ABAQUS

Appendix 3

Writing a VUMAT

Overview

- Introduction
- Motivation
- Steps Required in Writing a VUMAT
- VUMAT Interface
- Example: VUMAT for Kinematic Hardening Plasticity

Introduction

Introduction

– ABAQUS/Explicit has an interface that allows you to implement general constitutive equations.

- In ABAQUS/Explicit the user-defined material model is implemented in user subroutine **`VUMAT`**.

– Use **`VUMAT`** when none of the existing material models included in the ABAQUS/Explicit material library accurately represents the behavior of the material to be modeled.

Introduction

– These interfaces make it possible to define any (proprietary) constitutive model of arbitrary complexity.

– User-defined material models can be used with any ABAQUS/Explicit structural element type.

– Multiple user materials can be implemented in a single **`VUMAT`** routine and can be used together.

– In this lecture the implementation of material models in **`VUMAT`** will be discussed and illustrated with an example.

Motivation

Motivation

- Proper testing of advanced constitutive models to simulate experimental results often requires complex finite element models.
  - Complex material modeling
- Special analysis problems occur if the constitutive model simulates material instabilities and localization phenomena.
- The material model developer should be concerned only with the development of the material model and not with the development and maintenance of the FE software.
  - Developments unrelated to material modeling
  - Porting problems with new systems
  - Long-term program maintenance of user-developed code

Steps Required in Writing a VUMAT

Steps Required in Writing a VUMAT

– Proper definition of the constitutive equation, which requires one of the following:
  - Explicit definition of stress (Cauchy stress for large-strain applications)
  - Definition of the stress rate only (in corotational framework)

– Furthermore, it is likely to require:
  - Definition of dependence on time, temperature, or field variables
  - Definition of internal state variables, either explicitly or in rate form

Steps Required in Writing a VUMAT

– Transformation of the constitutive rate equation into an incremental equation using a suitable integration procedure:
  - Forward Euler (explicit integration)
  - Backward Euler (implicit integration)
  - Midpoint method

Steps Required in Writing a VUMAT

– **This is the hard part!** Forward Euler (explicit) integration methods are simple but have a stability limit,

$$|\Delta\varepsilon| < \Delta\varepsilon_{stab},$$

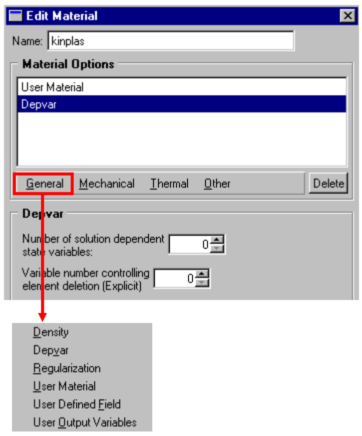
where $\Delta\varepsilon_{stab}$ is usually less than the elastic strain magnitude.

- For explicit integration the time increment must be controlled.
- For implicit or midpoint integration, the algorithm is more complicated and often requires local iteration. However, there is usually no stability limit.
- An incremental expression for the internal state variables must also be obtained.

Steps Required in Writing a VUMAT

– Coding the **VUMAT**.

- Follow FORTRAN 77 or C conventions.
- Make sure that the code can be vectorized.
- Make sure that all variables are defined and initialized properly.
- Use ABAQUS utility routines as required.
- Assign enough storage space for state variables with the *DEPVAR option.



Steps Required in Writing a VUMAT

– Verifying the **VUMAT** with a small (one element) input file.

1. Run tests with all displacements prescribed to verify the integration algorithm for stresses and state variables. Suggested tests include:
   – Uniaxial
   – Uniaxial in oblique direction
   – Uniaxial with finite rotation
   – Finite shear
2. Compare test results with analytical solutions or standard ABAQUS material models, if possible. If the above verification is successful, apply to more complicated problems.

ABAQUS

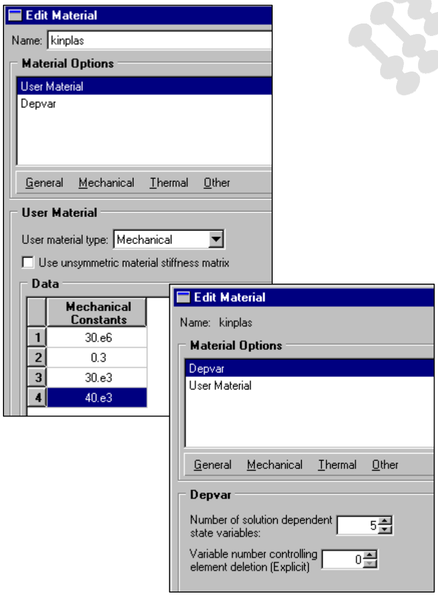
VUMAT Interface

VUMAT Interface

– The following acts as the interface to a **VUMAT** in which kinematic hardening plasticity is defined.

```
*MATERIAL, NAME=KINPLAS
*USER MATERIAL, CONSTANTS=4
30.E6, 0.3, 30.E3, 40.E3
*DEPVAR
5
*INITIAL CONDITIONS,
  TYPE=SOLUTION
```

Data line to specify initial solution-dependent variables

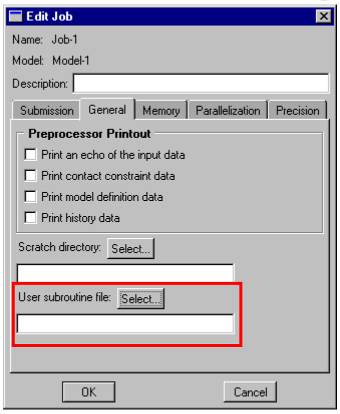


VUMAT Interface

– The user subroutine, which must be kept in a separate file, is invoked with the ABAQUS execution procedure, as follows:

```
abaqus job=... user=....
```

– The user subroutine must be invoked in a restarted analysis because user subroutines are not saved on the restart file.



VUMAT Interface

– Additional notes:

- Solution-dependent state variables can be output with identifiers SDV1, SDV2, etc. Contour, path, and X–Y plots of SDVs can be plotted in ABAQUS/Viewer.
- Include only a single **VUMAT** subroutine in the analysis. If more than one material must be defined, test on the material name in the **VUMAT** routine and branch.

VUMAT Interface

– The **VUMAT** subroutine header is shown below:

```
      subroutine vumat(
c Read only (unmodifiable) variables-
     1   nblock, ndir, nshr, nstatev, nfieldv, nprops, lanneal,
     2   stepTime, totalTime, dt, cmname, coordMp, charLength,
     3   props, density, strainInc, relSpinInc,
     4   tempOld, stretchOld, defgradOld, fieldOld,
     5   stressOld, stateOld, enerInternOld, enerInelasOld,
     6   tempNew, stretchNew, defgradNew, fieldNew,
c write only (modifiable) variables -
     7   stressNew, stateNew, enerInternNew, enerInelasNew)
c
      include 'vaba_param.inc'
c
```

VUMAT Interface

```
      dimension props(nprops), density(nblock), coordMp(nblock),
     1  charLength(nblock), strainInc(nblock, ndir+nshr),
     2  relSpinInc(nblock, nshr), tempOld(nblock),
     3  stretchOld(nblock, ndir+nshr),defgradOld(nblock,ndir+nshr+nshr),
     4  fieldOld(nblock, nfieldv), stressOld(nblock, ndir+nshr),
     5  stateOld(nblock, nstatev), enerInternOld(nblock),
     6  enerInelasOld(nblock), tempNew(nblock),
     7  stretchNew(nblock, ndir+nshr),defgradNew(nblock,ndir+nshr+nshr),
     8  fieldNew(nblock, nfieldv), stressNew(nblock,ndir+nshr),
     9  stateNew(nblock, nstatev), enerInternNew(nblock),
     1  enerInelasNew(nblock)
c
      character*80 cmname
```


VUMAT Interface

- **VUMAT variables**
  - The following quantities are available in **VUMAT**, but they cannot be redefined:
    - Stress, stretch, and SDVs at the start of the increment
    - Relative rotation vector and deformation gradient at the start and end of an increment and strain increment
    - Total and incremental values of time, temperature, and user-defined field variables at the start and end of an increment
    - Material constants, density, material point position, and a characteristic element length
    - Internal and dissipated energies at the beginning of the increment
    - Number of material points to be processed in a call to the routine (**NBLOCK**)
    - A flag indicating whether the routine is being called during an annealing process

VUMAT Interface

- The following quantities must be defined:
  - Stress and SDVs at the end of an increment
- The following variables may be defined:
  - Internal and dissipated energies at the end of the increment
- Many of these variables are equivalent or similar to those in **`UMAT`**.
- Complete descriptions of all parameters are provided in the **`VUMAT`** section in Chapter 25 of the ABAQUS Analysis User's Manual.

VUMAT Interface

The header is usually followed by dimensioning of local arrays. It is good practice to define constants via parameters and to include comments.

```
 parameter( zero = 0.d0, one = 1.d0, two = 2.d0, three = 3.d0,
1   third = one/three, half = 0.5d0, two_thirds = two/three,
2   three_halfs = 1.5d0)
```

The **`parameter`** assignments yield accurate floating point constant definitions on any platform.

VUMAT Interface

- **VUMAT conventions**
  - Stresses and strains are stored as vectors.
    - For plane stress elements: σ_{11}, σ_{22}, σ_{12}.
    - For plane strain and axisymmetric elements: σ_{11}, σ_{22}, σ_{33}, σ_{12}.
    - For three-dimensional elements: σ_{11}, σ_{22}, σ_{33}, $\sigma_{12,}$, σ_{23}, σ_{31}.
    - **For three-dimensional elements, this storage scheme is different than that for ABAQUS/Standard.**
  - The shear strain is stored as **tensor** shear strains,

$$\varepsilon_{12} = \frac{1}{2}\gamma_{12}.$$

VUMAT Interface

- The deformation gradient is stored similar to the way in which symmetric tensors are stored.
  - For plane stress elements: F_{11}, F_{22}, F_{12}, F_{21}.
  - For plane strain and axisymmetric elements: F_{11}, F_{22}, F_{33}, F_{12}, F_{21}.
  - For three-dimensional elements: F_{11}, F_{22}, F_{33}, F_{12}, F_{23}, F_{31}, F_{21}, F_{32}, F_{13}.

VUMAT Interface

- **VUMAT formulation aspects**
  - **Vectorized interface**
    - In `VUMAT` the data are passed in and out in large blocks (dimension `nblock`). `nblock` typically is equal to 64 or 128.
      - Each entry in an array of length `nblock` corresponds to a single material point. All material points in the same block have the same material name and belong to the same element type.
    - This structure allows vectorization of the routine.
      - A vectorized `VUMAT` should make sure that all operations are done in vector mode with `nblock` the vector length.
    - In vectorized code branching inside loops should be avoided.
      - Element type based branching should be outside the `nblock` loop.

VUMAT Interface

- **Corotational formulation**
  - The constitutive equation is formulated in a corotational framework, based on the **Green-Naghdi** rate.
    - The incremental rotation is obtained from the total rotation $\boldsymbol{F} = \boldsymbol{R} \cdot \boldsymbol{U}$ with the expression $\Delta\boldsymbol{\Omega} = \Delta\boldsymbol{R} \cdot \boldsymbol{R}^T$.
    - The strain increment is obtained with Hughes-Winget.
    - Other measures can be obtained from the deformation gradient.
    - The relative spin increment $\Delta\boldsymbol{\omega} - \Delta\boldsymbol{\Omega}$ is also provided.
      - The quantity $\Delta\boldsymbol{\omega}$ corresponds to the **Jaumann** rate. In ABAQUS it is used in certain instances: e.g., solid elements using the built-in linear elastic and plastic material models.

VUMAT Interface

- The user must define the Cauchy stress: this stress reappears during the next increment as the "old" stress.
- There is no need to rotate tensor state variables.
  - A rotation is needed, however, if a rate other than the Green-Naghdi rate is desired.
  - For example, to use the Jaumann rate, evaluate the expression defined by Hughes and Winget for the rotation increment using the relative spin increment:

$$\Delta \boldsymbol{R} = \left[\boldsymbol{I} - \frac{1}{2}(\Delta\boldsymbol{\omega} - \Delta\boldsymbol{\Omega}) \right]^{-1} \cdot \left[\boldsymbol{I} + \frac{1}{2}(\Delta\boldsymbol{\omega} - \Delta\boldsymbol{\Omega}) \right]$$

Then, rotate all "old" tensor quantities before performing constitutive updates. For example, for the stress tensor:

`stressOld`

$$\texttt{stressNew} \longrightarrow \boldsymbol{\sigma}_{t+\Delta t} = \Delta \boldsymbol{R} \cdot \boldsymbol{\sigma}_t \cdot \Delta \boldsymbol{R}^T + \Delta \boldsymbol{\sigma}$$

VUMAT Interface

– **VUMATs and hyperelasticity**

- Hyperelastic constitutive equations relate the Cauchy stress $\boldsymbol{\sigma}$ to the deformation gradient $\boldsymbol{F}$ through the left Cauchy-Green deformation tensor $\boldsymbol{B}$.
- Using $\boldsymbol{F}$ for hyperelastic constitutive models in a **VUMAT** presents some difficulties, however, because…
  - ABAQUS/Explicit uses a corotational system which automatically accounts for rigid body rotations.
  - The deformation gradient that is passed into the **VUMAT** is referred to a fixed basis associated with the original configuration.
    - It also incorporates the rotations—recall the deformation gradient can be written as $\boldsymbol{F} = \boldsymbol{RU}$, where $\boldsymbol{R}$ is the rotation tensor and $\boldsymbol{U}$ is the stretch tensor.

VUMAT Interface

- Thus, to avoid including the effects of the rotations twice, hyperelastic constitutive models implemented in a **VUMAT** should be formulated in terms of the stretch tensor $\boldsymbol{U}$.
  - This allows you to obtain the corotational Cauchy stress directly.
  - For example, for neo-Hookean hyperelasticity:

$$\boldsymbol{\sigma} = \frac{2}{J} C_{10} \left(\bar{\boldsymbol{B}} - \frac{1}{3} \mathrm{tr}(\bar{\boldsymbol{B}}) \boldsymbol{I} \right) + \frac{2}{D_1} (J-1) \boldsymbol{I}, \quad \bar{\boldsymbol{B}} = \boldsymbol{B} / J^{2/3}.$$

  - Substituting $\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U}$ into the above expressions yields:

$$\boldsymbol{\sigma} = \boldsymbol{R} \left\{ \frac{2}{J} C_{10} \left(\bar{\boldsymbol{U}}^2 - \frac{1}{3} \mathrm{tr}(\bar{\boldsymbol{U}}^2) \boldsymbol{I} \right) + \frac{2}{D_1} (J-1) \boldsymbol{I} \right\} \boldsymbol{R}^T, \text{ where } \bar{\boldsymbol{U}} = \boldsymbol{U} / J^{1/3}.$$

  - The corotational stress is the quantity contained within the curly brackets:

$$\boldsymbol{\sigma}^{\mathrm{corot}} = \frac{2}{J} C_{10} \left(\bar{\boldsymbol{U}}^2 - \frac{1}{3} \mathrm{tr}(\bar{\boldsymbol{U}}^2) \boldsymbol{I} \right) + \frac{2}{D_1} (J-1) \boldsymbol{I}.$$

ABAQUS

Example: VUMAT for Kinematic Hardening Plasticity

Example: VUMAT for Kinematic Hardening Plasticity

- **Governing equations**
  - Elasticity:

$$\sigma_{ij} = \lambda \delta_{ij} \varepsilon^{el}_{kk} + 2\mu \varepsilon^{el}_{ij},$$

or in a Jaumann (corotational) rate form:

$$\dot{\sigma}^{J}_{ij} = \lambda \delta_{ij} \dot{\varepsilon}^{el}_{kk} + s\mu \dot{\varepsilon}^{el}_{ij}.$$

- The Jaumann rate equation is integrated in a corotational framework:

$$\Delta\sigma^{J}_{ij} = \lambda \delta_{ij} \Delta\varepsilon^{el}_{kk} + 2\mu \Delta\varepsilon^{el}_{ij}.$$

Example: VUMAT for Kinematic Hardening Plasticity

- Plasticity:
  - Yield function: $\sqrt{\frac{3}{2}\left(S_{ij} - \alpha_{ij}\right)\left(S_{ij} - \alpha_{ij}\right)} - \sigma_y = 0.$
  - Equivalent plastic strain rate: $\dot{\bar{\varepsilon}}^{pl} = \sqrt{\frac{2}{3}\dot{\varepsilon}^{pl}_{ij}\dot{\varepsilon}^{pl}_{ij}}.$
  - Plastic flow law: $\dot{\varepsilon}^{pl}_{ij} = \frac{3}{2}\left(S_{ij} - \alpha_{ij}\right)\dot{\bar{\varepsilon}}^{pl}/\sigma_y.$
  - Prager-Ziegler (linear) kinematic hardening: $\dot{\alpha}_{ij} = \frac{2}{3} h \dot{\varepsilon}^{pl}_{ij}.$

Example: VUMAT for Kinematic Hardening Plasticity

- **Integration procedure**
  - We first calculate the equivalent stress based on purely elastic behavior (elastic predictor),

$$\bar{\sigma}^{pr} = \sqrt{\frac{3}{2}\left(S_{ij}^{pr} - \alpha_{ij}^{o}\right)\left(S_{ij}^{pr} - \alpha_{ij}^{0}\right)}, \qquad S_{ij}^{pr} = S_{ij}^{o} + 2\mu\Delta e_{ij}.$$

  - Plastic flow occurs if the elastic predictor is larger than the yield stress. The backward Euler method is used to integrate the equations,

$$\Delta\varepsilon_{ij}^{pl} = \frac{3}{2}\left(S_{ij}^{pr} - \alpha_{ij}^{o}\right)\Delta\bar{\varepsilon}^{pl} / \bar{\sigma}^{pr}.$$

  - After some manipulation we obtain a closed form expression for the equivalent plastic strain increment,

$$\Delta\bar{\varepsilon}^{pl} = \left(\bar{\sigma}^{pr} - \sigma_y\right) / \left(h + 3\mu\right).$$

Example: VUMAT for Kinematic Hardening Plasticity

– This leads to the following update equations for the shift tensor, the stress, and the plastic strain:

$$\Delta\alpha_{ij} = \eta_{ij} h \Delta\bar{\varepsilon}^{pl}, \quad \Delta\varepsilon_{ij}^{pl} = \frac{3}{2}\eta_{ij}\Delta\bar{\varepsilon}^{pl},$$

$$\sigma_{ij} = \alpha_{ij}^{o} + \Delta\alpha_{ij} + \eta_{ij}\sigma_y + \frac{1}{3}\delta_{ij}\sigma_{kk}^{pr}, \quad \eta_{ij} = \left(S_{ij}^{pr} - \alpha_{ij}^{o}\right) / \bar{\sigma}^{pr}.$$

– The integration procedure for kinematic hardening is described in the ABAQUS Analysis User's Manual.

– The appropriate coding is shown on the following pages.

Example: VUMAT for Kinematic Hardening Plasticity

- Coding for kinematic hardening plasticity VUMAT

```
c     J2 Mises plasticity with kinematic hardening for plane strain case.
c     The state variables are stored as:
c
c          state(*, 1) = back stress component 11
c          state(*, 2) = back stress component 22
c          state(*, 3) = back stress component 33
c          state(*, 4) = back stress component 12
c          state(*, 5) = equivalent plastic strain
c
      e      = props(1)
      xnu    = props(2)
      yield  = props(3)
      hard   = props(4)
c
c     elastic constants
c
      twomu  = e / ( one + xnu )
      thremu = three_halfs * twomu
      sixmu  = three * twomu
      alamda = twomu * ( e - twomu ) / ( sixmu - two * e )
      term   = one / ( twomu * ( one + hard/thremu ) )
      con1   = sqrt( two_thirds )
```


Example: VUMAT for Kinematic Hardening Plasticity

```
c
c If stepTime equals to zero, assume the material pure elastic and use
c initial elastic modulus
c
      if( stepTime .eq. zero ) then
c
      do i = 1, nblock
C
C  Trial Stress
         trace = strainInc (i, 1) + strainInc (i, 2) + strainInc (i, 3)
         stressNew(i, 1)=stressOld(i, 1) + alamda*trace
     1   +               twomu*strainInc(i,1)
         stressNew(i, 2)=stressOld(i, 2) + alamda*trace
     1   +               twomu*strainInc(i, 2)
         stressNew(i, 3)=stressOld(i, 3) + alamda*trace
     1   +               twomu*strainInc(i,3)
         stressNew(i, 4)=stressOld(i, 4)
     1   +               twomu*strainInc(i, 4)
      end do
c
      else
```

Example: VUMAT for Kinematic Hardening Plasticity

```
C
C    Plasticity calculations in block form
C
      do i = 1, nblock
C  Elastic predictor stress
        trace = strainInc(i, 1) + strainInc(i, 2) + strainInc(i, 3)
        sig1= stressOld(i, 1) + alamda*trace + twomu*strainInc(i, 1)
        sig2= stressOld(i, 2) + alamda*trace + twomu*strainInc(i, 2)
        sig3= stressOld(i, 3) + alamda*trace + twomu*strainInc(i, 3)
        sig4= stressOld(i, 4)                + twomu*strainInc(i, 4)
C  Elastic predictor stress measured from the back stress
        s1 = sig1 - stateOld(i, 1)
        s2 = sig2 - stateOld(i, 2)
        s3 = sig3 - stateOld(i, 3)
        s4 = sig4 - stateOld(i, 4)
C  Deviatoric part of predictor stress measured from the back stress
        smean = third * ( s1 + s2 + s3 )
        ds1 = s1 - smean
        ds2 = s2 - smean
        ds3 = s3 - smean
C  Magnitude of the deviatoric predictor stress difference
        dsmag = sqrt( ds1**2 + ds2**2 + ds3**2 + two*s4**2 )
```


Example: VUMAT for Kinematic Hardening Plasticity

```
c
c  Check for yield by determining the factor for plasticity, zero for
c  elastic, one for yield
c
        radius = con1 * yield
        facyld = zero
        if(  dsmag - radius .ge. zero ) facyld = one
c
c  Add a protective addition factor to prevent a divide by zero when DSMAG
c  is zero. If DSMAG is zero, we will not have exceeded the yield stress
c  and FACYLD will be zero.
c
        dsmag  = dsmag + ( one - facyld )
c
c  Calculated increment in gamma ( this explicitly includes the time step)
c
        diff   = dsmag - radius
        dgamma = facyld * term * diff
```

Example: VUMAT for Kinematic Hardening Plasticity

```
c
c  Update equivalent plastic strain
c
        deqps  = con1 * dgamma
        stateNew(i, 5) = stateOld(i, 5) + deqps
c
c  Divide DGAMMA by DSMAG so that the deviatoric stresses are explicitly
c  converted to tensors of unit magnitude in the following calculations
c
        dgamma = dgamma / dsmag
c
c  Update back stress
c
        factor  = hard * dgamma * two_thirds
        stateNew(i, 1) = stateOld(i, 1) + factor * ds1
        stateNew(i, 2) = stateOld(i, 2) + factor * ds2
        stateNew(i, 3) = stateOld(i, 3) + factor * ds3
        stateNew(i, 4) = stateOld(i, 4) + factor *  s4
```


Example: VUMAT for Kinematic Hardening Plasticity

```
c
c  Update stress
c
        factor   = twomu * dgamma
        stressNew(i, 1) = sig1 - factor * ds1
        stressNew(i, 2) = sig2 - factor * ds2
        stressNew(i, 3) = sig3 - factor * ds3
        stressNew(i, 4) = sig4 - factor *  s4
c
c  Update the specific internal energy -
c
        stressPower = half * (
     1         ( stressOld(i, 1)+stressNew(i, 1) )*strainInc(i, 1)
     2   +     ( stressOld(i, 2)+stressNew(i, 2) )*strainInc(i, 2)
     3   +     ( stressOld(i, 3)+stressNew(i, 3) )*strainInc(i, 3)
     4   + two*( stressOld(i, 4)+stressNew(i, 4) )*strainInc(i, 4) )
        enerInternNew(i) = enerInternOld(i)
     1   +                    stressPower/density(i)
```

Example: VUMAT for Kinematic Hardening Plasticity

```
c
c  Update the dissipated inelastic specific energy -
c
         smean = third* (stressNew(i, 1)+stressNew(i, 2)
     1   +                stressNew(i, 3))
         equivStress = sqrt( three_halfs
     1   *                 ( (stressNew(i, 1)-smean)**2
     2   +                   (stressNew(i, 2)-smean)**2
     3   +                   (stressNew(i, 3)-smean)**2
     4   +              two * stressNew(i, 4)**2 ) )
c
         plasticWorkInc = equivStress * deqps
         enerInelasNew(i) = enerInelasOld(i)
     1   +                   plasticWorkInc / density(i)
c
      end do
c
      end if
      return
      end
```


Example: VUMAT for Kinematic Hardening Plasticity

- **Remarks**
  - In the **datacheck** phase, **VUMAT** is called with a set of fictitious strains and a **TOTALTIME** and **STEPTIME** both equal to 0.0.
    - A check is done on the user's constitutive relation, and an initial stable time increment is determined based on calculated equivalent initial material properties.
    - You should ensure that elastic properties are used in this call to **VUMAT**; otherwise, too large an initial time increment may be used, leading to instability.
    - A warning message is printed to the status (**.sta**) file, informing the user that this check is being performed.

Example: VUMAT for Kinematic Hardening Plasticity

– Special coding techniques are used to obtain vectorized coding.
  - All small loops inside the material routine are “unrolled.”
  - The same code is executed regardless of whether the behavior is purely elastic or elastic plastic.
– Special care must be taken to avoid divides by zero.
  - No external subroutines are called inside the loop.
  - The use of local scalar variables inside the loop is allowed.
  - The compiler will automatically expand these local scalar variables to local vectors.
  - Iterations should be avoided.
– If iterations cannot be avoided, use a fixed number of iterations and do not test on convergence.