Implementation of material constitutive equations in finite element codes

J. Besson
Centre des Matériaux, École des Mines de Paris,
UMR CNRS 7633, BP 87 Evry cedex 91003
Introduction

Why is it important to know how to implement material constitutive equations in FE codes?

- Few constitutive equations are available in commercial codes
- Implement new constitutive equations in FE codes (ABAQUS, ANSYS, MARC, …, ASTER, CAST3M, …, Zébulon, WARP3D)
- Understand convergence problems
Outline

- Definition of a constitutive equation (FE code point of view)
- Numerical integration methods (explicites/implicites)
- Consistent tangent matrix
- Particular case: von Mises material
- Convergence
Definition of a constitutive equation

- For a displacement based FE formulation, nodal displacements are assumed to be known and therefore the deformations

- The constitutive equation must then supply: (i) stresses $\sigma$ and (ii) the consistent tangent matrix $L = \frac{\partial \Delta \sigma}{\partial \Delta \varepsilon}$ for a given strain increment $\Delta \varepsilon$.

- Complex constitutive equations are characterized by internal state variables $[A]$: the constitutive equation must provide an update of these variables consistent with the strain and time increment.
Role of the constitutive equation in the FEM

\[ [U](t_0) \text{ known} \]

- iteration \( i \)
- \( \Delta [U]_i \)
- \( i = i + 1 \)
- compute \( \delta \Delta [U] \) using \([R]\) and \([K]\)
- evaluate \( \varepsilon(t_1), \Delta \varepsilon \) for each element
- evaluate \([R]\) is \([R]\) small enough ?
- no
- compute \([F]\) using \(\varepsilon(t_1)\)
- compute \([K]\) using \(L\)
- obtain \(\sigma(t_1), \quad L \approx \frac{\partial \Delta \sigma}{\partial \Delta \varepsilon} \)
- for each element

- yes
- next increment

- box: global computation
- box: local time integration of the constitutive equations
Generic interface behavior/FEM

\[ \Delta t = t_1 - t_0 \]

- INPUT

  \[ [A](t_0) \]
  \[ \varepsilon(t_1) \]
  \[ \Delta \varepsilon \]

- OUTPUT

  \[ [A](t_1) \]
  \[ \varphi(t_1) \]
  \[ L \]

material behavior
Quantities characterising the material behavior

- Integrated variables/State variables ($V_{\text{int}}$)
- Auxiliary variables ($V_{\text{aux}}$)
- External parameters ($EP$)
- Coefficients ($CO$)
  \[ CO = CO(EP, V_{\text{int}}, V_{\text{aux}}) \]
- Interface: input variable ($\text{primal}$), associated dual variable ($\text{dual}$), tangent matrix $\partial \Delta \text{dual} / \partial \Delta \text{primal}$.
Examples of primal—dual couples

<table>
<thead>
<tr>
<th>problème</th>
<th>primal</th>
<th>dual</th>
</tr>
</thead>
<tbody>
<tr>
<td>mechanics (small deformation)</td>
<td>$\varepsilon$</td>
<td>$\sigma$</td>
</tr>
<tr>
<td>mechanics (finite strain)</td>
<td>$\mathbf{F}$</td>
<td>$\sigma$ ou $\mathbf{S}$</td>
</tr>
<tr>
<td>thermal problem</td>
<td>$(T, \text{grad}T)$</td>
<td>$(H, q)$</td>
</tr>
</tbody>
</table>
Constitutive equations as a differential equation

\[
\frac{d[A]}{dt} = [\dot{A}] = [G](A, t)
\]

\[
\frac{A_i}{dt} = [\dot{A}]_i = G_i(A_1, \ldots, A_n, t)
\]

- Time \((t)\) represents the imposed deformation but also an external parameter such as the temperature \((T(\vec{x}, t))\).
- The FE evaluation of the Constitutive equation for \((\Delta \xi, \Delta)\) corresponds to the integration of the previous equation from \(t_0\) to \(t_1\).
- In most cases: \([A] = (\xi_e, \ldots)\) so that \(\varphi(t_1) = E(t_1) : \xi_e(t_1)\).
- \(L\) must be computed ...
Integration methods of Constitutive equations

**Euler explicit method**

\[
[A](t_1) = [A](t_0) + [\dot{A}][A](t_0), t_0) \Delta t = [A](t_0) + [G][A](t_0), t_0) \Delta t
\]

- The method is not stable and should be avoided
- Explicit: because \([\dot{A}]\) is computed at \(t_0\) for the known couple \((A(t_0), t_0)\)
Runge–Kutta explicit method

- Numerical estimation of the derivatives of $[\dot{A}]$ (i.e. $d^2 [F]_A / dt^2$, $d^3 [F]_A / dt^3$, …)
- Error estimation to control the solution

The Runge–Kutta Integration method is easy to implement because it only uses the differential equation $[\dot{A}] = [G] ([A], t)$. It however has some drawbacks:

- Integration may require a large CPU time
- In the case of plastic materials, it is mandatory to compute the plastic multiplier which can be a difficult task (see below) in the case of temperature dependant material coefficients.
Runge–Kutta Method

Using a Taylor expansion, one gets for a time increment \([t, t + \Delta t]\) (which can differ from the FE time step \([t_0, t_1]\):

\[
\{v\} (t + \Delta t) = \{v\} (t) + \{\dot{v}\} (t) \Delta t + O(\Delta t^2)
\]

The accuracy of the Euler integration is therefore of magnitude \(O(\Delta t^2)\). Based on this first estimation of the increment, another one can be performed using the mid-point (i.e. \(t + \Delta t/2\)). Let:

\[
\{\delta v_1\} = \Delta t \{\dot{v}\} (t)
\]

and

\[
\{\delta v_2\} = \Delta t \{\dot{v}\} \left( t + \frac{\Delta t}{2}, \{v\} (t) + \frac{1}{2} \{\delta v_1\} \right)
\]

\[
= \Delta t \left( \{\dot{v}\} (t) + \frac{\Delta t}{2} \{\ddot{v}\} (t) \right)
\]

\[
= \{\delta v_1\} + \frac{\Delta t^2}{2} \{\ddot{v}\} (t)
\]
This provides one way to estimate \( \{ \ddot{v} \} (t) \). The second order Taylor expansion is:

\[
\{ v \} (t + \Delta t) = \{ v \} (t) + \{ \ddot{v} \} (t) \Delta t + \{ \dddot{v} \} (t) \frac{\Delta t^2}{2} + O(\Delta t^3)
\]

which can be simplified using the previous estimate of \( \{ \dddot{v} \} (t) \):

\[
\{ v \} (t + \Delta t) = \{ v \} (t) + \{ \dddot{v} \} (t) + O(\Delta t^3)
\]

The precision has been improved \( O(\Delta t^3) \) instead of \( O(\Delta t^2) \). This is a second order method.
The procedure can be generalized. This leads to a 4th order Runge–Kutta method which is written as:

\[
\begin{align*}
\{\delta v_1\} &= \Delta t \{\dot{v}\} (t, \{v\}) \\
\{\delta v_2\} &= \Delta t \{\dot{v}\} \left( t + \frac{\Delta t}{2}, \{v\} + \frac{1}{2} \{\delta v_1\} \right) \\
\{\delta v_3\} &= \Delta t \{\dot{v}\} \left( t + \frac{\Delta t}{2}, \{v\} + \frac{1}{2} \{\delta v_2\} \right) \\
\{\delta v_4\} &= \Delta t \{\dot{v}\} (t + \Delta t, \{v\} + \{\delta v_3\}) \\
\{v\} (t + \Delta t) &= \{v\} (t) + \frac{1}{6} \{\delta v_1\} + \frac{1}{3} \{\delta v_2\} + \frac{1}{3} \{\delta v_3\} + \frac{1}{6} \{\delta v_4\} + O(\Delta t^5)
\end{align*}
\]
Runge–Kutta Method: error control

Aim: Obtain a given precision while minimize the computational effort

Make large time steps when the \( \{ \dot{v} \} \) function varies little and smaller time steps if its evolution is rapid.

Let \( \Delta t \) be the time increment over which the integration has to be performed. It can be divided into \( n \) sub-steps so that:

\[
\Delta t = \sum_{k} \delta t_k
\]

The error is estimated by applying the RK4 method

- one time step \( \delta t \rightarrow \{ v_1 \} \)
- two time steps \( \delta t/2 \rightarrow \{ v_2 \} \)

This corresponds to 11 evaluations of \( \{ \dot{v} \} \).
Let \( \{v\} (t + \delta t) \) be the exact solution; one gets

\[
\{v\} (t + \delta t) = \{v_1\} + (\delta t)^5 \{\phi\} + O(\delta t^6)
\]

\[
\{v\} (t + \delta t) = \{v_2\} + 2(\delta t/2)^5 \{\phi\} + O(\delta t^6)
\]

\[
\{\phi\} \approx \text{constant} \approx \frac{1}{5!} \{v^{(5)}\}
\]

The difference between both estimations is an indicator the error:

\[
\{E\} = \{v_2\} - \{v_1\}
\]

This difference has to be kept smaller that a prescribed precision by adjusting \( \delta t \). This equation can be solved neglecting \( O(\delta t^6) \) terms:

\[
\{v\} (t + \delta t) = \{v_2\} + \frac{1}{15} \{E\} + O(\delta t^6)
\]

This is a better estimation (5th order).
\{E\} can then be used to modify the time step. Let \(\{E^0\}\) by the requested precision (note that the precision is a vector).

\[
\text{if } E_i < E^0_i, \forall i
\]

The time step can be increased

\[
\text{if } \exists i, E_i > E^0_i
\]

The time step must be decreased

The time step is corrected by the following factor:

\[
\alpha = \min_i \left| \frac{E^0_i}{E_i} \right|^{0.2}
\]

as the error varies as \(\delta t^5\) for the 4th order Runge–Kutta method.
\{E^0\} must now be chosen.

The required precision must be obtained over the whole time increment $\Delta t$ and not only on local sub steps $\delta t_k$. In that case the error is best defined as:

$$E^0_i = \epsilon \delta t \left| \frac{dv_i}{dt} \right| = \epsilon |\delta v_i|$$
**Implicite methods (or θ-methods)**

- Evaluate $\dot{A}$ at $t_\theta$ between $t_0$ et $t_1$

- \[ t_\theta = t_0 + \theta \Delta t \] with \(0 \leq \theta \leq 1\)

- Two solutions:
  \[
  \Delta [A] = G([A](t_0) + \theta \Delta [A], t_0 + \theta \Delta t) \Delta t
  \]
  \[
  = G([A]_\theta, t_\theta) \Delta t
  \]

- Implicit: $\Delta [A]$ appears on both left and right handsides of the previous equations

- Integrate the constitutive equation = solve the implicit equations

- $\theta = 0 \rightarrow$ Euler

- ... in the following the first method will only be considered
Solution obtained by the Newton-Raphson method

Write a residual vector

\[
[R] (\Delta [A]) = \Delta [A] - [G] ([A] (t_0) + \theta \Delta [A], t_0 + \theta \Delta t) \Delta t
\]

\[
R_i(\Delta A_1, \ldots, \Delta A_n) = \Delta A_i - G_i(A_1(t_0) + \theta \Delta A_1, \ldots, A_n(t_0) + \theta \Delta A_n) \Delta t
\]

1st order Taylor expansion around an estimation \(\Delta [A]_s\):

\[
[R] = [R] (\Delta [A]_s) + \frac{\partial [R]}{\partial \Delta [A]} (\Delta [A] - \Delta [A]_s) = [0]
\]

Construction of the next estimation:

\[
\Delta [A]_{s+1} = \Delta [A]_s - \left( \frac{\partial [R]}{\partial \Delta [A]} \right)^{-1}_{\Delta [A]=\Delta [A]_s} \cdot [R] (\Delta [A]_s)
\]

\[
[J] = \frac{\partial [R]}{\partial \Delta [A]} (J_{ij} = \frac{\partial R_j}{\partial A_i}): \text{Jacobian matrix, } [J]^* = [J]^{-1}
\]
Note

- The internal variable vector $[A]$ often contains 2nd order tensors.
- The Voigt notation is used.

\[
\begin{align*}
\varepsilon &= \begin{pmatrix}
\varepsilon_{11} \\
\varepsilon_{22} \\
\varepsilon_{33} \\
2\varepsilon_{12} \\
2\varepsilon_{23} \\
2\varepsilon_{31}
\end{pmatrix}, \\
\sigma &= \begin{pmatrix}
\sigma_{11} \\
\sigma_{22} \\
\sigma_{33} \\
\sigma_{12} \\
\sigma_{23} \\
\sigma_{31}
\end{pmatrix}, \\
x &= \begin{pmatrix}
x_{11} \\
x_{22} \\
x_{33} \\
\sqrt{2}x_{12} \\
\sqrt{2}x_{23} \\
\sqrt{2}x_{31}
\end{pmatrix}
\end{align*}
\]
Integration — Simple Example: von Mises plasticity

- Additive Decomposition of deformations:
  \[ \vec{\varepsilon} = \vec{\varepsilon}_e + \vec{\varepsilon}_p \]

- Flow surface
  \[ \phi = \sigma_{eq} - R(p) \]

- Normality
  \[ \dot{\vec{\varepsilon}}_p = \dot{p} \frac{\partial \phi}{\partial \sigma} = \frac{3}{2} \dot{p} \frac{\vec{s}}{\sigma_{eq}} = \dot{\vec{n}} \]

- Internal variables: \((\vec{\varepsilon}_e, p)\)
• $\dot{p}$ is computed using the constancy condition: $\dot{\phi} = 0$

$$\dot{\phi} = \frac{\partial \phi}{\partial \varepsilon} : \dot{\varepsilon} + \frac{\partial \phi}{\partial \dot{p}} \dot{p} = \varepsilon : \dot{\varepsilon} - H \dot{p}$$

• avec $\varepsilon = E : \varepsilon_e = E : (\varepsilon - \varepsilon_p) \rightarrow \dot{\varepsilon} = E : \dot{\varepsilon}_e = E : (\dot{\varepsilon} - \dot{\varepsilon}_p)$

• so that:

$$\dot{p} = \frac{n : E : \ddot{\varepsilon}}{n : E : n + H}$$
• Differential equations to be integrated:

\begin{align*}
\text{sur } \xi_e & \quad \dot{\xi}_e = \dot{\xi} - \dot{p}n \\
\text{sur } p & \quad \dot{p} = \frac{n : E : \dot{\xi}}{n : E : n + H}
\end{align*}

• Pay attention to the dependance on external parameters (temperature...)

• Ready for the Runge–Kutta integration
von Mises plasticity: implicit integration

\[
\dot{\xi}_e = \ddot{\xi} - \dot{p}n \quad \rightarrow \quad \Delta \xi_e = \Delta \varepsilon - \Delta p n
\]

\[
\dot{p} = \frac{n : E : \ddot{\xi}}{n : E : n + H} \quad \rightarrow \quad \Delta p = \frac{n : E : \Delta \varepsilon}{n : E : n + H}
\]

- Evaluation of \(n, E, H\) ? ... at time \(t_\theta = t_0 + \theta \Delta t\).

- Application:

\[
n = \frac{3}{2} \frac{s^\theta}{\sigma_{eq}} \quad \text{avec} \quad \sigma^\theta = E^\theta : \varepsilon_e^\theta \quad \varepsilon_e = \varepsilon_e^0 + \theta \Delta \varepsilon_e
\]

\[
E^\theta = E(T^\theta) = E(T^0 + \theta \Delta T)
\]

\[
H^\theta = H(p^\theta) = H(p^0 + \theta \Delta p)
\]
The equation

\[ \Delta p = \frac{n : E : \Delta \varepsilon}{n : E : n + H} \]

is exact but can be replaced by:

\[ \phi = \sigma_{eq} - R(p) = 0 \]

It is wrong if \( R \) depends on an external parameter (temperature, \ldots) as:

\[ \dot{p} = \frac{n : E : \dot{\varepsilon} - R_{,T} \dot{T}}{n : E : n + H} \]

The correct incremental equation is then:

\[ \Delta p = \frac{n^{\theta} : E^{\theta} : \Delta \varepsilon - R_{,T}^{\theta} \Delta T}{n^{\theta} : E^{\theta} : n^{\theta} + H^{\theta}} \]

This method should be avoided!
Residual vector

\[ R_e = \Delta \bar{\varepsilon}_e + \Delta p n^\theta - \Delta \varepsilon \]

\[ R_p = \phi = \sigma_{eq}^\theta - R(p^\theta) \]

\[ R = (R_e, R_p) \]
Jacobian matrix ... a though job

- The Jacobian matrix can be written by blocks:

\[
[J] = \begin{pmatrix}
\frac{\partial R_e}{\partial \Delta \varepsilon_e} & \frac{\partial R_e}{\partial \Delta p} \\
\frac{\partial \Delta \varepsilon_e}{\partial R_p} & \frac{\partial \Delta p}{\partial R_p}
\end{pmatrix}
\]
Computation of the blocks related to $R_e = \Delta \varepsilon_e + \Delta p \mathbf{n}^\theta - \Delta \varepsilon$

\[
\frac{\partial R_e}{\partial \Delta \varepsilon_e} = 1 + \Delta p \frac{\partial \mathbf{n}}{\partial \sigma} : \frac{\partial \sigma}{\partial \varepsilon_e} : \frac{\partial \varepsilon_e}{\partial \Delta \varepsilon_e}
\]

\[
\mathbf{N} = \frac{1}{\sigma_{eq}} \left( \frac{3}{2} - \mathbf{n} \otimes \mathbf{n} \right)
\]

\[
\Rightarrow \quad \frac{\partial R_e}{\partial \Delta \varepsilon_e} = 1 + \Delta p \mathbf{N}^\theta : \mathbf{E}^\theta
\]

\[
\frac{\partial R_e}{\partial \Delta p} = \mathbf{n}^\theta
\]
Computation of the blocks related to $R_p = \sigma_{eq}^\theta - R(p^\theta)$

\[
\frac{\partial R_p}{\partial \Delta \varepsilon_e} = \frac{\partial \sigma_{eq}}{\partial \varepsilon} : \frac{\partial \varepsilon}{\partial \varepsilon_e} : \frac{\partial \varepsilon_e}{\partial \Delta \varepsilon_e} = \theta \mathbf{n} : \mathbf{E}
\]

\[
\frac{\partial R_p}{\partial \Delta p} = - \frac{\partial R}{\partial p} \frac{\partial p}{\partial \Delta p} = -\theta H^\theta
\]
Tangent matrix vs. consistent tangent matrix

- Tangent matrix

\[ \dot{\tilde{\sigma}} = L_p : \ddot{\tilde{\xi}} \]

- Calculated as:

\[ \dot{\tilde{\sigma}} = E : (\dot{\tilde{\xi}} - \dot{\tilde{p}} n) \]

and

\[ \dot{\tilde{p}} = \frac{n : E : \ddot{\tilde{\xi}}}{n : E : n + H} \]

imply

\[ \tilde{L}_p = E - \frac{(E : n) \otimes (n : E)}{n : E : n + H} \]
• Consistent tangent matrix

\[
L \approx \frac{\partial \Delta \sigma}{\partial \Delta \varepsilon}
\]

\[
\Delta \sigma = E : (\Delta \varepsilon - \Delta p n)
\]

\[
\delta \Delta \sigma = E : (\delta \Delta \varepsilon - \delta \Delta p n - \Delta p \delta n)
\]

\[
\ldots
\]

it can be shown that

\[
L \approx L_p - \Delta p E : N : E + O(\Delta p^2)
\]
Automatic and generic computation of the consistent tangent matrix

- Internal variables and residuals can be expressed in a very general way as:

\[
[A] = (\xi_e, [a])
\]

\[
[R] = ([R]_e, [R]_a)
\]

\[
[R]_e = \Delta \xi_e + \Delta \xi_{\text{irr}} - \Delta \xi
\]

- Influence of a small variation of \(\Delta \xi\) on the internal variables \((\xi_e, [a])\)? (around the solution)

- \([R]\) must stay null

\[
\delta [R] = [0] = \delta \begin{pmatrix} \Delta \xi_e + \Delta \xi_{\text{irr}} \\ [R]_a \end{pmatrix} - \delta \begin{pmatrix} \Delta \xi \\ [0] \end{pmatrix}
\]

\[
\delta [R] = \frac{\partial [R]}{\partial [A]} \left( \delta \Delta \xi \right) - \begin{pmatrix} \delta \Delta \xi \\ [0] \end{pmatrix} = [J] . \delta A - \begin{pmatrix} \delta \Delta \xi \\ [0] \end{pmatrix}
\]
Consequently

$$\delta \Delta A = [J]^{-1} \cdot \begin{pmatrix} \delta \Delta \varepsilon \\ 0 \end{pmatrix}$$

- $[J]^{-1}$ can be divided in sub-blocks:

$$[J]^{-1} = [J]^* = \begin{pmatrix} [J]_{ee}^* & [J]_{ea}^* \\
[J]_{ae}^* & [J]_{aa}^* \end{pmatrix},$$

- One therefore gets:

$$\delta \Delta \varepsilon_e = [J]_{ee}^* \cdot \delta \Delta \varepsilon$$

- Using the Hooke law (elasticity):

$$\varepsilon(t_1) = \varepsilon(t_0) + \Delta \varepsilon = \varepsilon(t_1) : \varepsilon_e(t_1) = \varepsilon(t_1) : (\varepsilon_e(t_0) + \Delta \varepsilon_e)$$

so that:

$$\delta \Delta \varepsilon = \varepsilon(t_1) : \delta \Delta \varepsilon_e = \varepsilon(t_1) : [J]_{ee}^* : \delta \Delta \varepsilon$$
• The consistent tangent matrix is therefore given by:

\[ L = E(t_1) : J_{ee}^* \]

• In case where \( E \) depends on an internal variable (e.g. \( d=\text{damage} \)) then

\[ \Delta \sigma = \frac{\partial E}{\partial d} \delta \Delta d : \varepsilon_e + E(t_1) : \Delta \varepsilon_e \]

\[ L = \frac{\partial E}{\partial d} : (\varepsilon_e \otimes [J]^*_e) + E : J_{ee}^* \]
Explicit/Implicit

<table>
<thead>
<tr>
<th>Explicit</th>
<th>Implicit</th>
</tr>
</thead>
<tbody>
<tr>
<td>easy to implement</td>
<td>difficult to implement</td>
</tr>
<tr>
<td>slow</td>
<td>fast</td>
</tr>
<tr>
<td>( \mathbf{L} ) ?</td>
<td>direct computation of ( \mathbf{L} )</td>
</tr>
</tbody>
</table>

- \( \mathbf{L} \) can be evaluated by perturbation

\[
L_{ijkl} = \frac{\sigma_{ij}(\Delta \xi + \delta \xi \mu^{kl}) - \sigma_{ij}(\Delta \xi)}{\delta \xi}
\]
Choosing de $\theta$ in the case of plasticity

The hardening law is

$$R(p) = 300 + 100(1 - \exp(-200p)) \quad \text{MPa}$$

\[J(\sigma) = R \text{ à } \theta = 1 : +\]

\[J(\sigma) = R \text{ à } \theta = \frac{1}{2} : \square\]
Solution(s)

- Use $\theta = 1$
- Write the residual $R_p$ for $\theta = 1$

$$
R_e = \Delta \varepsilon_e + \Delta p_n^\theta - \Delta \varepsilon
$$
$$
R_p = \sigma_{\text{eq}}^1 - R(p^1)
$$

- Internal variables must be evaluated at both $t_\theta (R_e)$ and $t_1 = t + \Delta t (R_p)$!
Plasticity: variable temperature

It is assumed that the flow stress $R$ depends on temperature $T$. The consistency condition is expressed as:

$$\dot{f} = \mathbf{n} : \mathbf{\dot{\varepsilon}} - R_p \dot{p} - R_T \dot{T} = 0$$

and one gets

$$\dot{p} = \frac{\mathbf{n} : \mathbf{C} : \mathbf{\dot{\varepsilon}_t} - R_T \dot{T}}{R_p + \mathbf{n} : \mathbf{C} : \mathbf{n}}$$

using the previous example with:

$$R(p, T) = [300 + 100(1 - \exp(-200p))] [1 - T/200]$$

It is shown that omitting the $R_T \dot{T}$ term in the consistency condition leads to wrong results:
In the case where the elasticity coefficients also depend on temperature, this dependence must also be accounted for while writing the consistency condition. The plastic multiplier is then expressed as:

\[
\dot{p} = \frac{n : C : \dot{\xi}_t - T \dot{n} : C_{,T} : \epsilon_e - R_{,T} \dot{T}}{R_{,p} + n : C : n}
\]

It may become difficult to take into account the various possible dependancies when several external parameters are prescribed. This problems are avoided in the case of the \( \theta \)-method as in all case the yield condition is deirectly used \( f_{t+\Delta t} = 0 \) and not \( \dot{f} = 0 \).
Prandtl–Reuss law: creep

In the case of a viscous material, $\dot{\rho}$ is directly obtained from the creep law:

$$\dot{\rho} = \phi(\sigma, A_i)$$

$\theta$–method:

$$r_p = \Delta p - \Delta t \phi(J - R, \ldots) \theta = 0$$

The partial derivatives related to the computation of the Jacobian matrix are:

$$\frac{\partial r_p}{\partial \Delta \xi_e} = -\theta \Delta t \phi, \omega \mathbf{C} : \mathbf{n}$$

$$\frac{\partial r_p}{\partial \Delta p} = 1 + \theta \Delta t R_{,p} \phi, \omega$$

There are not longer problems related to the calculation of the consistency condition. A “creep law” can be used to mimic plasticity. For instance using a Norton law

$$\phi(\omega) = \left(\frac{\omega}{K}\right)^n$$

if $n$ is high enough, one gets

$$J - R \simeq K$$
• example with $n = 10$ and $K = 1, 10, 50$. When $K$ is small enough, the viscoplastic solution tends towards the plastic solution

![Graph showing deformation and stress for different values of K]
Multi–kinematic law: constitutive equations

- The law is expressed using the following state internal variables:

  \( \varepsilon_e \)  elastic strain tensor
  \( \dot{\alpha}_i \)  kinematic hardening variable tensors
  \( r \)  isotropic hardening variable

- The following variable is an auxiliary variable:

  \( p \)  cumulated plastic strain

Forces associated to the state variables are:

\[
\begin{align*}
\sigma & = \mathbf{C} : \varepsilon_e \\
\dot{X}_i & = \mathbf{C}_i : \dot{\alpha}_i \\
R & = cr
\end{align*}
\]
• The back-stress $\mathbf{\Sigma}$ is given by:

$$\mathbf{\Sigma} = \sum_i \mathbf{\Sigma}_i$$

The evolution laws are given by:

$$\dot{\mathbf{\Sigma}}_e + \dot{\mathbf{\Sigma}}_p = \dot{\mathbf{\Sigma}}_t$$

$$\dot{\mathbf{\alpha}}_i = \dot{\mathbf{\Sigma}}_p - \dot{\mathbf{p}} \mathbf{D}_i : \mathbf{\alpha}_i$$

$$\dot{\mathbf{r}} = \dot{\mathbf{p}} - \dot{\mathbf{p}} \mathbf{r}$$

• The plasticity criterion is given by:

$$f = \| \mathbf{\sigma} - \mathbf{X} \|_M - \sigma_y - R \geq 0$$

$\sigma_y$ is the size of the initial elastic domain. The norm $\| . \|_M$ is used to model plastic anisotropy:

$$\| \mathbf{a} \|_M = \left( \mathbf{a}_i \mathbf{M} : \mathbf{a}_i \right)^{\frac{1}{2}}$$

where $\mathbf{M}$ is a fourth order tensor such that $\mathbf{M} : \mathbf{1} = 0$. 

Multi–kinematic law
In the viscous case (studied in the following)

\[ \dot{p} = \phi(f, \ldots) \]

The flow direction (normality) is expressed as:

\[ n = \frac{\partial f}{\partial \tilde{\sigma}} = \frac{1}{\|\tilde{\sigma} - \tilde{X}\|_M} M : (\tilde{\sigma} - \tilde{X}) \]

To compute the Jacobian matrix, the following tensor is also needed:

\[ N = \frac{\partial n}{\partial \sigma} = \frac{\partial^2 f}{\partial \sigma^2} = \frac{1}{\|\tilde{\sigma} - \tilde{X}\|_M} \left( M - \frac{1}{\|\tilde{\sigma} - \tilde{X}\|^2_\tilde{M}} M : (\tilde{\sigma} - \tilde{X}) \otimes \tilde{M} : (\tilde{\sigma} - \tilde{X}) \right) \]

\( C_i, D_i \) and \( M \) are used to model anisotropy. The isotropic case corresponds to:

\[ \tilde{C}_i = C_i \tilde{1}, \tilde{D}_i = D_i \tilde{1} \] and \( \tilde{M} = \tilde{J} \).

Runge–Kutta is straightforward!
Multi–kinematic law: \( \theta \)–method

- The time discretization of the previous equations leads to:

\[
\begin{align*}
\mathbf{r}_e &= \Delta \varepsilon_e + \Delta p \mathbf{n} - \Delta \varepsilon_t = 0 \\
\mathbf{r}_{\alpha_i} &= \Delta \alpha_i - \Delta p \mathbf{n} + \Delta p \mathbf{D}_i : \alpha_i + \theta D_i \Delta \alpha_i \Delta p = 0 \\
r_r &= \Delta r - \Delta p (1 - br) = 0 \\
r_p &= \Delta p - \phi(f, \ldots) \Delta t = 0
\end{align*}
\]

- Variables \( \varepsilon_e, \alpha_i, r \) are considered e time \( t + \theta \Delta t \) and are equal to: \( \varepsilon_e(t) + \theta \Delta \varepsilon_e, \alpha_i(t) + \theta \Delta \alpha_i, r + \theta \Delta r \).

- Plasticity can be treated writing:

\[
 r_p = \left\| \mathbf{\sigma} - \mathbf{X} \right\|_M - R - \sigma_y = 0 \]

- The Jacobian matrix is expressed in the following slides ...
\[ \tilde{r}_e = \Delta \varepsilon_e + \Delta p \tilde{n} - \Delta \varepsilon_t = 0 \]

\[
\frac{\partial \tilde{r}_e}{\partial \Delta \varepsilon_e} = 1 + \theta \Delta p \tilde{N} C
\]

\[
\frac{\partial \tilde{r}_e}{\partial \Delta \alpha_i} = \Delta p \frac{\partial \tilde{n}}{\partial \tilde{X}_i} \frac{\partial \tilde{X}_i}{\partial \alpha_i} \frac{\partial \alpha_i}{\partial \Delta \alpha_i} = -\theta \Delta p \tilde{N} C_i
\]

\[
\frac{\partial \tilde{r}_e}{\partial \Delta r} = 0
\]

\[
\frac{\partial \tilde{r}_e}{\partial \Delta p} = \tilde{n}
\]
\begin{align*}
\mathbf{r}_{\alpha_i} &= \Delta \mathbf{\alpha}_i - \Delta p \mathbf{n} + \Delta p \mathbf{D}_i : \mathbf{\alpha}_i + \theta D_i \Delta \mathbf{\alpha}_i \Delta p = 0 \\
\frac{\partial \mathbf{r}_{\alpha_i}}{\partial \Delta \mathbf{\alpha}_e} &= -\Delta p \frac{\partial \mathbf{n}}{\partial \mathbf{\sigma}} \frac{\partial \mathbf{\sigma}}{\partial \Delta \mathbf{\epsilon}_e} \frac{\partial \mathbf{\epsilon}_e}{\partial \Delta \mathbf{\epsilon}_e} = -\theta \Delta p \mathbf{N} \mathbf{C} \\
\frac{\partial \mathbf{r}_{\alpha_i}}{\partial \Delta \mathbf{\alpha}_i} &= 1 + \theta \Delta p \mathbf{D}_i \\
\frac{\partial \mathbf{r}_{\alpha_i}}{\partial \Delta r} &= 0 \\
\frac{\partial \mathbf{r}_{\alpha_i}}{\partial \Delta p} &= -\mathbf{n} + \mathbf{D}_i : \mathbf{\alpha}_i
\end{align*}
\[ r_r = \Delta r - \Delta p(1 - br) = 0 \]

\[
\begin{align*}
\frac{\partial r_r}{\partial \Delta \varepsilon_e} & = 0 \\
\frac{\partial r_r}{\partial \Delta \alpha_i} & = 0 \\
\frac{\partial r_r}{\partial \Delta r} & = 1 + \theta \Delta p d \\
\frac{\partial r_r}{\partial \Delta p} & = br
\end{align*}
\]
\[ r_p = \Delta p - \phi(f, \ldots) \Delta t \]

\[
\frac{\partial r_p}{\partial \Delta \varepsilon_{\varepsilon}} = -\frac{\partial F}{\partial f} \frac{\partial f}{\partial \varepsilon} \frac{\partial \varepsilon}{\partial \varepsilon_{\varepsilon}} \frac{\partial \varepsilon_{\varepsilon}}{\partial \Delta \varepsilon_{\varepsilon}} = -\theta \Delta t \phi, f \mathbf{n} : \mathbf{C}_{\varepsilon} \\
\frac{\partial r_p}{\partial \Delta \alpha_i} = \theta \Delta t \phi, f \mathbf{n} : \mathbf{C}_i \\
\frac{\partial r_p}{\partial \Delta r} = -\frac{\partial \phi}{\partial R} \frac{\partial R}{\partial r} \frac{\partial r}{\partial \Delta r} \Delta t = \theta \Delta t c\phi, f \\
\frac{\partial r_p}{\partial \Delta p} = 1 \]
Multi–kinematic law : static recovery

Static recovery can easily be added by modifying the evolution laws for the hardening (both isotropic and kinematic) variables:

\[ \dot{\alpha}_i = \dot{\varepsilon}_p - \dot{p}D_i : \alpha_i - S_i : \alpha_i \]
\[ \dot{r} = \dot{p} - \dot{p}br - sr \]

In the calculation of the Jacobian matrix, the following terms must be added:

\[ -\theta \Delta tS_i \quad \dot{\alpha_i} \quad \frac{\partial r_{\alpha_i}}{\partial \Delta \alpha_i} \]
\[ -\theta \Delta ts \quad \dot{r} \quad \frac{\partial r_{\alpha_i}}{\partial \Delta r} \]
Multi–kinematic law: variable temperature

- Material coefficients may depend on external parameters and state variables.
- These coefficients must be evaluated at $t$, $t + \theta \Delta t$ or $t + \Delta t$.
- In the case of the Runge–Kutta integration, using a viscous creep law allows to bypass the computation of the plastic multiplier using the consistency condition.
- An error is often done...

The relationships

$$ X = \frac{2}{3} C \alpha \quad \dot{\alpha} = \dot{\varepsilon}_p - \frac{3}{2} \dot{\rho} \frac{D}{C} \alpha $$

are replaced by

$$ \dot{X} = \frac{2}{3} C \dot{\varepsilon}_p - D \dot{\rho} X $$

which is only valid if $C$ is a constant. In fact:

$$ \dot{X} = \frac{2}{3} \frac{d(C \alpha)}{dt} = \frac{2}{3} C_T \ddot{T} \alpha + \frac{2}{3} C \ddot{\alpha} $$
Comparaison of the results with

\[ C = 30000 \left(1 - \frac{T}{200}\right) \quad D = 200 \]

\[ K = 20 \quad n = 10 \quad R = 300 \quad E = 200000 \]

and the following load path
In both cases, ratchetting is obtained but the results strongly differ.