Non Linear Computational Mechanics - Athens MP06/2013

The Finite Element method for nonlinear structural problems

Vincent Chiaruttini
vincent.chiaruttini@onera.fr

ONERA
THE FRENCH AEROSPACE LAB

www.onera.fr
Outline

- Generalities on computational strategies for nonlinear problems
  - Examples (contact, crack propagation, non-linear behaviour, geometrical non-linearities)
  - Classical algorithm for nonlinear or time dependant problems

- Local numerical aspects of plasticity
  - Elastic-plastic behaviour
  - Local integration of non-linear models

- Global numerical aspects of plasticity
  - Solution process
  - Consistent tangent matrix

- Examples of solution process
- Presentation of Z-mat
Non-linearities in structural problems

- **Contact**
  - Due to the non-penetration condition

- **Crack propagation problem under time dependant loading**
  - When crack propagates the solution becomes non-linearly time dependant

- **Geometrical nonlinearities**
  - For large deformation, instabilities can also occur (buckling)

- **Nonlinear constitutive relationship**
  - Non linear behaviour: elastoplasticity, damage, viscosity
Solution process

- **Iterative algorithm**
  - Scalar example finding $u \mid f(u) = 0$
  - For any kind of regular function, no direct process exists
  - Iterative algorithm building $u_n \rightarrow u \mid f(u) = 0$
  - Stop when a convergence criterion is satisfied $\text{rank } k \mid |f(u_k)| < \varepsilon_{\text{crit}}$

- **Newton method**
  - Built on the linear verification of the first order Taylor development nullity
    $$f(u_{k+1}) \approx f(u_k) + (u_{k+1} - u_k) f'(u_k) = 0$$
    $$\Rightarrow u_{k+1} = u_k - f(u_k) / f'(u_k)$$

---

For any kind of regular function, no direct process exists. Iterative algorithm building $u_n \rightarrow u \mid f(u) = 0$. Stop when a convergence criterion is satisfied $\text{rank } k \mid |f(u_k)| < \varepsilon_{\text{crit}}$. Newton method built on the linear verification of the first order Taylor development nullity $f(u_{k+1}) \approx f(u_k) + (u_{k+1} - u_k) f'(u_k) = 0$. $\Rightarrow u_{k+1} = u_k - f(u_k) / f'(u_k)$.
Solution process

**Newton method**
- Built on the linear verification of the first order Taylor development nullity

\[ f(u_{k+1}) \approx f(u_k) + (u_{k+1} - u_k) f'(u_k) = 0 \]

\[ \Rightarrow u_{k+1} = u_k - \frac{f(u_k)}{f'(u_k)} \]

- Convergence depends on \( u_0 \)

- When converges
  - Rank k error \( e_k = u_k - u \)
  - Recurrence on error relationship \( e_{k+1} - e_k = u_{k+1} - u_k = -\frac{f(u_k)}{f'(u_k)} \)
  - Taylor expansion closely to the exact solution

\[ f(u_k) = f'(u)e_k + \frac{1}{2} f''(u)e_k^2 + o(e_k^2) \]

\[ f'(u_k) = f'(u) + f''(u)e_k + f'''(u)e_k^2 + o(e_k^2) \]

\[ e_{k+1} = e_k - \frac{2f'e_k + f''e_k^2}{2f' + f''e_k + f'''e_k^2} + o(e_k^2) = \frac{f''(u)}{2f'(u)}e_k^2 + o(e_k^2) = O(e_k^2) \]

- Quadratic convergence
  Close enough to the solution each iteration produces twice more significant new digits
Solution process

- **Newton method**
  - Quadratic convergence
    - Require to update the derivative at each iteration

- **Modified Newton methods**
  - Constant direction \( f'(u_k) \approx K = \text{cst} = f'(u_0) \)
    \[
    u_{k+1} = u_k - \frac{f(u_k)}{K}
    \]
  - Linear convergence \( e_{k+1} = (1 - f'(u)/K)e_k + o(|e_k|) = O(|e_k|) \)
Solution process

- **Newton method**
  - Quadratic convergence
    - Require to update the derivative at each iteration

- **Modified Newton methods**
  - Constant direction $f'(u_k) \approx K = \text{cst} = f'(u_0)$ Linear convergence
  - Secant update
    $$ u_{k+1} = u_k + f(u_k) \frac{u_k - u_{k-1}}{f(u_k) - f(u_{k-1})} $$
    $$ e_{k+1} = O(|e_k|^{1+\sqrt{5}/2}) $$
  - Golden ratio convergence order
Solution process

Newton method for a set of equations

- Vectorial function
  \[ F(U) = 0 \]

- At each iteration a linear system is solved
  \[ 0 = F(U_k) + \left( \frac{\partial F^i}{\partial U^j}(U_k) \right) U^j_k \]
  where the operator constitutes the rigidity matrix at \( U_k \)

- Quadratic convergence when close to solution
  - The neighbourhood where such kind of convergence is observed depends on the vectorial function properties

**Convergence is insured for convex functions**
Examples of convergence using a Newton algorithm

- Solve $f(x) = x^4 + x^2 - 1 = 0$
- Solution $x = \sqrt{\frac{\sqrt{5} - 1}{2}} \approx 0.7861513777574233$
- $f'(x) = 4x^3 + 2x$ \hspace{1cm} x := x - f(x)/f'(x)$
- Starting point, $x = 0.5$

<table>
<thead>
<tr>
<th>iter</th>
<th>$x$</th>
<th>$f(x)$</th>
<th>$f'$</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.9583333333334E+00</td>
<td>-6.87500000000000E+00</td>
<td>0.15000000000000E+01</td>
<td>0.1721819555759101E+00</td>
</tr>
<tr>
<td>2</td>
<td>0.818212508647757E+00</td>
<td>0.7618664400077163E+00</td>
<td>0.5437210648148149E+01</td>
<td>0.3206113089033369E-01</td>
</tr>
<tr>
<td>3</td>
<td>0.7874707949846017E+00</td>
<td>0.1176640788709701E+00</td>
<td>0.3827505524260790E+01</td>
<td>0.1319417227178432E-02</td>
</tr>
<tr>
<td>4</td>
<td>0.7861537049353379E+00</td>
<td>0.4646978771958370E-02</td>
<td>0.3528216445455353E+01</td>
<td>0.2327177914596135E-05</td>
</tr>
<tr>
<td>5</td>
<td>0.7861513777646758E+00</td>
<td>0.818161391923402E-05</td>
<td>0.3515797756248939E+01</td>
<td>0.7252531908363835E-11</td>
</tr>
<tr>
<td>6</td>
<td>0.7861513777574233E+00</td>
<td>0.2549823861167783E-10</td>
<td>0.3515775842609722E+01</td>
<td>0.0000000000000000E+00</td>
</tr>
</tbody>
</table>
Examples of convergence using a Newton algorithm

- Express \( f(x) = x^4 + x^2 - 1 = 0 \) as
  \[
  \begin{align*}
  f_1(x_1, x_2) &= x_1^2 + x_2^2 - 1 = 0 \\
  f_2(x_1, x_2) &= x_1^2 - x_2 = 0
  \end{align*}
  \]

- Solution \( x_1 = \sqrt{\frac{\sqrt{5} - 1}{2}} \approx 0.7861513777574233 \)

- Jacobian matrix:
  \[
  J = \begin{pmatrix}
  \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\
  \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2}
  \end{pmatrix}
  = \begin{pmatrix}
  2x_1 & 2x_2 \\
  2x_1 & -1
  \end{pmatrix}
  \]

  \[
  J^{-1} = \frac{1}{D} \begin{pmatrix}
  -1 & -2x_2 \\
  -2x_1 & 2x_1
  \end{pmatrix}
  \]

  with \( D = -2x_1 - 4x_1x_2 \)

- Iterative process:

  \[
  \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} := \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - J^{-1} \begin{pmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{pmatrix}
  \]

  Starting point, \( x_1 = 0.5 \quad x_2 = 1 \)

<table>
<thead>
<tr>
<th>iter</th>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( f_1(x) )</th>
<th>( f_2(x) )</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.91666667908430E+00</td>
<td>0.66666665673255E+00</td>
<td>0.25000000000000E+00</td>
<td>-0.75000000000000E+00</td>
<td>0.13051530132687E+00</td>
</tr>
<tr>
<td>2</td>
<td>0.79599567399256E+00</td>
<td>0.61904761857456E+00</td>
<td>0.2847223174240E+00</td>
<td>0.1736114381088E+00</td>
<td>0.98442962351444E-02</td>
</tr>
<tr>
<td>3</td>
<td>0.78621253923518E+00</td>
<td>0.6180344780730E+00</td>
<td>0.1682906707772E-01</td>
<td>0.1456149440315E-01</td>
<td>0.61161477763582E-04</td>
</tr>
<tr>
<td>4</td>
<td>0.78615138013714E+00</td>
<td>0.6180339874998E+00</td>
<td>0.96735527118568E-04</td>
<td>0.9570904336912E-04</td>
<td>0.23797219750321E-08</td>
</tr>
<tr>
<td>5</td>
<td>0.78615137775742E+00</td>
<td>0.61803398874989E+00</td>
<td>0.37417605327095E-08</td>
<td>0.37415486510905E-08</td>
<td>0.00000000000000E+00</td>
</tr>
<tr>
<td>6</td>
<td>0.78615137775742E+00</td>
<td>0.61803398874989E+00</td>
<td>0.5019856058607E-16</td>
<td>-0.71232082732297E-16</td>
<td>0.00000000000000E+00</td>
</tr>
</tbody>
</table>
ODE integration

- Time dependant problem are ruled by differential equations
  - Reduce high order differential systems to first order
    \[
    \frac{d^2 y}{dt^2} + g(t) \frac{dy}{dt} = r(t) \iff \begin{cases} 
    \frac{dy}{dt} = z(t) \\
    \frac{dz}{dt} = r(t) - g(t) z(t)
    \end{cases}
    \]
  - General formulation
    \[
    \dot{\mathbf{Y}} = [F(t, \mathbf{Y})] \quad \dot{\mathbf{Y}}(t = t_0) = [\mathbf{Y}_0]
    \]
- Euler-type integration schemes
  - Finite difference time discretization
    \[
    \dot{\mathbf{Y}}(t_n) = Y_n, \quad t_{n+1} = t_n + \Delta t, \quad \dot{\mathbf{Y}}(t) = F(t, \mathbf{Y}) 
    \]
    forwrd methods \[
    \dot{\mathbf{Y}}(t_{n+1}) \approx \frac{\mathbf{Y}_{n+1} - \mathbf{Y}_n}{\Delta t}
    \]
  - \(\theta\)-method (method-B)
    \[
    \frac{\mathbf{Y}_{n+1} - \mathbf{Y}_n}{\Delta t} = \theta F_{n+1}(\mathbf{Y}_{n+1}) + (1 - \theta) F_n(\mathbf{Y}_n)
    \]
  - Explicit Forward Euler (\(\theta=0\))
    Conditionally stable
    1\textsuperscript{st} order accurate
  - Crank-Nicholson (\(\theta=0.5\))
    Unconditionally stable
    2\textsuperscript{nd} order accurate
  - Implicit Euler (\(\theta=1\))
    Unconditionally stable
    1\textsuperscript{st} order accurate
**ODE integration**

- **Time dependant problem are ruled by differential equations**
  - Runge-Kutta explicit integration
    - Minimal multiple evaluations of \( F(t, y(t)) \) on a given time increment to insure a specific order accuracy, based on Taylor expansion
      \[
      y_{n+1} = y_n + \Delta t y'_n + (\Delta t/2) y''_n + o(\Delta t^2)
      \]
    - RK1 is the forward explicit Euler scheme
    - RK2 using one mid-point sub calculation at \( t_{n+\frac{1}{2}} \)
      \[
      y_{n+1/2} = y_n + \frac{\Delta t}{2} f(t_n, y_n)
      \]
      \[
      y_{n+1} = y_n + \Delta t f(t_{n+1/2}, y_{n+1/2})
      \]
    - 2\(^{nd}\) order method
    - RK4 popular method using 4 sub calculations
      \[
      y_{n+1} = y_n + \Delta t/6 (k_1 + 2k_2 + 2k_3 + k_4)
      \]
      \[
      k_1 = f(t_n, y_n) \quad \text{Initial slope on the interval}
      \]
      \[
      k_2 = f(t_{n+1/2}, y_n + (\Delta t/2) k_1) \quad \text{Mid interval slope using } k_1
      \]
      \[
      k_3 = f(t_{n+1/2}, y_n + (\Delta t/2) k_2) \quad \text{Mid interval slope using } k_2
      \]
      \[
      k_4 = f(t_{n+1/2}, y_n + \Delta t k_3) \quad \text{Final slope using } k_3 \text{ on the interval}
      \]
    - 4\(^{th}\) order method
Outline

- **Generalities on computational strategies for nonlinear problems**
  - Examples (contact, crack propagation, non-linear behaviour, geometrical non-linearities)
  - Classical algorithm for nonlinear or time dependant problems

- **Local numerical aspects of plasticity**
  - Elastic-plastic behaviour
  - Local integration of non-linear models

- **Global numerical aspects of plasticity**
  - Solution process
  - Consistent tangent matrix

- **Examples of solution process**

- **Presentation of Z-mat**
Local numerical aspects of plasticity

Assumptions
- Small strain: linearised kinematic
- Quasi-static loadings: no dynamic effects

Small strain elastoplastic problem
- Elastic domain
  \[ f(\sigma) < 0 \]
  - A usual elastic behaviour is applied
- Yield surface
  \[ f(\sigma) = 0 \]
  - Irreversible/dissipative phenomena can occur
- Example Von Mises criterion
  \[ f(\sigma) = \sqrt{3/2} \sigma_D : \sigma_D - R \quad \text{with} \quad \sigma_D = \sigma - \frac{1}{3} \text{Tr}(\sigma) I \]
Local numerical aspects of plasticity

- Small strain elastoplastic problem
  - Plastic strain
    - Strain partition
      \[ \varepsilon = \varepsilon^e + \varepsilon^{in} = \varepsilon^e + \varepsilon^p \]
  - Rigidity
    \[ \sigma = A : \varepsilon^e = A : (\varepsilon - \varepsilon^p) \]

- Yield surface evolution (convex), hardening
  - When \( f = 0 \) the yield surface evolves
    - Isotropic hardening \( \rightarrow \) yield surface increase
    - Kinematic hardening \( \rightarrow \) yield surface translation

- Flow rules
  - Plasticity (normality rule)
    \[ \dot{\varepsilon}^p = \dot{\lambda} \frac{\partial f}{\partial \sigma}, \quad \dot{\lambda} \geq 0, \quad f(\sigma) \leq 0, \quad \dot{\lambda} f(\sigma) = 0 \]
  - Cumulated plastic strain rate
    \[ p(t) = \int_0^t \sqrt{\frac{3}{2} (\dot{\varepsilon}^p(\tau) : \dot{\varepsilon}^p(\tau))} d\tau \quad (= \dot{\lambda} \text{ for a Von Mises criterion}) \]
Local numerical aspects of plasticity

- **Common formalism for viscoplasticity/multipotentials/large strain**
  - **Strain partition**
    - **Small strain**
      \[ \varepsilon = \varepsilon^e + \varepsilon^{in} = \varepsilon^e + \varepsilon^p + \varepsilon^{vp} \]
    - **Large strain**
      \[ \mathbf{F} = \mathbf{E} \mathbf{P} \]
  - **Elasticity**
    \[ \sigma = \mathbf{A} : \varepsilon^e \]
  - **Flow rules**
    - **Plasticity (state must stay on the yield surface at the end of the increment)**
      \[ \dot{\varepsilon}^p = \sum_s \dot{\lambda}^s \mathbf{n}^s \quad \dot{\lambda}^s \text{ from } \dot{f}^s = 0 \]
    - **Viscoplasticity (state expected to be on the relevant equipotential at the end of the increment)**
      \[ \dot{\varepsilon}^{vp} = \sum_s \dot{\mathbf{v}}^s \mathbf{m}^s \quad \dot{\mathbf{v}}^s = \left( \frac{f^s}{K} \right)^n \]
  - **Hardening rules**
    \[ \dot{Y}_I = fct(Y_I, \varepsilon^p, \varepsilon^{vp}) \]
Elastoplastic example

Relations \( t \in [0, T] \)

Equilibrium and strain definition

\[
\nabla \sigma + f = 0 \\
\varepsilon = \frac{1}{2} (\nabla u + \nabla^T u)
\]

Behaviour

\[
\sigma = A : (\varepsilon - \varepsilon^p) \\
\sigma^{eq} = \sqrt{\frac{3}{2}} ||\sigma_D|| \\
\dot{\varepsilon}^p = \dot{p} \frac{3}{2\sigma^{eq}} \sigma_D \\
\dot{p} \geq 0 \\
\sigma^{eq} - R(p) \leq 0 \\
\dot{p} (\sigma^{eq} - R(p)) = 0
\]

Boundary conditions (initial equilibrium and no plasticity)

\[
\begin{align*}
\mathbf{u} &= \mathbf{u}_d \text{ on } \partial_u \Omega \\
\mathbf{\sigma} \cdot \mathbf{n} &= \mathbf{F} \text{ on } \partial_f \Omega \\
\varepsilon^p(t = 0) &= 0
\end{align*}
\]
Numerical solution process

- **Elastoplastic example**
  - Mechanical state: \( S(x, t) = \{ u(x, t), \varepsilon(x, t), \varepsilon^p(x, t), \sigma(x, t) \} \)
  - Temporal discretization
    - Incremental temporal approach using a regular grid: \( t_{n+1} = t_n + \Delta t \)
    The solution is searched incrementally at each time step \( t_{n+1} \) (all previous step states being known)
  - Spatial discretization
    - FE method in displacement
      \[
      \forall u^* \in U_{ad}^0, \quad \int_\Omega \sigma_{n+1} : \varepsilon^* d\Omega = \int_\Omega f_{n+1} \cdot u^* d\Omega + \int_{\partial F \Omega} F_{n+1} \cdot u^* dS
      \]
      - Nonlinearity comes from the non-linear relation between stress and strain relation
  - Local integration of the mechanical behaviour
    - At each point of the structure process is defined by the following process
      \[
      (u_{n+1}, S_n) \rightarrow \sigma_{n+1} = F(u_{n+1}, S_n)
      \]
  - Global equilibrium
    - Verified by the solution of the nonlinear variational formulation
      \[
      R(u_{n+1}; u^*, S_n) = 0, \quad \forall u^* \in U_{ad}^0
      \]
Local behaviour integration process

- **Generic interface for any constitutive equation**
  - Gauss point process (where the element variational formulation is integrated)
  - Definition
    - *External parameters* \( (\epsilon_p) \) imposed as input
    - *Integrated variables* \( (v_{int}) \)
    - *Auxiliary variables* \( (v_{aux}) \) just for output
    - *Coefficients* \( (coef) \) material parameters
  - *Primal and dual variables* prescribed variables and associated fluxes
    - Primal: strain increment on the current time interval (input to the local integration process)
    - Dual: stress obtained useful for variational formulation (as output of the local integration process)

\[
\mathcal{F}(u_{n+1}, S_n)
\]
Local behaviour integration process

- **Time integration**
  - Normality rule as an ODE
    \[ \dot{\varepsilon}^p = \dot{\sigma} \left( \frac{3}{2\sigma} \right) \varepsilon_{eq} \Rightarrow \varepsilon_{n+1}^p = \varepsilon_n^p + \int_{t_n}^{t_{n+1}} \dot{\varepsilon}^p(\tau) \, d\tau \]
  - Explicit integration → Runge-Kutta (RK4 usually, beware of stability condition)
  - Implicit integration → θ-methods (stable but requires a local Jacobian computation)

\[ \text{Interval} \quad [1 : 4] \]

\[ f(x) = 1 + \sqrt{x} \quad f'(x) = 0.5 / \sqrt{x} \]
\[ f_0 = 2 \quad f_0' = 0.5 \]
\[ f_1 = 3 \quad f_1' = 0.25 \]
\[ f_{1/2} = 1 + \sqrt{2.5} \quad f_{1/2}' = 0.5 / \sqrt{2.5} \]

- \( \theta = 0: f_1 \approx 2 + 3 \times 0.5 = 3.5 \)
- \( \theta = 1: f_1 \approx 2 + 3 \times 0.25 = 2.75 \)
- \( \theta = 1/2: f_1 \approx 2 + 3 \times 0.5 \times 0.375 = 3.125 \)
Local behaviour integration process

- **Elastic prediction and correction algorithm**

  - Solution depends on the test $\sigma^{eq} - R(p) \leq 0$ (is the evolution purely elastic on $\Delta t$?)
  
  - Elastic prediction
    
    $$\sigma^{e}_{n+1} = \sigma_{n} + A : \Delta \varepsilon^{e}_{n}$$
  
  - Convexity of the yield surface gives
    
    $$f(\sigma^{e}_{n+1}, p_{n}) \geq f(\sigma^{e}_{n+1}, p_{n} + \Delta p_{n})$$
  
  - $f(\sigma^{e}_{n+1}, p_{n}) \leq 0$ purely elastic evolution, prediction is correct
    
    $$\begin{cases}
    \sigma^{e}_{n+1} = \sigma^{e}_{n+1} \\
    \varepsilon^{p}_{n+1} = \varepsilon^{p}_{n} \\
    p_{n+1} = p_{n}
    \end{cases}$$
  
  - Else a plastic evolution is observed
    
    - A correction must be applied on the elastic prediction
    
    $$\Delta p_{n} > 0, \quad \Delta p_{n} \cdot f(\sigma^{e}_{n+1}, p_{n} + \Delta p_{n}) = 0 \Rightarrow f(\sigma^{e}_{n+1}, p_{n} + \Delta p_{n}) = 0$$
    
    $$\sigma^{e}_{n+1} = \sigma^{e}_{n+1} - A : \Delta \varepsilon^{p}_{n} \quad \Delta \varepsilon^{p}_{n} = \Delta p_{n} \sqrt{\frac{3}{2} N_{n+1}} \quad \Delta p_{n} > 0$$
    
    where $N_{n+1}$ is the outward unit normal to the final yield surface
Local behaviour integration process

- Elastic prediction and correction algorithm
  - Solution depends on the test \( \sigma^{eq} - R(p) \leq 0 \) (is the evolution purely elastic on \( \Delta t \) ?)
    - Elastic prediction
      \[
      \sigma_{n+1}^e = \sigma_n + A : \Delta \varepsilon_{n}^e
      \]
    - Convexity of the yield surface gives
      \[
      f(\sigma_{n+1}^e, p_n) \geq f(\sigma_{n+1}^e, p_n + \Delta p_n)
      \]
    - \( f(\sigma_{n+1}^e, p_n) \leq 0 \) purely elastic evolution, prediction is correct
    - Else a plastic evolution is observed
      - A correction must be applied on the elastic prediction

\[
\Delta p_n > 0, \quad \Delta p_n f(\sigma_{n+1}^e, p_n + \Delta p_n) = 0 \Rightarrow f(\sigma_{n+1}^e, p_n + \Delta p_n) = 0
\]
\[
\sigma_{n+1} = \sigma_{n+1}^e - A : \Delta \varepsilon_{n}^p \quad \Delta \varepsilon_{n}^p = \Delta p_n \sqrt{\frac{3}{2} \frac{N}{n+1}} \quad \Delta p_n > 0
\]

where \( \frac{N}{n+1} \) is the outward unit normal to the final yield surface
Radial return algorithm

- For von Mises based criteria or isotropic or linear kinematic hardening
  - The corrective term \( A : \Delta \varepsilon_p \) is oriented by the final normal that can be computed \textit{a priori}

- For von Mises criterion the final normal is collinear to the elastic predictor deviator

\[
\frac{N_{n+1}}{\sigma_{Dn+1}} = \frac{\sigma_{e}^{e}D_{n+1}}{||\sigma_{e}^{e}D_{n+1}||} \quad \sigma_{n+1} = \sigma_{n+1}^e - A : \Delta p_n \sqrt{\frac{3}{2N_{n+1}}}
\]

\[
\sigma_{n+1}^{eq} = \sigma_{n+1}^{e,eq} - 3\mu \Delta p_n \quad \mu \text{ Lamé coefficient}
\]

- Final state on the yield surface gives to verify

\[
\sigma_{n+1}^{e,eq} - 3\mu \Delta p_n - R(p_n + \Delta p_n) = 0
\]

that allows to find \( \Delta p_n \)
Generalized radial return algorithm

- Closest point projection technique

  - For a generalized normality rule

    \[ \Delta \varepsilon^p = \frac{\partial f}{\partial \sigma} \Delta p \quad \Delta \alpha_I = \frac{\partial f}{\partial Y_I} \Delta p \]

  - Fluxes

    \[ \Delta \sigma = A : (\Delta \varepsilon - \Delta \varepsilon^p) \]
    \[ \Delta Y_I = M_I \Delta \alpha_I \]

    \[ \Delta \Sigma = \left[ \frac{\Delta \sigma}{\Delta Y_I} \right] = \left[ A : \frac{\Delta \varepsilon}{\Delta \alpha_I} \right] - \left[ \frac{A}{0} \right] \frac{\partial f}{\partial \sigma} + \left[ \begin{array}{c} 0 \\ M_I \end{array} \right] \frac{\partial f}{\partial Y_I} \right] \Delta p \]
Generic formulation of the local integration process

Find state variables increment, $\Delta \xi^e$ and $\Delta \alpha_I$, using strain partition rule and hardening rules.

$$
\mathcal{F}_e = \Delta \xi^e + \Delta \rho \eta_\theta + \Delta \xi^{th} - \Delta \xi
$$

$$
\mathcal{F}_{pi} = r_{pi}(\xi^e, \alpha_I)
$$

Jacobian matrix $[J] =$

$$
\begin{pmatrix}
\frac{\partial \mathcal{F}_e}{\partial \Delta \xi^e} & \frac{\partial \mathcal{F}_e}{\partial \Delta \alpha_I} \\
\frac{\partial \mathcal{F}_{pi}}{\partial \Delta \xi^e} & \frac{\partial \mathcal{F}_{pi}}{\partial \Delta \alpha_I}
\end{pmatrix}
$$

Note:

$$
\mathcal{N} = \frac{\partial \eta}{\partial \Delta \eta} = \frac{\partial}{\partial \Delta \eta} \left( \frac{3}{2} \mathcal{S} \right) = \frac{1}{J} \left( \frac{3}{2} \mathcal{J} - \eta \otimes \eta \right)
$$

(with $\mathcal{S} = \mathcal{J} : \sigma$) accounts for normal rotation during the increment.
Outline

- Generalities on computational strategies for nonlinear problems
  - Examples (contact, crack propagation, non-linear behaviour, geometrical non-linearities)
  - Classical algorithm for nonlinear or time dependant problems

- Local numerical aspects of plasticity
  - Elastic-plastic behaviour
  - Local integration of non-linear models

- Global numerical aspects of plasticity
  - Solution process
  - Consistent tangent matrix

- Examples of solution process

- Presentation of Z-mat
Global aspects

- **Solving the global discrete problem**
  - **Global problem**
    \( \mathcal{R}(u_{n+1}; u^*, S_n) = 0, \forall u^* \in \mathcal{U}^0_{ad} \) with \( \mathcal{R}(u_{n+1}; u^*, S_n) = \)
    \[
    \int_{\Omega} \mathcal{F}(u_{n+1}, S_n) : \varepsilon^* d\Omega - \int_{\Omega} f_{n+1} u^* d\Omega - \int_{\partial F \Omega} F_{n+1} u^* dS
    \]
  - **Global Newton algorithm**
    - At each time step the solution is searched iteratively starting from the last converged solution leading to the following linear system at iteration \( i \)
      \[
      \mathcal{R}(u^i_{n+1}; u^*, S_n) + \mathcal{R}'(u^i_{n+1}; u^*, S_n)[u_{n+1}^{i+1} - u_{n+1}^i] = 0, \forall u^* \in \mathcal{U}^0_{ad}
      \]
    - To obtain the residual evaluation, it is necessary to perform the local integration to get
      \[
      \mathcal{F}(u^i_{n+1}, S_n)
      \]
    - To calculate the global consistent tangent operator, it is necessary to obtain a local consistent tangent operator (that can be kept constant if a modified Newton algorithm with constant operator is involved)
      \[
      \mathcal{R}'(u^i_{n+1}; u^*, S_n) = \int_{\Omega} \frac{\partial \mathcal{F}}{\partial u}(u_{n+1}, S_n) : \varepsilon^* d\Omega
      \]
    - When the residual is small enough the solution is reached
      \[
      \| \mathcal{R}(u^i_{n+1}; u^*, S_n) \| < \eta_{NL}
      \]
Local consistent tangent operator

In the local integration process

After convergence,
\[
\begin{pmatrix}
    d\Delta \varepsilon \\
    0
\end{pmatrix}
= [J]
\begin{pmatrix}
    d\Delta \varepsilon^e \\
    d\Delta \alpha_i
\end{pmatrix}
\]

... then
\[
\begin{pmatrix}
    d\Delta \varepsilon^e \\
    d\Delta \alpha_i
\end{pmatrix}
= [J]^{-1}
\begin{pmatrix}
    d\Delta \varepsilon \\
    0
\end{pmatrix}
\]

\[
[J]^{-1} = \begin{pmatrix}
    \frac{\partial \varepsilon^e}{\partial \varepsilon} & x \\
    x & x
\end{pmatrix}
\]

with \[
[H] = \frac{\partial \varepsilon^e}{\partial \varepsilon}\]

Consistent tangent matrix:

\[
\mathbf{L}_c = \frac{\partial \varepsilon}{\partial \varepsilon^e} : \frac{\partial \Delta \varepsilon^e}{\partial \varepsilon} = \Lambda : [H]
\]
Outline

- Generalities on computational strategies for nonlinear problems
  - Examples (contact, crack propagation, non-linear behaviour, geometrical non-linearities)
  - Classical algorithm for nonlinear or time dependant problems
- Local numerical aspects of plasticity
  - Elastic-plastic behaviour
  - Local integration of non-linear models
- Global numerical aspects of plasticity
  - Solution process
  - Consistent tangent matrix
- Examples of solution process
- Presentation of Z-mat
Viscoelastic with isotropic hardening

\[ f(\sigma, R) = J(\sigma) - R - \sigma_y \]

\[ J(\sigma) = \sqrt{(3/2)s : s}; \quad s = \sigma - (1/3) \text{trace} \sigma; \quad \sigma_y = \text{init yield} \]

\[ \dot{\varepsilon}^p = \dot{\rho}n; \quad \dot{\rho} = \sqrt{(2/3)\dot{\varepsilon}^p : \dot{\varepsilon}^p}; \quad R = (1 - \exp(-bp)) \]

**Time independent (TI) behavior:** \( f = 0 \)

**Time dependent (TD) behavior:** \( \dot{\rho} = \left( \frac{f}{K} \right)^n \)

<table>
<thead>
<tr>
<th></th>
<th>3D</th>
<th>1D tension</th>
</tr>
</thead>
<tbody>
<tr>
<td>TI</td>
<td>( J(\sigma) - R - \sigma_y = 0 )</td>
<td>( \sigma = R + \sigma_y )</td>
</tr>
<tr>
<td>TD</td>
<td>( J(\sigma) - R - \sigma_y = K\dot{\rho}^{1/n} )</td>
<td>( \sigma = R + \sigma_y + K(\dot{\varepsilon}^p)^{1/n} )</td>
</tr>
</tbody>
</table>
**Viscoelastic with isotropic hardening**

**Implementation**

Unkowns = $\Delta \varepsilon^e$, $\Delta p$

Time–independent plasticity:

$$
\mathcal{F}_e = \Delta \varepsilon^e + \Delta p n_\theta + \Delta \varepsilon^{th} - \Delta \varepsilon
$$

$$
\mathcal{F}_p = f(\vec{\sigma}_{t+\Delta t}) = 0
$$

$\Delta p$ is the increment of equiv (visco-)plastic strain

$n_\theta$ is the normal to the yield surface at $t + \theta \Delta t$

Time–dependent plasticity, replace previous $r_p$ by:

$$
\mathcal{F}_p = \Delta p - \Delta t \left( \frac{f}{K} \right)^n = 0 \quad \text{or} \quad \mathcal{F}_p = f - K \left( \frac{\Delta p}{\Delta t} \right)^{1/n} = 0
$$
Viscoelastic with isotropic hardening

**Implementation**

Time-independent plasticity:

\[
[J] = \begin{pmatrix}
\mathbf{1} + \theta \mathbf{N}_\theta : \mathbf{\Lambda}_\theta \Delta p \\
\mathbf{\Lambda}_1 : \mathbf{n}_1 \\
-\mathbf{n}_\theta \\
-\mathbf{H} = -dR/dp
\end{pmatrix}
\]

Incremental consistent operator \( \mathbf{L}_c \) versus tangent continuous operator \( \mathbf{L}_t \)

\[
\mathbf{L}_c = \mathbf{L}_t - 4\mu^2 \Delta p \mathbf{N}
\]
Viscoelastic with isotropic hardening

Implementation

Time-dependent plasticity, now

\[ F_p = \Delta p - \Delta t \left\langle \frac{f}{K} \right\rangle^n = 0 \]

\[ \frac{\partial F_p}{\partial \Delta p} = 1 \; ; \; \frac{\partial F_p}{\partial \Delta \varepsilon^e} = \frac{\partial F_p}{\partial \Delta \sigma} \frac{\partial \Delta \sigma}{\partial \Delta \varepsilon^e} = \frac{n}{K} \left( \frac{f}{K} \right)^{n-1} \]
Large strain viscoelastic with isotropic hardening

- Elasticity: \( T = \Lambda : E^e \quad E^e = \log \mathbf{U}^e \)
- Flow: \( L^p = D^p \) with \( D^p = \mathbf{p} \mathbf{n} = \mathbf{p} \frac{\partial f}{\partial \mathbf{T}} \)
- Integration: \( \dot{F}^p = D^p F^p \quad F^p_{n+1} = \exp \left( D^p_{n+1} \Delta t \right) F^p_n \)

- After a few manipulations...

\[
F^e_{n+1} = F^e_{n+1} F^p_{n+1}^{-1} = F^e_{n+1} F^p_{n}^{-1} \exp \left( D^p_{n+1} \Delta t \right) = F^* \exp \left( D^p_{n+1} \Delta t \right)
\]

\[
F^* = F^e_{n+1} F^p_{n}^{-1} = R^* U^*
\]

\[
F^* = F^e_{n+1} \exp \left( D^p_{n+1} \Delta t \right) = R^e_{n+1} U^e_{n+1} \exp \left( D^p_{n+1} \Delta t \right)
\]

\[
R^* = R^e_{n+1} \quad U^* = U^e_{n+1} \exp \left( D^p_{n+1} \Delta t \right)
\]

- Final additive form for the incremental strain partition

\[
E^* = E^e_{n+1} + D^p_{n+1} \Delta t = E^e_{n+1} + \mathbf{n}_{n+1} \Delta \mathbf{p}
\]
System of residuals

- For plasticity, \( f(\mathbf{T}, R) = J((\mathbf{T})) - R = 0 \)
- Viscoplasticity, \( f(\mathbf{T}_{n+1}, R_{n+1}) - K \left( \frac{\Delta p}{\Delta t} \right)^{1/n} = 0 \)
- The unknowns are \( \mathbf{E}_{n+1}^e \) et \( \Delta p \), and the system is formed by 2 and either 3 or 4

\[
\begin{align*}
\mathcal{F}_e &= -\mathbf{E}^* + \mathbf{E}_{n+1}^e + n_{n+1} \Delta p \\
\mathcal{F}_p &= J(\Lambda : \mathbf{E}_{n+1}^e) - R(p + \Delta p) \\
\mathcal{F}_p &= J(\Lambda : \mathbf{E}_{n+1}^e) - R(p + \Delta p) - K \left( \frac{\Delta p}{\Delta t} \right)^{1/n} \\
\text{with } n_{n+1} &= \frac{3}{2} \frac{T'_{n+1}}{J(T_{n+1})} \\
T_{n+1} &= \Lambda : \mathbf{E}_{n+1}^e
\end{align*}
\]
Algorithm

- PreStep at the beginning of the step

$$\tilde{F}^* = \tilde{F}_{n+1} \tilde{F}^{-1}_n$$  $$\tilde{F}^* = R^* U^*$$  $$E^* = \log(U^*)$$

- StrainPart to compute Cauchy stress and $F^p$, which is saved as an auxiliary variable to compute $F^e$ at the beginning of the next step

$$T_{n+1} = \tilde{\Lambda} : E^e_{n+1}$$  $$\sigma_{n+1} = R^* T_{n+1} R^* T$$

$$U^e_{n+1} = \exp(E^e_{n+1})$$  $$F^e_{n+1} = R^* U^e_{n+1}$$  $$F^p_{n+1} = (F^e_{n+1})^{-1} F_{n+1}$$

- CalcGradF to express the residuals and their derivatives

$$\begin{pmatrix}
\frac{\partial F_e}{\partial E^e_{n+1}} & \frac{\partial F_e}{\partial \Delta p} \\
\frac{\partial F_p}{\partial E^e_{n+1}} & \frac{\partial F_p}{\partial \Delta p}
\end{pmatrix} = \begin{pmatrix}
I + \Delta p \frac{\partial n_{n+1}}{\partial T_{n+1}} : \tilde{\Lambda} \\
\tilde{n}_{n+1} : \tilde{\Lambda} - \frac{\partial R}{\partial \Delta p} - \frac{K}{n \Delta t} \left\langle \frac{\Delta p}{\Delta t} \right\rangle^{(1/n)-1}
\end{pmatrix}$$
Tangent matrix

\[
\begin{pmatrix}
\frac{\partial F_e}{\partial E_{e,n+1}} & \frac{\partial F_e}{\partial \Delta p} \\
\frac{\partial F_p}{\partial E_{e,n+1}} & \frac{\partial F_p}{\partial \Delta p}
\end{pmatrix}
\begin{pmatrix}
E_{e,n+1}^e \\
\Delta p
\end{pmatrix}
= [H]
\begin{pmatrix}
E_{e,n+1}^e \\
\Delta p
\end{pmatrix}
= \begin{pmatrix}
E^* \\
0
\end{pmatrix}
\]

- The top left block of \([H]^{-1}\) is \(\frac{\partial E_{e,n+1}^e}{\partial E^*}\), so that, at convergence

\[
\frac{\partial \sigma_{n+1}}{\partial E^*} = \frac{\partial \sigma_{n+1}}{\partial T_{\tilde{\alpha},n+1}^\pi} \cdot \frac{\partial T_{n+1}}{\partial E^*}
\]

avec \(\frac{\partial T_{n+1}}{\partial E^*} = \Lambda : \frac{\partial E_{e,n+1}^e}{\partial E^*}\)

- Since \(\tilde{\alpha}^*\) does not depend on \(\tilde{T}\)

\[
\frac{\partial \sigma_{n+1}}{\partial T_{\tilde{\alpha},n+1}^\pi} = -\frac{1}{J^2} \frac{\partial J}{\partial T_{\tilde{\alpha},n+1}^\pi} R^* T_{\tilde{\alpha},n+1}^\pi R^* T + \frac{1}{J} R^* I R^* T
\]

- and:

\[
\frac{\partial \sigma_{n+1}}{\partial E^*} = \frac{1}{J} \left( R^* I R^* T \right) : \left( \Lambda : \frac{\partial E_{e,n+1}^e}{\partial E^*} \right)
\]
Outline

- Generalities on computational strategies for nonlinear problems
  - Examples (contact, crack propagation, non-linear behaviour, geometrical non-linearities)
  - Classical algorithm for nonlinear or time dependant problems
- Local numerical aspects of plasticity
  - Elastic-plastic behaviour
  - Local integration of non-linear models
- Global numerical aspects of plasticity
  - Solution process
  - Consistent tangent matrix
- Examples of solution process
- Presentation of Z-mat
Presentation of the Z-mat library

- Numerous material models, plus user material
- Interface with the classical FE softwares
- Provide automatic time stepping and consistent tangent stiffness
- Coefficients presenting unlimited dependence on internal variables
- ZeBFRoNT, automatic code generation
- MuLTiMaT concept, for recursive multiscale modeling
Inside Z-mat

- Constitutive Model
  - Integration Methods
    - Runge-Kutta
    - Theta method
  - Constitutive Models
    - Basic Objects
      - Elasticity
      - Thermal Strain
      - Isotropic Hardening
      - Kinematic Hardening
    - Predefined Constitutive Models
  - ...
Object oriented design

Material objects
- Flow
- Behavior
- Isotropic hardening
- Criterion
- Kinematic hardening
- Elasticity
- Damage
- Potential
- Thermal strain
- etc...

Typical assembly for viscoplasticity
- Behavior
- Elasticity
- Thermal strain
- Potential
- Criterion
- Flow
- Isotropic hardening
- Kinematic hardening

Isotropie and non-linear kinematic model

\[ \varepsilon^h = \alpha(T - T_{ref}) \]

\[ f = J \left( \sigma' - \sum_i X_i \right) - R \]

\[ \dot{p} = \left( \frac{f}{k} \right)^n, \quad \varepsilon^{ev} = \dot{p}n \]

\[ R = R_0 + Q(1 - e^{-bp}) \]

\[ \dot{X} = \frac{2}{3} C \alpha, \quad \ddot{\alpha} = \dot{p} \left[ n - \frac{3D}{2C} \dot{X} \right] \]
### Datafile examples

#### Plasticity
- ***behavior gen_evp
- **elasticity isotropic
  - young 100000.
  - poisson 0.3
- **potential gen_evp ep
- *criterion mises
- *flow plasticity
- *isotropic nonlinear
  - R0 210. Q 50. b 10.
- *kinematic nonlinear
  - C 20000. D 500.

#### Viscoplasticity
- ***behavior gen_evp
- **elasticity isotropic
  - young 100000.
  - poisson 0.3
- **potential gen_evp ev
  - *criterion mises
  - *flow norton
    - K 1000. n 4.5
  - *isotropic nonlinear
    - R0 210. Q 50. b 10.
  - *kinematic nonlinear
    - C 20000. D 500.

#### Crystal viscoplasticity
- ***behavior gen_evp
- **elasticity cubic
  - y1111 100000.
  - y1122 75000.
  - y1212 112000.
- **potential octahedral
  - *flow norton
    - K 1000. n 4.5
  - *isotropic nonlinear
    - R0 210. Q 50. b 10.
  - *kinematic nonlinear
    - C 20000. D 500.
  - *interaction slip
    - h1 1. h2 1.2 h3 1.4 h5 1.3 h6 1.8
ZebFront interface layer for efficient behaviour development

- Preprocessor, using building bricks like elasticity, flow, etc...
- Use a macrolanguage, with a limited number of keywords like Coefs, StrainPart, derivative, implicit, etc...
- Generate C++ code
Explicit model implementation - ZebFront

```plaintext
@Class NORTON_BEHAVIOR : BASIC_NL_BEHAVIOR
{
    @Name norton;
    @SubClass ELASTICITY elasticity;
    @Coefs K, n;
    @tVarInt eel;
    @sVarInt evcum;
};

@StrainPart {
    sig = *elasticity*eel;
    m_tg_matrix = *elasticity;
}

@Derivative {
    TENSOR2 sprime, norm;
    double J;
    sig = *elasticity*eel;
    sprime = deviator(sig);
    J = sqrt(1.5*(sprime|sprime));
    devcum = pow(J/K, n);
    norm = sprime*(1.5/J);
    deel = deto-dvcum*norm;
}
```

Nom du comportement
Objet matrice d'élasticité
Coefficients de Norton
Variable interne tensorielle : $\varepsilon_e$
Variable interne scalaire : $p$
Calcul de la contrainte après intégration
$\sigma = \frac{E \varepsilon_e}{1-\nu}$
Matrice tangente approchée (RK !)
Calcul du vecteur dérivé $\dot{\mathbf{Y}}$
Calcul du déviateur $\dot{\sigma}$
Calcul du deuxième invariant
Fluage de Norton : $\dot{\rho} = (\frac{J}{K})^n$
Direction de l'écoulement
Déformation élastique
Implicit model implementation - ZebFront

```c
@CalcGradF {
    ELASTICITY& E=*elasticity;
    sig = E*eel;
    f_vec_eel -= deto;
    TENSOR2 sigeff = deviator(sig);
    double J = sqrt(1.5*(sigeff|sigeff));
    if (J>(double)0.0) {
        TENSOR2 norm = sigeff*(1.5/J);
        f_vec_eel += norm*devcum;
        f_vec_evcum -= dt*pow(J/K,n);
        SMATRIX dn_ds = unit32;
        dn_ds -= norm ^ norm;
        dn_ds *= theta*devcum/J;
        deel_deel += dn_ds *E;
        deel_devcum += norm;
        double dv_df = tdt*n*pow(J/K,n-1)/K;
        TENSOR2 df_fs = dv_df*norm;
        devcum_deel -= df_fs*E;
    }
}
```

Intégration implicite

\[ \sigma = E \varepsilon \]

\[ \text{Re} = \Delta \varepsilon - \Delta \zeta \]

Déviateur \( \sigma' \)

Deuxième invariant

Si on a plastifié

Direction de l’écoulement \( n \)

\[ \text{Re} = \Delta \varepsilon - \Delta \varepsilon + \Delta p n \]

\[ \Delta p = \left( \frac{f}{K} \right)^n \Delta t \]

\[ \frac{\partial \text{Re}}{\partial \Delta \varepsilon} \]

\[ \frac{\partial \text{Re}}{\partial \Delta p} \]

\[ \frac{\partial \Delta p}{\partial \Delta \varepsilon} \]

\[ \frac{\partial \Delta p}{\partial \Delta p} = 1 \]
Multimat capabilities

- Use homogenization rules
  - Localization rules
  - Local constitutive equations (possibly multimat)

- Macroscopic level (0)
  - ***behavior mori_tanaka
  - **material 0.65 matrice *file matrice.mat **material 0.35
  - fibre *file elas.mat *rotation
  - x1 0.2 0.3 0.4 x2 0.7 0.1 -0.3
  - ***return

- Material at level (1) to be defined
Multimat capabilities

Level 1
matrice.mat
***behavior berveiller_zaoui **mu 75000. **nu 0.3 **material 0.50 austenite *file austenite.mat
**material 0.50 ferrite *file ferrite.mat ***return
fibre.mat
***behavior linear_elastic
**elasticity orthotropic y1111 100000. y2222 120000. ... y3131 90000. ***return

Level 2
austenite.mat
***behavior gen_evp **elasticity isotropic young 260000. poisson 0.3 **potential gen_evp ep *flow plasticity *isotropic constant R0 130. ***return
ferrite.mat
***behavior gen_evp ... ***return
References


*Nonlinear Finite Elements for Continua and Structures.*

*Mecanique non–linéaire des matériaux. Hermes.*


*Une présentation de la méthode des éléments finis. Maloine.*


*Computational Inelasticity. Springer Verlag.*