. Cook, A constitutive model and data for metals subjected to large strains, high strain rates and high temperatures, in: Proceedings of the 7th International Symposium on Ballistics, Vol. 21, The Hague, The Netherlands, 1983, pp. 541–547.
[5] S. Nemat-Nasser, On finite deformation elasto-plasticity, International Journal of Solids and Structures 18 (10) (1982) 857–872.
[6] M.-H. Yu, Generalized plasticity, Springer Science & Business Media, 2006.

[7] G. I. Taylor, H. Quinney, The latent energy remaining in a metal after cold working, Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character 143 (849) (1934) 307–326.
[8] J. C. Simo, T. J. R. Hughes, Computational inelasticity, Springer, 1998.
[9] M. L. Wilkins, Calculation of elastic-plastic flow, Tech. rep., Alder, B. (Ed.), Methods in Computational Physics, Vol. 3. Academic Press, pp. 211-263 (1963).
[10] G. Maenchen, S. Sacks, The tensor code, in methods of computational physics, 3, 181-210. b. alder, s. fernback and m. rotenberg editors (1964).
[11] F. Dunne, N. Petrinic, Introduction to computational plasticity, Oxford University Press New York, 2005.
[12] J.-P. Ponthot, Unified stress update algorithms for the numerical simulation of large deformation elasto-plastic and elasto-viscoplastic processes, International Journal of Plasticity 18 (1) (2002) 91–126.
[13] R. Zaera, J. Fernández-Sáez, An implicit consistent algorithm for the integration of thermoviscoplastic constitutive equations in adiabatic conditions and finite deformations, International journal of solids and structures 43 (6) (2006) 1594–1612.
[14] F. S. Acton, Numerical methods that work, Maa, 1990.
[15] W. H. Press, Numerical recipes 3rd edition: The art of scientific computing, Cambridge university press, 2007.
[16] W. K. Rule, S. Jones, A revised form for the johnson–cook strength model, International Journal of Impact Engineering 21 (8) (1998) 609–624.
[17] L. Schwer, Optional strain-rate forms for the johnson cook constitutive model and the role of the parameter epsilon 0, in: Proceedings of the 6th European LS-DYNA Users Conference, Anwenderforum, Frankenthal, Germany, 2007.
[18] G. Jansen van Rensburg, S. Kok, Tutorial on state variable based plasticity: an abaqus uhard subroutine.
[19] G. I. Taylor, The testing of materials at high rates of loading, Journal of the institution of civil engineers 26 (8) (1946) 486–519.
[20] M. Calamaz, D. Coupard, F. Girot, Numerical simulation of titanium alloy dry machining with a strain softening constitutive law, Machining Science and Technology 14 (2) (2010) 244–257.
[21] A. Hor, F. Morel, J.-L. Lebrun, G. Germain, Modelling, identification and application of phenomenological constitutive laws over a large strain rate and temperature range, Mechanics of Materials 64 (2013) 91–110.
[22] M. Bäker, Finite element simulation of high-speed cutting forces, Journal of Materials Processing Technology 176 (1) (2006) 117–126.