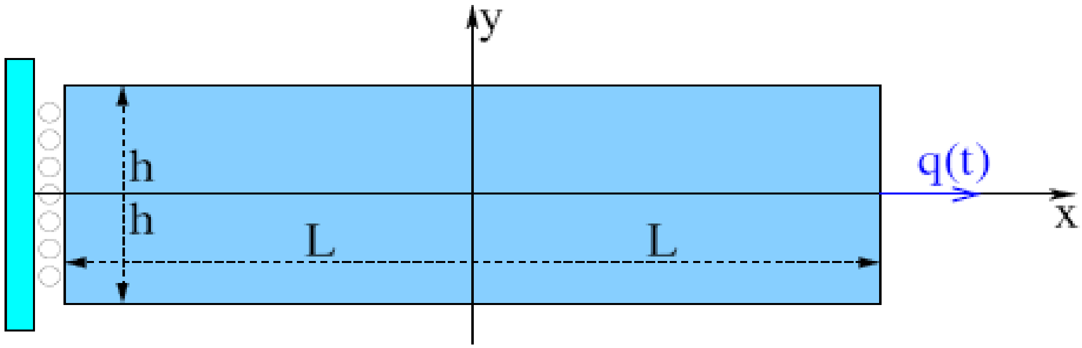
TD5: Exercices

Ex 1 - Introduce a Newton type algorithm in strip_plast.m

Ex 2 - Introduce modified Newton algorithm in plast_T.m (with constant operator)

Ex 1 : *strip_plast.m*



$u_x(L,y) = q(t)$ $\quad u_x(-L,y) = 0$ $\quad T_x(x,\pm h) = 0$

$T_y(L,y) = 0$ $\quad T_y(-L,y) = 0$ $\quad T_y(x,\pm h) = 0$

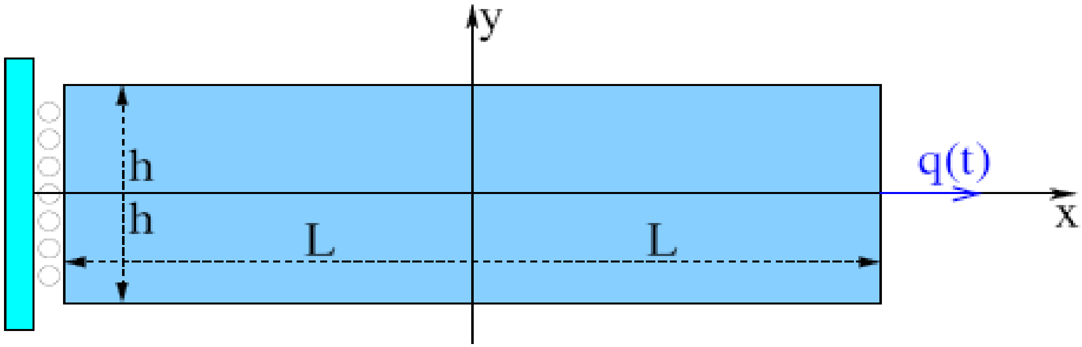
Material: homogeneous, elastic, perfectly plastic (R'(p)=0)

Exact solution available:

+ plane strain
+ homogeneous solution
  - ε_{xx}, ε_{yy} (other 0)
  - σ_{xx}, σ_{zz} (other 0)

$\varepsilon_{xx} = q/2L$; $\quad \varepsilon_{xy} = 0$; $\quad \sigma_{yy} = 0$

Ex 1 : *strip_plast.m*



$\varepsilon_{xx} = q/2L$; $\varepsilon_{xy} = 0$

+ $\underline{\underline{\sigma}}_n$, p_n, $\underline{\underline{\varepsilon}}_n$: known

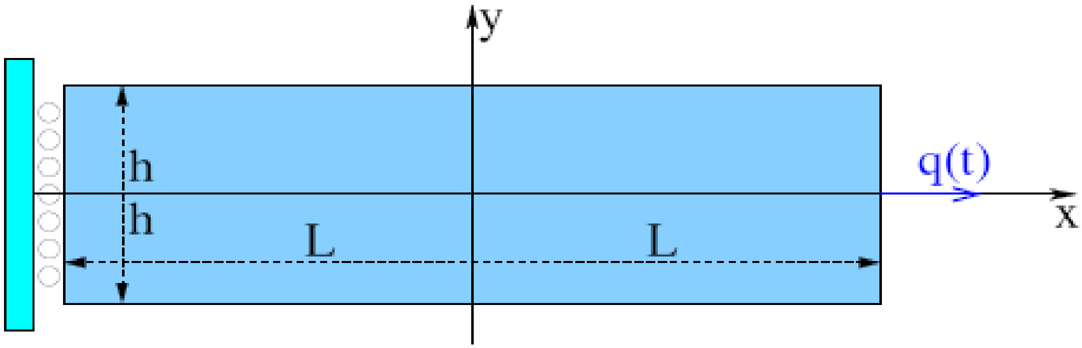
+ New loading increment : Δq ===> $\Delta\varepsilon_{xx} = \Delta q/2L$

+ Compute $\underline{\underline{\sigma}}_{n+1}$, p_{n+1}, $\underline{\underline{\varepsilon}}_{n+1}$?

$$\sigma_{yy} = 0 ==> \delta\varepsilon_{n,yy}^{(0)} = -\delta\varepsilon_{n,xx}^{(0)} A(1,2)/A(2,2)$$

$$A = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & (1-2\nu)/2 \end{bmatrix}$$

Ex 1 : *strip_plast.m*



$\varepsilon_{xx} = q/2L$; $\varepsilon_{xy} = 0$

Choice:

$$\sigma_{yy} = 0 ==> \delta\varepsilon^{(0)}_{n,yy} = -\delta\varepsilon^{(0)}_{n,xx} A(1,2) / A(2,2)$$

Radial return algo == > $\underline{\underline{\sigma}}^{(0)}_{n+1} = \mathcal{F}(\Delta\underline{\underline{\varepsilon}}^{(0)}_n, \mathcal{S}_n)$

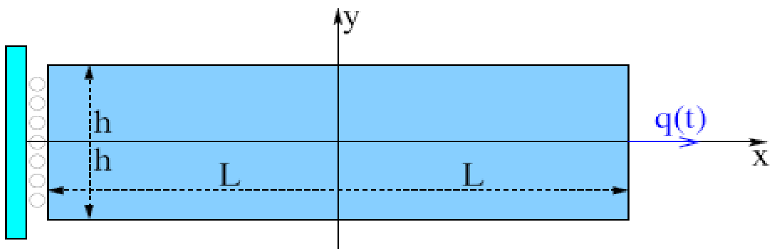
A priori: $\sigma^{(0)}_{yy} \neq 0$

$$\Delta\underline{\underline{\varepsilon}}^{(1)}_n = \Delta\underline{\underline{\varepsilon}}^{(0)}_n + \delta\underline{\underline{\varepsilon}}^{(0)}_n \quad (\delta\varepsilon^{(0)}_{n;xx} = 0)$$

$$\delta\varepsilon^{(0)}_{n;yy} \text{ such as } \left(\underline{\underline{\sigma}}^{(0)}_{n+1} + A : \delta\underline{\underline{\varepsilon}}^{(0)}_n\right)_{yy} = 0$$

$$==> \delta\varepsilon^{(0)}_{n,yy} = -\sigma^{(0)}_{n+1,yy} / A(2,2)$$

Ex 1 : *strip_plast.m*



Algorithm (strip_plast.m)

$\underline{\underline{\sigma}}_n, \underline{\underline{\varepsilon}}_n, p_n$ known

Temporal discretization (loading increment): Δq

find $\underline{\underline{\sigma}}_{n+1}, \underline{\underline{\varepsilon}}_{n+1} = \underline{\underline{\varepsilon}}_n + \Delta\underline{\underline{\varepsilon}}^n, p_{n+1} = p_n + \Delta p_n$

Initialization

$\Delta\underline{\underline{\varepsilon}}^{n,0} = \delta\underline{\underline{\varepsilon}}^{n,0}$

$\delta\varepsilon_{xx}^{n,0} = \Delta q / 2L$ (imposed)

$\delta\varepsilon_{yy}^{n,0} = -\delta\varepsilon_{xx}^{n,0} A(1,2)/A(2,2)$ such as $(\underline{\underline{\sigma}}^0)_{yy} = (A : \delta\underline{\underline{\varepsilon}}^{n,0})_{yy} = 0$ (*choice*)

Radial return algo ==> $\underline{\underline{\sigma}}_{n+1}^{(0)} = \mathcal{F}(\Delta\underline{\underline{\varepsilon}}_n^{(0)}, \mathcal{S}_n)$

while $\sigma_{yy}^{n+1,k} \neq 0$

$\Delta\underline{\underline{\varepsilon}}^{n,k+1} = \Delta\underline{\underline{\varepsilon}}^{n,k} + \delta\underline{\underline{\varepsilon}}^{n,k}$

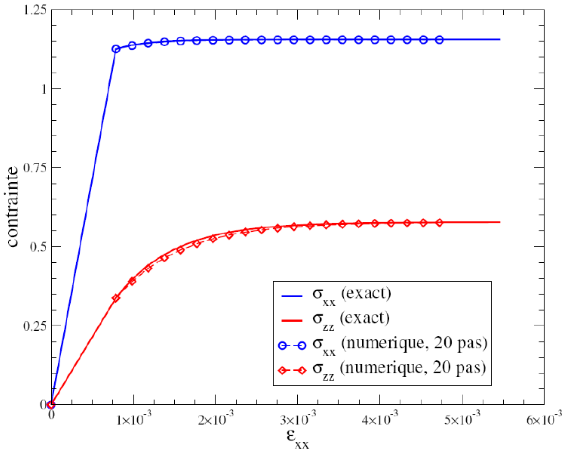
$\delta\varepsilon_{xx}^{n,k} = 0$ (imposed)

$\delta\varepsilon_{yy}^{n,k} = -(\sigma_{yy}^{n+1,k} + A(1,2)\delta\varepsilon_{xx}^{n,k})/A(2,2) = -\sigma_{yy}^{n+1,k}/A(2,2)$

such as $(\underline{\underline{\sigma}}^{n+1,k} + A : \delta\underline{\underline{\varepsilon}}^{n,k})_{yy} = 0$ (*choice*)

Radial return algo ==> $\underline{\underline{\sigma}}_{n+1}^{(k)} = \mathcal{F}(\Delta\underline{\underline{\varepsilon}}_n^{(k)}, \mathcal{S}_n)$

Ex 1 : *strip_plast.m*



Example : Strip_plast.m

```
L=1;
E=100;
nu=.1;
sigma0=1;
H=0;
q=[0:.002:.1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Pre-processor
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

numstep=length(q);
p=0;
sigma=zeros(4,1);
sigma_old=sigma;
output=zeros(numstep,2);
toll=1.d-4;
```

$$[A] = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & (1-2\nu)/2 \end{bmatrix}$$

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% History analysis
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for istep=2:numstep,
  Dq=q(istep)-q(istep-1);
  iter=0;
  resid=1;
  Dp=0;
  Deps=zeros(3,1);
  while resid > toll,
    iter=iter+1;
    if iter==1
      Deps=Dq/(2*L)*[1 -nu/(1-nu) 0]';
    else
      Depsyy=-sigma_new(2)*(1+nu)* ...
                        (1-2*nu)/(E*(1-nu));
      Deps=Deps+[0 Depsyy 0]';
    end
    [Dp,sigma_new]=RR_VonMises(E,nu, ...
                   H,sigma0,sigma,p,Deps);
    resid=abs(sigma_new(2));
  end
  p=p+Dp;
  sigma=sigma_new;
  output(istep,:)=[sigma(1) sigma(3)];
end
```

TD5 Exercice 1

Exercice 1 :

- Introduce a Newton type algorithm in strip_plast.m
- Compare with current version

Ex 2 : Elastoplasticity – Global aspect

Hole-plast.inp

hole.msh

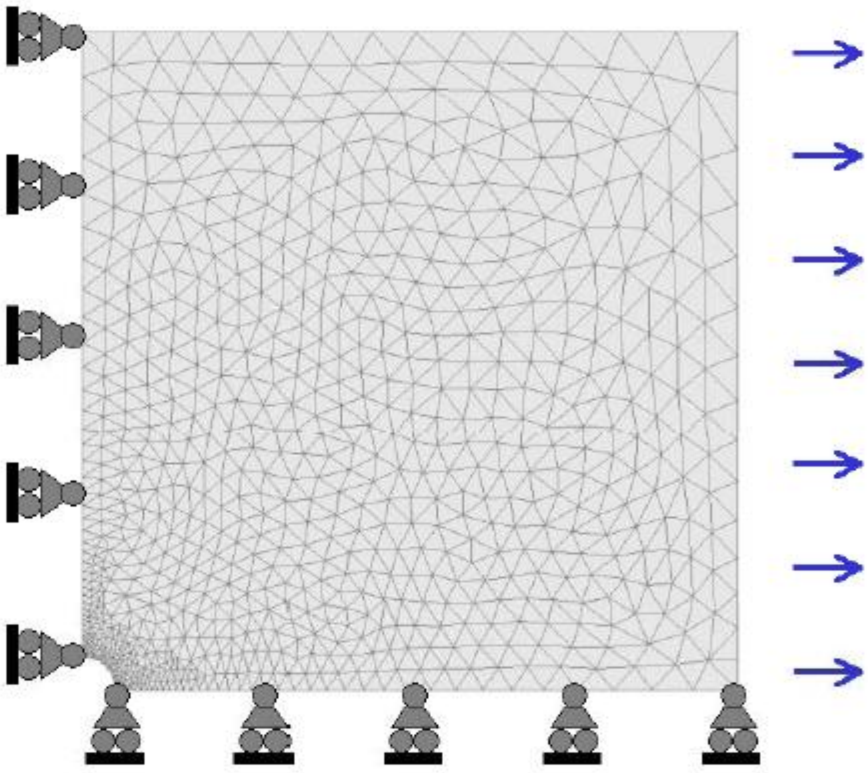
***ANALYSIS**
STATIC,TYPE=PLAST
1. 2. 3. 4. 5. 6. 7. 8.
**

*MATERIAL,TYPE=ISOTROPIC
YOUNG=1
POISSON=.3
HARDENING=0.
SIGMA0=0.88
**

*SOLID
ELSET=4
**

*DBC
ELSET=1,DIR=2,VAL=0.
ELSET=3,DIR=1,VAL=0.
ELSET=2,DIR=1,VAL=1.
**

*ENDFILE



$$\underline{T}^D(\underline{x},t) = \lambda(t)\underline{T}^D(\underline{x})$$

$$\underline{u}^D(\underline{x},t) = \lambda(t)\underline{u}^D(\underline{x})$$

$$f(\underline{\underline{\sigma}}, p) = \sigma^{eq} - R(p) \quad ; \quad \mathrm{R(p)} = \sigma_0$$

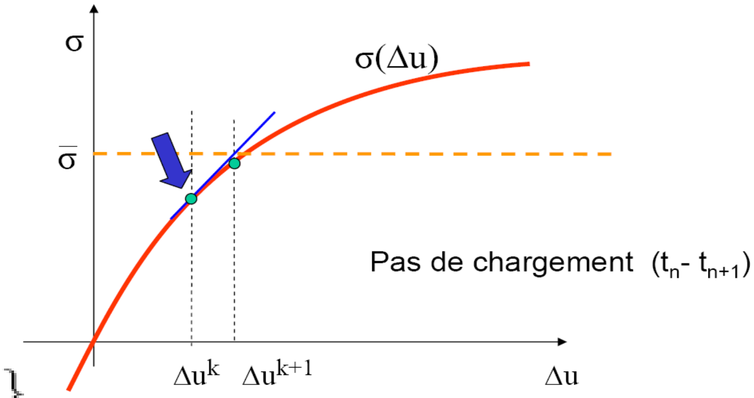
Ex 2 : Elastoplasticity – Global aspect

To do:

Knowing $\mathcal{S}_n = \{\underline{u}_n, \underline{\underline{\varepsilon}}_n, \underline{\underline{\varepsilon}}^{\mathrm{P}}_n, \underline{\underline{\sigma}}_n\}$

and loading $(\underline{f}_{n+1}, \underline{u}^{\mathrm{D}}_{n+1}, \underline{T}^{\mathrm{D}}_{n+1})$ at instant $t=t_{n+1}$

compute $\mathcal{S}_{n+1} = \{\underline{u}_{n+1}, \underline{\underline{\varepsilon}}_{n+1}, \underline{\underline{\varepsilon}}^{\mathrm{P}}_{n+1}, \underline{\underline{\sigma}}_{n+1}\}$



Find $\underline{u}_{n+1} \in \mathcal{C}(\underline{u}^{\mathrm{DD}}_{n+1}), \quad \mathcal{R}(\underline{u}_{n+1}; \underline{w}, \mathcal{S}_n) = 0 \qquad \forall \underline{w} \in \mathcal{C}(\underline{0})$

$$\mathcal{R}(\underline{u}_{n+1}; \underline{w}, \mathcal{S}_n) = \int_\Omega \mathcal{F}(\underline{u}_{n+1}; \mathcal{S}_n) : \underline{\underline{\varepsilon}}[\underline{w}] \, \mathrm{d}V - \int_\Omega \rho \underline{f}_{n+1} . \underline{w} \, \mathrm{d}V - \int_{S_T} \underline{T}^{\mathrm{D}}_{n+1} . \underline{w} \, \mathrm{d}S.$$

== > Newton type algo (TD5)

with $\underline{\underline{\sigma}}_{n+1} = \mathcal{F}(\Delta \underline{\underline{\varepsilon}}_n; \mathcal{S}_n)$

== > Radial return algorithm (TD4)

$$\underline{\underline{\sigma}}_{n+1} = \underline{\underline{\sigma}}_n + A : \Delta \underline{\underline{\varepsilon}}_n - 2\mu \, \Delta \underline{\underline{\varepsilon}}^p_n$$

Ex 2 : Newton type algorithm

Newton type algorithm: $\Delta \underline{u}_n^{(k)} = \Delta \underline{u}_n^{(k-1)} + \delta \underline{u}_n^{(k)}$

$$\mathcal{R}(\Delta \underline{u}_n; \underline{w}, \mathcal{S}_n) = \int_\Omega \mathcal{F}(\underline{\underline{\varepsilon}}[\Delta \underline{u}_n]; \mathcal{S}_n) : \underline{\underline{\varepsilon}}[\underline{w}] \, \mathrm{d}V - \int_\Omega \rho \underline{f}_{n+1} . \underline{w} \, \mathrm{d}V - \int_{S_T} \underline{T}^{\mathrm{D}}_{n+1} . \underline{w} \, \mathrm{d}S.$$

$$\underline{\underline{\sigma}}_{n+1}^{(k+1)} = F(\underline{\underline{\varepsilon}}(\Delta \underline{u}_n^{(k)}); S_n)$$

Linearization:

$$\underline{\underline{\sigma}}_{n+1}^{(k+1)} = \underline{\underline{\sigma}}_{n+1}^{(k)} + A^{EP} : \delta \underline{\underline{\varepsilon}}^{(k)} + o(\delta \underline{\underline{\varepsilon}}^{(k)})$$

Local tangent operator:

$$\mathcal{A}^{\mathrm{EP}}(\Delta \underline{\underline{\varepsilon}}_n^{(k)}; \mathcal{S}_n) = \frac{\partial \underline{\underline{\sigma}}_{n+1}}{\partial \Delta \underline{\underline{\varepsilon}}_n}(\Delta \underline{\underline{\varepsilon}}_n^{(k)}; \mathcal{S}_n)$$

Solve linear problem:

$$\mathcal{R}(\Delta \underline{u}_n^{(k)}; \underline{w}, \mathcal{S}_n) + \langle \mathcal{R}'(\Delta \underline{u}_n^{(k)}; \underline{w}, \mathcal{S}_n) \, , \, \delta \underline{u}_n^{(k)} \rangle = 0$$

$$\langle \mathcal{R}'(\underline{u}_{n+1}^{(k)}; \underline{w}, \mathcal{S}_n) \, , \, \delta \underline{u}_n^{(k)} \rangle = \int_\Omega \underline{\underline{\varepsilon}}[\delta \underline{u}_n^{(k)}] : \mathcal{A}^{\mathrm{EP}}(\Delta \underline{\underline{\varepsilon}}_n^{(k)}; \mathcal{S}_n) : \underline{\underline{\varepsilon}}[\underline{w}] \, \mathrm{d}V$$

Assembly

Ex 2 : Tangent Matrix

RR_VonMisesTM.m

$$\langle \mathcal{R}'(\underline{u}_{n+1}^{(k)}; \underline{w}, \mathcal{S}_n), \delta \underline{u}_n^{(k)} \rangle = \int_\Omega \underline{\underline{\varepsilon}}[\delta \underline{u}_n^{(k)}] : \boldsymbol{\mathcal{A}}^{\mathrm{EP}}(\Delta \underline{\underline{\varepsilon}}_n^{(k)}; \mathcal{S}_n) : \underline{\underline{\varepsilon}}[\underline{w}] \, \mathrm{d}V$$

$$\boldsymbol{\mathcal{A}}^{\mathrm{EP}}(\Delta \underline{\underline{\varepsilon}}_n, \mathcal{S}_n) = \begin{cases} \boldsymbol{\mathcal{A}} & \text{si } f_{n+1}^{\mathrm{elas}} < 0 \\ \boldsymbol{\mathcal{A}} - \boldsymbol{\mathcal{D}}(\Delta \underline{\underline{\varepsilon}}_n^{(k)}; \mathcal{S}_n) & \text{si } f_{n+1}^{\mathrm{elas}} > 0 \end{cases}$$

$$\boldsymbol{\mathcal{D}}(\Delta \underline{\underline{\varepsilon}}_n; \mathcal{S}_n) = 3\mu(\gamma - \beta) \left(\frac{\underline{\underline{s}}_{n+1}^{\mathrm{elas}}}{\sigma_{n+1}^{\mathrm{elas,eq}}} \otimes \frac{\underline{\underline{s}}_{n+1}^{\mathrm{elas}}}{\sigma_{n+1}^{\mathrm{elas,eq}}} \right) + 2\mu\beta\boldsymbol{\mathcal{K}}$$

$$\beta = \frac{3\mu\Delta p_n}{\sigma_{n+1}^{\mathrm{elas,eq}}} = 1 - \frac{R_{n+1}}{\sigma_{n+1}^{\mathrm{elas,eq}}} \qquad \gamma = \frac{3\mu}{3\mu + R'_{n+1}}$$

$$f(\underline{\underline{\sigma}}, p) = \sigma^{eq} - R(p)$$

```
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
% Radial Return for Von Mises Linear Isotropic Hardening with tangent matrix
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function [AEP,Dp,sigma_new]=RR_VonMisesTM(A,E,nu,H,sigma0,sigma,p,Deps)
```

Ex 2 : Tangent matrix

Von Mises criterion (linear isotropic hardening)

RR_VonMisesTM.m

$$\mathcal{D}(\Delta\underline{\underline{\varepsilon}}_n;\mathcal{S}_n) = 3\mu(\gamma-\beta)\left(\frac{\underline{\underline{s}}^{\text{elas}}_{n+1}}{\sigma^{\text{elas,eq}}_{n+1}} \otimes \frac{\underline{\underline{s}}^{\text{elas}}_{n+1}}{\sigma^{\text{elas,eq}}_{n+1}}\right) + 2\mu\beta\mathcal{K}$$

$$\beta = \frac{3\mu\Delta p_n}{\sigma^{\text{elas,eq}}_{n+1}} = 1 - \frac{R_{n+1}}{\sigma^{\text{elas,eq}}_{n+1}} \qquad \gamma = \frac{3\mu}{3\mu + R'_{n+1}}$$

Plane strain: $\Delta\varepsilon_{33} = 0$

$$\Delta\underline{\underline{e}}_n = \left(\Delta\underline{\underline{\varepsilon}}_n - \frac{1}{3}\text{tr}\Delta\underline{\underline{\varepsilon}}_n\underline{\underline{1}}\right) = \mathcal{K} : \Delta\underline{\underline{\varepsilon}}_n$$

$$\Delta\underline{\underline{\varepsilon}} \rightarrow \{\Delta\varepsilon_{11}, \Delta\varepsilon_{22}, \Delta\varepsilon_{33}, 2\Delta\varepsilon_{12}\}$$

$$\mathcal{K} = \frac{1}{3}\begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 2 & 0 \\ 0 & 0 & 0 & 3/2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1/2 \end{bmatrix} - \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
% Radial Return for Von Mises Linear Isotropic Hardening with tangent m
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function [AEP,Dp,sigma_new]=RR_VonMisesTM(A,E,nu,H,sigma0,sigma,p,Deps)

mu=E/(2*(1+nu));
kappa=E/(3*(1-2*nu));
vl=[1 1 1 0]';
MK=1/3*[2 -1 -1 0; -1 2 -1 0; ...
        -1 -1 2 0; 0  0  0 3/2];
```

Ex 2 : Tangent matrix

RR_VonMisesTM.m

$$\boldsymbol{\mathcal{A}}^{\mathrm{EP}}(\Delta\underline{\underline{\varepsilon}}_n, \mathcal{S}_n) = \begin{cases} \boldsymbol{\mathcal{A}} & \text{si } f_{n+1}^{\mathrm{elas}} < 0 \\ \boldsymbol{\mathcal{A}} - \boldsymbol{\mathcal{D}}(\Delta\underline{\underline{\varepsilon}}_n^{(k)}; \mathcal{S}_n) & \text{si } f_{n+1}^{\mathrm{elas}} > 0 \end{cases}$$

$$\boldsymbol{\mathcal{D}}(\Delta\underline{\underline{\varepsilon}}_n; \mathcal{S}_n) = 3\mu(\gamma - \beta)\left(\frac{\underline{\underline{s}}_{n+1}^{\mathrm{elas}}}{\sigma_{n+1}^{\mathrm{elas,eq}}} \otimes \frac{\underline{\underline{s}}_{n+1}^{\mathrm{elas}}}{\sigma_{n+1}^{\mathrm{elas,eq}}}\right) + 2\mu\beta\boldsymbol{\mathcal{K}}$$

$$\beta = \frac{3\mu\Delta p_n}{\sigma_{n+1}^{\mathrm{elas,eq}}} = 1 - \frac{R_{n+1}}{\sigma_{n+1}^{\mathrm{elas,eq}}} \qquad \gamma = \frac{3\mu}{3\mu + R'_{n+1}}$$

$$(\underline{\underline{s}}^{\mathrm{elas}} \otimes \underline{\underline{s}}^{\mathrm{elas}}) : \Delta\underline{\underline{\varepsilon}} = \underline{\underline{s}}^{\mathrm{elas}}\left(\underline{\underline{s}}^{\mathrm{elas}} : \Delta\underline{\underline{\varepsilon}}\right)$$

$$= \{s^{\mathrm{elas}}\}\left(\{s^{\mathrm{elas}}\}^T\{\Delta\varepsilon\}\right) = \{s^{\mathrm{elas}}\}\{s^{\mathrm{elas}}\}^T\{\Delta\varepsilon\}$$

```
if(f_elas>0)                                % if plastic process
  Dp=f_elas/(3*mu+H);                       % increment of plastic eq. strain
  sigeq_new=sigeq_elas-3*mu*Dp;             % new equivalent stress
  n_elas=s_elas/sigeq_elas;
  Depsp=3/2*Dp*n_elas;                      % increment over step of plastic defs
  sigma_new=sigma_elas-2*mu*Depsp;
  beta=3*mu*Dp/sigeq_elas;                  % coefficients gamma and beta
  gamma=3*mu/(3*mu+H);
  D=3*mu*(gamma-beta)*n_elas*n_elas'+...    % D matrix
    2*mu*beta*MK;
  AEP=A-[D(1:2,1:2) D(1:2,4); ...           % selects components for plane strain
              D(4,1:2) D(4,4)];
else                                        % elseif elastic process
  Dp=0;
  sigma_new=sigma_elas;                     % new total stress
  AEP=A;
end
```

Ex 2 : plast_T.m

For one load step :

$$\text{trouver } \Delta \underline{u}_n \in \mathcal{C}(\Delta \underline{u}_n^{\mathrm{D}}), \quad \mathcal{R}(\Delta \underline{u}_n; \underline{w}, \mathcal{S}_n) = 0 \qquad \forall \underline{w} \in \mathcal{C}(\underline{0})$$

$$\mathcal{R}(\Delta\underline{u}_n;\underline{w},\mathcal{S}_n) = \underbrace{\int_\Omega \mathcal{F}(\underline{\underline{\varepsilon}}[\Delta\underline{u}_n];\mathcal{S}_n):\underline{\underline{\varepsilon}}[\underline{w}]\,\mathrm{d}V}_{-\text{Fint}} \underbrace{- \int_\Omega \rho \underline{f}_{n+1}.\underline{w}\,\mathrm{d}V - \int_{S_T} \underline{T}^{\mathrm{D}}_{n+1}.\underline{w}\,\mathrm{d}S}_{-\text{Fext}}$$

-Fint **-Fext**

Assembly of elemental contribution == >LcoorT3.m

Newton type algorithm

$$\Delta\underline{u}_n^{(k+1)} = \Delta\underline{u}_n^{(k)} + \delta\underline{u}_n^{(k)}$$

$$\mathcal{R}(\Delta\underline{u}_n^{(k)};\underline{w},\mathcal{S}_n) + \langle \mathcal{R}'(\Delta\underline{u}_n^{(k)};\underline{w},\mathcal{S}_n)\,,\ \delta\underline{u}_n^{(k)} \rangle = 0$$

$$\{R\} + [KEP] * \{\delta U\} = 0$$

$$\langle \mathcal{R}'(\underline{u}_{n+1}^{(k)};\underline{w},\mathcal{S}_n)\,,\ \delta\underline{u}_n^{(k)} \rangle = \int_\Omega \underline{\underline{\varepsilon}}[\delta\underline{u}_n^{(k)}] : \boldsymbol{\mathcal{A}}^{\mathrm{EP}}(\Delta\underline{\underline{\varepsilon}}_n^{(k)};\mathcal{S}_n) : \underline{\underline{\varepsilon}}[\underline{w}]\,\mathrm{d}V$$

$$[KEP]$$

$$\delta\underline{u}_n^{(1)} = \delta\underline{u}_n^{(1,0)} + \Delta\underline{u}_n^{(D)} \quad ; \quad \delta\underline{u}_n^{(1,0)} \in C(\underline{0});\ \Delta\underline{u}_n^{(D)} \in C(\Delta\underline{u}_n^{D})$$

$$\Delta\underline{u}_n^{(k)} = \Delta\underline{u}_n^{(k-1)} + \delta\underline{u}_n^{(k)} \quad ; \quad \delta\underline{u}_n^{(k)} \in C(\underline{0})$$

Ex 2 : plast_T.m

$$\mathcal{R}(\Delta\underline{u}_n;\underline{w},\mathcal{S}_n)=\underbrace{\int \mathcal{F}(\underline{\varepsilon}[\Delta\underline{u}_n];\mathcal{S}_n):\underline{\varepsilon}[\underline{w}]\,\mathrm{d}V}_{\text{-Fint}}\underbrace{-\int_\Omega \rho\underline{f}_{n+1}.\underline{w}\,\mathrm{d}V-\int_{S_T}\underline{T}^{\mathrm{D}}_{n+1}.\underline{w}\,\mathrm{d}S}_{\text{-Fext}}$$

```
imposed_disp=displ;
for step=1:analysis.numstep,                     % loop over all load steps
  Dlambda=  analysis.history(step+1)
               - analysis.history(step);          % delta of load multiplier applied in step
  Ddisp=Dlambda*imposed_disp;                     % initialisation of displ. incr. in step

  R=-Fext*analysis.history(step+1)-Fint- ...
                        FDu*Dlambda;              % initialisation of residuum vector
  resid=sqrt(R'*R);

%  NR procedure for one single step
  iter=0;
  toll=1.d-4;
  while resid > toll,
   iter=iter+1;

    ...    Newton iteration ...

   R=-Fext*analysis.history(step+1)-Fint;         % residuum vector
   resid=sqrt(R'*R);
 end                                              % end of iterations in one time step
  pG=pG+DpG;                                      % updates equivalent plastic strains
  stressG=stressG_new;                            % updates stress at Gauss points
  displ=displ+Ddisp;                              % updates displacements
end                                               % end loop over time steps
```

$$\delta\underline{u}_n^{(1)}=\delta\underline{u}_n^{(1,0)}+\Delta\underline{u}_n^{(D)}$$

$$\delta\underline{u}_n^{(1,0)}\in C(\underline{0})$$

$$\Delta\underline{u}_n^{(D)}\in C(\Delta\underline{u}_n^{D})$$

$$\Delta\underline{u}_n^{(k)}=\Delta\underline{u}_n^{(k-1)}+\delta\underline{u}_n^{(k)}$$

$$\delta\underline{u}_n^{(k)}\in C(\underline{0})$$

Ex 2 : plast_T.m

Newton type algorithm

$$\mathcal{R}(\Delta \underline{u}_n; \underline{w}, \mathcal{S}_n) = \int_\Omega \mathcal{F}(\underline{\underline{\varepsilon}}[\Delta \underline{u}_n]; \mathcal{S}_n) : \underline{\underline{\varepsilon}}[\underline{w}] \, \mathrm{d}V - \int_\Omega \rho \underline{f}_{n+1} . \underline{w} \, \mathrm{d}V - \int_{S_T} \underline{T}^{\mathrm{D}}_{n+1} . \underline{w} \, \mathrm{d}S$$

```
iter=0;
 toll=1.d-4;
 while resid > toll,
  iter=iter+1;

% global elastic prediction

  DDU = - KEP\R;                          % prediction with consistent KEP
  ir=find(dof>0);                         % finds non-zero entries in "dof"
  Ddisp(ir)=Ddisp(ir)+DDu(dof(ir));       % updates displacements
```

Solve (linearized problem)

$$\mathcal{R}(\Delta \underline{u}_n^{(k)}; \underline{w}, \mathcal{S}_n) + \langle \mathcal{R}'(\Delta \underline{u}_n^{(k)}; \underline{w}, \mathcal{S}_n) \, , \, \delta \underline{u}_n^{(k)} \rangle = 0$$

$$\{R\} + [KEP] * \{DDU\} = 0$$

$$\Delta \underline{u}_n^{(k)} = \Delta \underline{u}_n^{(k-1)} + \delta \underline{u}_n^{(k)}$$ =DDU

```
% local plastic correction

  KEP(:,:)=0.d0;
  FDu(:)=0;                    % sets FDu to zero
  Fint(:)=0;                   % sets Fint to zero
  for e=1:analysis.NE,         % loop over elements
```

== > Assembly of elemental contributions < == LcoorT3 < == RRVonMisesTM

```
  end

 R=-Fext*analysis.history(step+1)-Fint;     % residuum vector
 resid=sqrt(R'*R);
end                                          % end of iterations in on
```

Ex 2 : plast_T.m: Assembly

$$\mathcal{R}(\Delta \underline{u}_n^{k+1}; \underline{w}, \mathcal{S}_n) = \int_\Omega \mathcal{F}(\underline{\underline{\varepsilon}}[\Delta \underline{u}_n^{k+1}]; \mathcal{S}_n) : \underline{\underline{\varepsilon}}[\underline{w}] \, dV - \int_\Omega \rho \underline{f}_{n+1} \cdot \underline{w} \, dV - \int_{S_T} \underline{T}^D_{n+1} \cdot \underline{w} \, dS$$

-Fint

KEP*δu

$$\mathcal{R}(\Delta \underline{u}_n^{(k)}; \underline{w}, \mathcal{S}_n) + \langle \mathcal{R}'(\Delta \underline{u}_n^{(k)}; \underline{w}, \mathcal{S}_n) \, , \, \delta \underline{u}_n^{(k)} \rangle = 0$$

```
for e=1:analysis.NE,                                % loop over elements
  nodes=connec(e,:);
  DUe=reshape(Ddisp(nodes,:)',[Dne,1]);             % increments of nodal displ.
  T=coor(nodes,:);                                  % creates element
  dofe=reshape(dof(nodes,:)',[1,Dne]);              % list of dof associated to element
  pe=find(dofe>0);                                  % gets position of unknown displ. compon.
  Ie=dofe(pe);                                      % gets valu
  [KEPe,Finte,stressG_new(e,:,:),DpG(e,:)]=...
     eval(['lcorr_T',analysis.Etag, ...
           '(DUe,T,pG(e,:),'...
           'stressG(e,:,:),materials)']);
  KEP(Ie,Ie)=KEP(Ie,Ie)+KEPe(pe,pe);                % matrix assemblage
  pe_UDe=find(dofe<0);                              % gets position of given displ. compon.
  Ie_UDe=-dofe(pe_UDe);                             % gets p
  UDe=imposed_disp(Ie_UDe)';                        % gets d
  FDu(Ie)=FDu(Ie)-KEPe(pe,pe_UDe)*UDe;              % rhs fr
  Fint(Ie)=Fint(Ie)+Finte(pe);                      % assembling vector of internal forces
end
```

Lcoor_T3: elemental contributions KEP_e & Fint_e

Standard assembly procedure KEP & Fint

Ex 2 : lcoor_T3.m : elemental contributions

```
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
%  Local plastic correction for T3
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function [KEPe,Finte,sigma_new,Dp]=lcorr_T3(DUe,T,p,sigma,materials)

 x11=T(1,1); x21=T(2,1); x31=T(3,1);
 x12=T(1,2); x22=T(2,2); x32=T(3,2);
 S=.5*((x21-x11)*(x32-x12)-...
        (x31-x11)*(x22-x12));
 Be=[x22-x32,0,x32-x12,0,x12-x22,0;
     0,x31-x21,0,x11-x31,0,x21-x11;
     x31-x21,x22-x32,x11-x31, ...
     x32-x12,x21-x11,x12-x22]/(2*S);
 Deps=Be*DUe;
 [AEP,Dp,sigma_new]=RR_VonMisesTM...
          (materials.A,materials.young,...
           materials.poisson,materials.H,...
           materials.sigma0,sigma(1,1:4)',...
           p,Deps);
 Finte=-S*Be'*sigma_new([1:2 4]);
 KEPe=S*Be'*AEP*Be;
```

Compute $\Delta\varepsilon=\varepsilon(\Delta U)$

RR_VonMisesTM: sigma_e & A^{EP}_e

Fint_e

$$\mathcal{R}(\Delta\underline{u}_n;\underline{w},\mathcal{S}_n)=\int_\Omega \mathcal{F}(\underline{\underline{\varepsilon}}[\Delta\underline{u}_n];\mathcal{S}_n):\underline{\underline{\varepsilon}}[\underline{w}]\,\mathrm{d}V-\int_\Omega \rho\underline{f}_{n+1}\cdot\underline{w}\,\mathrm{d}V-\int_{S_T}\underline{T}^{\mathrm{D}}_{n+1}\cdot\underline{w}\,\mathrm{d}S$$

KEP_e

$$\langle\mathcal{R}'(\underline{u}^{(k)}_{n+1};\underline{w},\mathcal{S}_n)\,,\,\delta\underline{u}^{(k)}_n\rangle=\int_\Omega \underline{\underline{\varepsilon}}[\delta\underline{u}^{(k)}_n]:\boldsymbol{\mathcal{A}}^{\mathrm{EP}}(\Delta\underline{\underline{\varepsilon}}^{(k)}_n;\mathcal{S}_n):\underline{\underline{\varepsilon}}[\underline{w}]\,\mathrm{d}V$$

TD5: Exercices

Exercie 2 :

- Introduce modified Newton algorithm in plast_T.m (with constant operator)
- Compare to the current version