Improving the robustness of implicit schemes using homotopy-based algorithms

MFront Users Meeting

[1] CEA, DES, IRESNE, DEC, SESC, LMPC, Cadarache, France

T. Helfer[1], Maxence Wangermez[1] (and many others!)

Outline

Homotopy methods to enhance the robustness of implicit schemes

Studied algorithms

First test case

Second test case

Conclusions

1 Homotopy methods to enhance the robustness of implicit schemes

Notations

- t denotes the time at the beginning of the time step.
- θ denotes a numerical parameter between 0 and 1.
- $\left.\vec{Y}\right|_{t}$ denotes the value of the state variables at the beginning of the time step.
- $\left.\vec{Y}\right|_{t+\theta\,\Delta\,t}$ denotes the value of the state variables at an intermediate time $t+\theta\,\Delta\,t$:

$$\left.\vec{Y}\right|_{t+\Delta\,t} = \left.\vec{Y}\right|_{t} + \theta\,\Delta\,\vec{Y}$$

- $\left.\vec{Y}\right|_{t+\Delta\,t}$ denotes the value of the state variables at the end of the time step:

$$\left.\vec{Y}\right|_{t+\Delta\,t} = \left.\vec{Y}\right|_{t} + \Delta\,\vec{Y}$$

Residual

- For time dependent behaviours, the evolution of the state variables $\vec{Y}$ is given by a system of ordinary differential equations:

$$\dot{\vec{Y}} = G\left(\vec{Y}, \vec{Z}\right)$$

where $\vec{Z}$ denotes a set of external state variables.

- Using the generalised mid-point rule, this equation can be discretised as:

$$\Delta\,\vec{Y} - G\left(\left.\vec{Y}\right|_{t} + \theta\,\Delta\,\vec{Y}\right)\,\Delta\,t = \vec{0}$$

or

$$\vec{F}\left(\Delta\,\vec{Y}\right) = \vec{0} \quad \text{with} \quad \vec{F}\left(\Delta\,\vec{Y}\right) = \Delta\,\vec{Y} - G\left(\left.\vec{Y}\right|_{t} + \theta\,\Delta\,\vec{Y}\right)\,\Delta\,t$$

- $\vec{F}$ is called the residual of the implicit system.
- For time independent behavious, the residual ensures that the material is on the yield surface(s) at the end of the time step.

Newton algorithm

- Let $\Delta\,\vec{Y}^{(n)}$ be the estimation at the n^{th} iteration of the Newton algorithm. $\Delta\,\vec{Y}^{(n+1)}$ is then given by:

  $$\Delta\,\vec{Y}^{(n+1)} = \Delta\,\vec{Y}^{(n)} + \delta\,\Delta\,\vec{Y}^{(n)} \quad \text{with} \quad \delta\,\Delta\,\vec{Y}^{(n)} = -J^{-1}\left(\Delta\,\vec{Y}^{(n)}\right)\,\vec{F}\left(\Delta\,\vec{Y}^{(n)}\right)$$

  where J is the jacobian matrix given by:

  $$J\left(\Delta\,\vec{Y}^{(n)}\right) = \frac{\partial \vec{F}}{\partial \Delta\,\vec{Y}}\left(\Delta\,\vec{Y}^{(n)}\right)$$

- In `MFront`, the jacobian matrix can be computed analytically or evaluated numerically using a centered finite difference scheme.

Globalization

- The greatest shortcoming of the Newton algorithm is that the initial estimation $\vec{Y}^{(0)}$ of the solution must be close enough from the solution:
  - This condition can be difficult to met in pratice.
- Some globalization algorithms are already available (see below):
  - Levenberg-Marquardt.
  - Powell's dog leg.
  - Hand-crafted line-search.
- The aim of this talk is to study two homotopy-inspired algorithms
- Note: in all cases, the consistent tangent operator can still be computed and its expression is unchanged.

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_{1}\left(\Delta\,\vec{Y}\right) = \vec{F}\left(\Delta\,\vec{Y}\right)$

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_1\left(\Delta\vec{Y}\right) = \vec{F}\left(\Delta\vec{Y}\right)$
    - $\vec{F}_0$ is trivial to solve.

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0,1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_1\left(\Delta\,\vec{Y}\right) = \vec{F}\left(\Delta\,\vec{Y}\right)$
    - $\vec{F}_0$ is trivial to solve.
- Homotopy algorithms thus introduced a continuous mapping $\Delta\,\vec{Y}_{\mu}$ which transforms $\Delta\,\vec{Y}_0$ (supposed trivial to compute) to a searched solution $\Delta\,\vec{Y}$

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_1\left(\Delta\,\vec{Y}\right) = \vec{F}\left(\Delta\,\vec{Y}\right)$
    - $\vec{F}_0$ is trivial to solve.
- Homotopy algorithms thus introduced a continuous mapping $\Delta\,\vec{Y}_{\mu}$ which transforms $\Delta\,\vec{Y}_0$ (supposed trivial to compute) to a searched solution $\Delta\,\vec{Y}$
- Two main ingredients must be choosen:

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_{1}\left(\Delta\,\vec{Y}\right) = \vec{F}\left(\Delta\,\vec{Y}\right)$
    - $\vec{F}_{0}$ is trivial to solve.
- Homotopy algorithms thus introduced a continuous mapping $\Delta\,\vec{Y}_{\mu}$ which transforms $\Delta\,\vec{Y}_{0}$ (supposed trivial to compute) to a searched solution $\Delta\,\vec{Y}$
- Two main ingredients must be choosen:
  - The expression of $\vec{F}_{\mu}$

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_{1}\left(\Delta\,\vec{Y}\right) = \vec{F}\left(\Delta\,\vec{Y}\right)$
    - $\vec{F}_{0}$ is trivial to solve.
- Homotopy algorithms thus introduced a continuous mapping $\Delta\,\vec{Y}_{\mu}$ which transforms $\Delta\,\vec{Y}_{0}$ (supposed trivial to compute) to a searched solution $\Delta\,\vec{Y}$
- Two main ingredients must be choosen:
  - The expression of $\vec{F}_{\mu}$
  - How to make μ evolve from 0 to 1.

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_1\left(\Delta\vec{Y}\right) = \vec{F}\left(\Delta\vec{Y}\right)$
    - $\vec{F}_0$ is trivial to solve.
- Homotopy algorithms thus introduced a continuous mapping $\Delta\vec{Y}_{\mu}$ which transforms $\Delta\vec{Y}_0$ (supposed trivial to compute) to a searched solution $\Delta\vec{Y}$
- Two main ingredients must be choosen:
  - The expression of $\vec{F}_{\mu}$
  - How to make μ evolve from 0 to 1.
- By virtue of the implicit function theorem, one have:

$$\frac{\partial\Delta\,\vec{Y}_{\mu}}{\partial\mu} = -\left(\frac{\partial F}{\partial\Delta\,\vec{Y}}\right)^{-1}\frac{\partial\vec{F}_{\mu}}{\partial\mu} = -J^{-1}\,\frac{\partial\vec{F}_{\mu}}{\partial\mu}$$

Homotopy algorithm

- The idea of homotopy algorithm is to introduce:
  - an homotopy parameter $\mu \in [0, 1]$
  - a modified residual $\vec{F}_{\mu}$ such that:
    - $\vec{F}_{1}\left(\Delta\,\vec{Y}\right) = \vec{F}\left(\Delta\,\vec{Y}\right)$
    - $\vec{F}_{0}$ is trivial to solve.
- Homotopy algorithms thus introduced a continuous mapping $\Delta\,\vec{Y}_{\mu}$ which transforms $\Delta\,\vec{Y}_{0}$ (supposed trivial to compute) to a searched solution $\Delta\,\vec{Y}$
- Two main ingredients must be choosen:
  - The expression of $\vec{F}_{\mu}$
  - How to make μ evolve from 0 to 1.
- By virtue of the implicit function theorem, one have:

$$\frac{\partial \Delta\,\vec{Y}_{\mu}}{\partial \mu} = -\left(\frac{\partial F}{\partial \Delta\,\vec{Y}}\right)^{-1}\frac{\partial \vec{F}_{\mu}}{\partial \mu} = -J^{-1}\,\frac{\partial \vec{F}_{\mu}}{\partial \mu}$$

- Hence, an explicit solution exists:

$$\Delta\,\vec{Y} = \Delta\,\vec{Y}\,(0) - \int_{0}^{1} J^{-1}\,\frac{\partial \vec{F}_{\mu}}{\partial \mu}\,\mathrm{d}\,\mu$$

Outline

2. Studied algorithms

First algorithm - I

- The θ parameter can be considered as the homotopy parameter:

$$\vec{F}\left(\Delta\,\vec{Y}\,(\theta)\,,\theta\right) = 0$$

First algorithm - I

- The θ parameter can be considered as the homotopy parameter:

$$\vec{F}\left(\Delta\,\vec{Y}\,(\theta)\,,\theta\right)=0$$

- For time dependent materials, $\Delta\,\vec{Y}_0$ is given by the explicit Euler approximation:

$$\Delta\,\vec{Y}_0 = G\left(\left.\vec{Y}\right|_t\right)\,\Delta\,t$$

First algorithm - I

- The θ parameter can be considered as the homotopy parameter:

$$\vec{F}\left(\Delta\,\vec{Y}\,(\theta)\,,\theta\right)=0$$

- For time dependent materials, $\Delta\,\vec{Y}_0$ is given by the explicit Euler approximation:

$$\Delta\,\vec{Y}_0=G\left(\left.\vec{Y}\right|_{t}\right)\,\Delta\,t$$

- Plasticity is a problem. For instance, if one consider an isotropic behaviour with linear hardening (see https://thelfer.github.io/tfel/web/IsotropicLinearHardeningPlasticity.html for details), the residual :

$$f_p=\frac{\left.\sigma_{eq}^{\mathrm{tr}}\right|_{t+\theta\,\Delta\,t}-\sigma_0-R\;\left.p\right|_{t}}{\left(3\;\left.\mu\right|_{t+\theta\,\Delta\,t}+R\right)\;\theta}$$

  does not depend on the increments of the state variables for $\theta=0$, i.e. the jacobian matrix is singular.

First algorithm - II

- $\dfrac{\partial \vec{F}}{\partial \theta}$ is not trivial to compute and would require developments on the user side (i.e. one can not simply swith from one algorithm to the other).

First algorithm - II

- $\frac{\partial \vec{F}}{\partial \theta}$ is not trivial to compute and would require developments on the user side (i.e. one can not simply swith from one algorithm to the other).
- It is very simple to code a **fall-back mechanism** in case of non convergence:

First algorithm - II

- $\dfrac{\partial \vec{F}}{\partial \theta}$ is not trivial to compute and would require developments on the user side (i.e. one can not simply swith from one algorithm to the other).
- It is very simple to code a **fall-back mechanism** in case of non convergence:
  - If the implicit algorithm does not converge, decrease θ and increase it after convergence. The key idea here is that $\Delta\,\vec{Y}_{\theta(n)}$ is a good approximation for $\Delta\,\vec{Y}_{\theta(n+1)}$

First algorithm - II

- $\dfrac{\partial \vec{F}}{\partial \theta}$ is not trivial to compute and would require developments on the user side (i.e. one can not simply swith from one algorithm to the other).
- It is very simple to code a **fall-back mechanism** in case of non convergence:
  - If the implicit algorithm does not converge, decrease θ and increase it after convergence. The key idea here is that $\Delta\,\vec{Y}_{\theta(n)}$ is a good approximation for $\Delta\,\vec{Y}_{\theta(n+1)}$
  - This fall-back mechanism is:

First algorithm - II

- $\frac{\partial \vec{F}}{\partial \theta}$ is not trivial to compute and would require developments on the user side (i.e. one can not simply swith from one algorithm to the other).
- It is very simple to code a **fall-back mechanism** in case of non convergence:
  - If the implicit algorithm does not converge, decrease θ and increase it after convergence. The key idea here is that $\Delta\,\vec{Y}_{\theta(n)}$ is a good approximation for $\Delta\,\vec{Y}_{\theta(n+1)}$
  - This fall-back mechanism is:
    - trivial to implement.

First algorithm - II

- $\dfrac{\partial \vec{F}}{\partial \theta}$ is not trivial to compute and would require developments on the user side (i.e. one can not simply swith from one algorithm to the other).
- It is very simple to code a **fall-back mechanism** in case of non convergence:
  - If the implicit algorithm does not converge, decrease θ and increase it after convergence. The key idea here is that $\Delta\,\vec{Y}_{\theta(n)}$ is a good approximation for $\Delta\,\vec{Y}_{\theta(n+1)}$
  - This fall-back mechanism is:
    - trivial to implement.
    - does not require any user defined extension.

First algorithm - II

- $\dfrac{\partial \vec{F}}{\partial \theta}$ is not trivial to compute and would require developments on the user side (i.e. one can not simply swith from one algorithm to the other).
- It is very simple to code a **fall-back mechanism** in case of non convergence:
  - If the implicit algorithm does not converge, decrease θ and increase it after convergence. The key idea here is that $\Delta\,\vec{Y}_{\theta(n)}$ is a good approximation for $\Delta\,\vec{Y}_{\theta(n+1)}$
  - This fall-back mechanism is:
    - trivial to implement.
    - does not require any user defined extension.
    - has almost zero cost when unused.

First algorithm - III

```
@AdditionalConvergenceChecks {
  if ((!converged) && (iter == iterMax)) {
    ++numberOfHomotopySubSteps;
    if (numberOfHomotopySubSteps != maximumNumberOfHomotopySubSteps) {
      zeros = lastZeros;
      dTheta /= 2;
      theta -= dTheta;
      numberOfHomotopySubSteps *= 2;
      iter = 0;
    }
  }
  if (converged) {
    --numberOfHomotopySubSteps;
    if (numberOfHomotopySubSteps != 0) {
      theta += dTheta;
      lastZeros = zeros;
      iter = 0;
      converged = false;
    }
  }
}
```

- Implementation using the `@AdditionalConvergenceCheck` block.

Second algorithm - I

- We consider the following modified residual:

$$\vec{F}_{\mu}\left(\Delta\,\vec{Y}\right) = (1-\mu)\,\vec{F}_{0}\left(\Delta\,\vec{Y}\right) + \mu\,\vec{F}\left(\Delta\,\vec{Y}\right)$$

where $\vec{F}_{0}$ is the residual associated with a purely elastic material:

$$\vec{F}_{0} = \begin{pmatrix} \Delta\,\underline{\epsilon}^{el} - \Delta\,\underline{\epsilon}^{to} \\ \vdots \\ \Delta\,y_{n_y} \end{pmatrix}$$

- Comments:
  - $\dfrac{\partial \vec{F}_{\mu}}{\partial \mu} = \vec{F}\left(\Delta\,\vec{Y}\right) - \vec{F}_{0}\left(\Delta\,\vec{Y}\right)$
  - The implementation does not require any user extensions, at least if the `StandardElasticity` brick is used (or the `StandardElastoViscoPlasticity` brick)
  - One can use a `Runge-Kutta` method to determine $\Delta\,\vec{Y}_{\mu}$ and a final Newton resolution for accuracy.

Second algorithm - II

- Plasticity is a problem. Consider the perfect plastic case:

$$
\begin{aligned}
f_{p,\mu} &= (1-\mu)\,\Delta\,p + \frac{\mu}{E}\,(\sigma_{eq} - R_0) \\
&= \frac{\mu}{E}\,\left(\sigma_{eq} - R_0 + \frac{(1-\mu)}{\mu}\,\Delta\,p\right)
\end{aligned}
$$

Second algorithm - II

- Plasticity is a problem. Consider the perfect plastic case:

$$
\begin{aligned}
f_{p,\mu} &= (1-\mu)\,\Delta\,p + \frac{\mu}{E}\,(\sigma_{eq} - R_0) \\
&= \frac{\mu}{E}\left(\sigma_{eq} - R_0 + \frac{(1-\mu)}{\mu}\,\Delta\,p\right)
\end{aligned}
$$

  - There is a value μ_c such that the solution must satisfy $\sigma_{eq} = 0$. In this case, the normal is not defined, leading a divergence of $\Delta p\,(\mu)$:

$$
\Delta p\,(\mu) \underset{\theta \to \pm\mu_c}{\rightarrow} \pm\infty
$$

Second algorithm - II

- Plasticity is a problem. Consider the perfect plastic case:

$$
\begin{aligned}
f_{p,\mu} &= (1-\mu)\,\Delta\,p + \frac{\mu}{E}\,(\sigma_{eq} - R_0) \\
&= \frac{\mu}{E}\,\left(\sigma_{eq} - R_0 + \frac{(1-\mu)}{\mu}\,\Delta\,p\right)
\end{aligned}
$$

  - There is a value μ_c such that the solution must satisfy $\sigma_{eq} = 0$. In this case, the normal is not defined, leading a divergence of $\Delta p\,(\mu)$:

$$
\Delta p\,(\mu) \underset{\theta \to \pm\mu_c}{\rightarrow} \pm\infty
$$

- Complementary functions allows to overcome this problem:

$$
f_{p,\mu} = (1-\mu)\,\Delta\,p + \mu\,\Phi_{\varepsilon}\left(-\frac{\sigma_{eq} - R_0}{E}, \Delta\,p\right)
$$

where Φ_{ε} is the regularized Fisher-Burmeister complementary function.

Second algorithm - II

- Plasticity is a problem. Consider the perfect plastic case:

$$
\begin{aligned}
f_{p,\mu} &= (1-\mu)\,\Delta\,p + \frac{\mu}{E}\,(\sigma_{eq} - R_0) \\
&= \frac{\mu}{E}\,\left(\sigma_{eq} - R_0 + \frac{(1-\mu)}{\mu}\,\Delta\,p\right)
\end{aligned}
$$

  - There is a value μ_c such that the solution must satisfy $\sigma_{eq} = 0$. In this case, the normal is not defined, leading a divergence of $\Delta p\,(\mu)$:

$$
\Delta p\,(\mu) \underset{\theta \rightarrow \pm\mu_c}{\rightarrow} \pm\infty
$$

- Complementary functions allows to overcome this problem:

$$
f_{p,\mu} = (1-\mu)\,\Delta\,p + \mu\,\Phi_{\varepsilon}\left(-\frac{\sigma_{eq} - R_0}{E}, \Delta\,p\right)
$$

where Φ_{ε} is the regularized Fisher-Burmeister complementary function.

  - See `https://thelfer.github.io/tfel/web/FischerBurmeister.html` for details.

Outline

2. First test case

Analysis of an integration failure

- This test is based on a `MTest` file generated by the `Alcyone` fuel performance code.
- The viscoplastic strain rate is given by:

$$\underline{\dot{\epsilon}}^{vis} = \dot{p}\,\underline{\mathbf{n}} \quad \text{with} \quad \dot{p} = \dot{\varepsilon}_0 \sinh\left(\frac{\sigma_{eq}}{\sigma_0}\right)$$

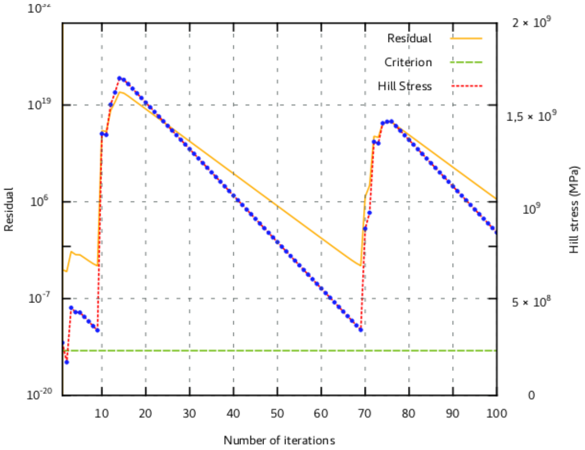
where:

  - p is the equivalent viscoplastic strain;
  - σ_{eq} is the Hill stress:

$$\sigma_{eq} = \sqrt{\underline{\sigma}\,:\,\underline{\underline{\mathbf{H}}}\,:\,\underline{\sigma}}$$

  - $\underline{\mathbf{n}} = \dfrac{\partial \sigma_{eq}}{\partial \underline{\sigma}}$ is flow direction given by the normal to the Hill stress;
  - $\dot{\varepsilon}_0$ and σ_0 are material parameters.

Evolution of the residual with the Newton method



Homotopy algorithms

- The first algorithm, used as a fall-back, allows to convergence.
  - Only on homotopy substep is required.
- The second algorithm can be very effective.
  - We have implemented various strategies:
    - Direct determination of the solution with a Runge-Kutta algorithm with corrector/predictor and adaptative time steps.
    - Estimation of the solution with an RK4 algorithm and then a standard Newton.
  - Only 24 evalutations of the residual and jacobian are required in some cases.

Outline

3. Second test case

Scherzinger's test: description

- Scherzinger, W. M. 2017. *"A Return Mapping Algorithm for Isotropic and Anisotropic Plasticity Models Using a Line Search Method."* Computer Methods in Applied Mechanics and Engineering 317 (April): 526–53. https://doi.org/10.1016/j.cma.2016.11.026.
- From the null stress state, impose loading increment up to 30 times the yield stress in each direction of the π-plane
- Applied to a perfect plastic behaviour following the `Hosford` criterion:

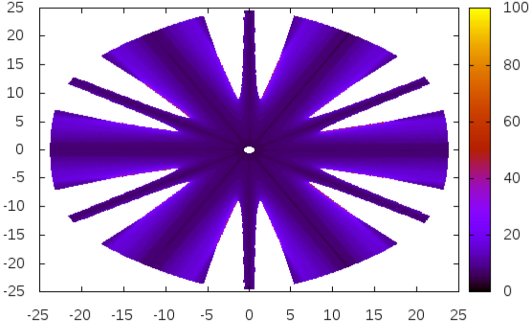
$$\sigma^{H}_{eq} = \sqrt[a]{\frac{1}{2}\left(|\sigma_1 - \sigma_2|^a + |\sigma_1 - \sigma_3|^a + |\sigma_2 - \sigma_3|^a\right)}$$

- The behaviour becomes harder to integrate as *a* grows.
- Pratical exponents are 6 and 8.
- For an exponent of 100, the yield surface is very close to TRESCA'

Scherzinger’s test: implementation

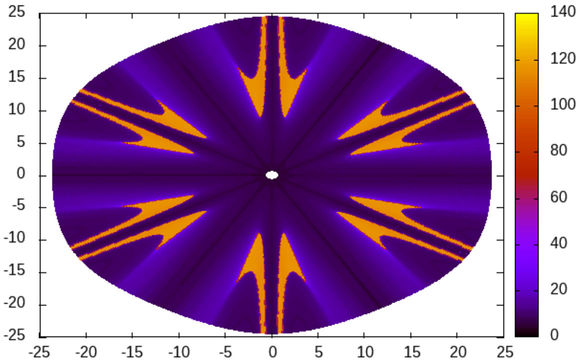
```python
from math import pi,cos,sin
import tfel.math     as tmath
import tfel.material as tmaterial
import mtest

nmax  = 1000
nmax2 = 1000
E  = 150e9
nu = 0.3
s0 = 150e6

for a in [pi*(-1.+(2.*i)/(nmax-1)) for i in range(0,nmax)]:
    for x in [1+(29.*i)/(nmax2-1) for i in range(0,nmax2)]:
        nbiter=0
        s      = tmath.makeStensor3D(tmaterial.buildFromPiPlane(cos(a),sin(a)))
        seq    = tmaterial.computeHosfordStress(s,100,1.e-12);
        s     *= x*(s0/seq)
        e0     = (s[0]-nu*(s[1]+s[2]))/E
        e1     = (s[1]-nu*(s[0]+s[2]))/E
        e2     = (s[2]-nu*(s[0]+s[1]))/E
        m     = mtest.MTest()
        mtest.setVerboseMode(mtest.VerboseLevel.VERBOSE_QUIET)
        m.setModellingHypothesis('Tridimensional')
        m.setMaximumNumberOfSubSteps(1)
        m.setBehaviour('abaqus','src/libABAQUSBEHAVIOUR.so',
                       'HOSFORDPERFECTPLASTICITY100_3D');
        m.setExternalStateVariable("Temperature",293.15)
        m.setImposedStrain('EXX',{0:0,1:e0})
        m.setImposedStrain('EYY',{0:0,1:e1})
        m.setImposedStrain('EZZ',{0:0,1:e2})
        m.setImposedStrain('EXY',0)
        m.setImposedStrain('EXZ',0)
        m.setImposedStrain('EYZ',0)
        s  = mtest.MTestCurrentState()
        wk = mtest.MTestWorkSpace()
        m.completeInitialisation()
        m.initializeCurrentState(s)
        m.initializeWorkSpace(wk)
        try:
            m.execute(s,wk,0,1)
            nbiter=s.getInternalStateVariableValue("NumberOfIterations")
        except:
            nbiter=100
            break
        print(str((x/seq)*cos(a))+" "+str((x/seq)*sin(a))+" "+str(nbiter))
    print()
```

Scherzinger's test: results ($a = 6$)



- Newton algorithm: Results are consistent with Scherzinger's ones

Scherzinger's test: results ($a = 6$)



- The first homotopy algorithm indeed increases the robustness of the implicit resolution, without special treatment of plasticity.

Higher Hosford exponents

- Results with higher Hosford exponents (for instance 100) are mixed:

Higher Hosford exponents

- Results with higher Hosford exponents (for instance 100) are mixed:
  - Homotopy algorithms only slightly increase the robustness compared to the standard Newton algorithm. The Levenberg-Marquart algorithm is fare more effective.

Higher Hosford exponents

- Results with higher Hosford exponents (for instance 100) are mixed:
  - Homotopy algorithms only slightly increase the robustness compared to the standard Newton algorithm. The Levenberg-Marquart algorithm is fare more effective.
  - The second algorithm is fairly sensible to a the numerical parameters used:

Higher Hosford exponents

- Results with higher Hosford exponents (for instance 100) are mixed:
  - Homotopy algorithms only slightly increase the robustness compared to the standard Newton algorithm. The Levenberg-Marquart algorithm is fare more effective.
  - The second algorithm is fairly sensible to a the numerical parameters used:
    - Convergence criteria of the Runge-Kutta algorithm with adaptative time steps.

Higher Hosford exponents

- Results with higher Hosford exponents (for instance 100) are mixed:
  - Homotopy algorithms only slightly increase the robustness compared to the standard Newton algorithm. The Levenberg-Marquart algorithm is fare more effective.
  - The second algorithm is fairly sensible to a the numerical parameters used:
    - Convergence criteria of the Runge-Kutta algorithm with adaptative time steps.
    - Value used to regularized the Fischer-Burmeister function.

Outline

3. Conclusions

- Only **preliminary** results were presented.
- Some results are very encouraging, but challenges remain:
  - Choice of the role of the homotopy algorithm. Three possible roles were highlighted in this talk:
    1. Direct determination of the solution.
    2. Fall-back mechanism.
    3. Estimation of the solution.
  - Choice of the strategy to update the homotopy parameter:
    - Depends on the role of the homotopy algorithm.
  - Time independent behaviours remain a problem.

Outline

Homotopy methods at structural scale

A. Homotopy methods at structural scale

Standard resolution scheme

- Mechanical equilibrium: find$\Delta\vec{\mathbb{U}}$ such as:

$$\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{O}} \quad \text{with} \quad \vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{F}}_i\left(\Delta\vec{\mathbb{U}}\right)\Big|_{t+\Delta\,t}-\vec{\mathbb{F}}_e\Big|_{t+\Delta\,t}$$

Standard resolution scheme

- Mechanical equilibrium: find$\Delta\vec{\mathbb{U}}$ such as:

$$\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{O}}\quad\text{with}\quad\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{F}}_{i}\left(\Delta\vec{\mathbb{U}}\right)\Big|_{t+\Delta\,t}-\vec{\mathbb{F}}_{e}\Big|_{t+\Delta\,t}$$

- element contribution to inner forces:

$$\begin{aligned}\vec{\mathbb{F}}_{i}^{e}&=\int_{V^{e}}\underline{\sigma}_{t+\Delta t}\left(\Delta\,\underline{\epsilon}^{to},\Delta\,t\right)\,:\,\underline{\mathbf{B}}\,\mathrm{d}V\\&=\sum_{i=1}^{N^{G}}\left(\underline{\sigma}_{t+\Delta\,t}\left(\Delta\underline{\epsilon}^{to}\left(\vec{\eta}_{i}\right),\Delta\,t\right)\,:\,\underline{\underline{\mathbf{B}}}\left(\vec{\eta}_{i}\right)\right)w_{i}\end{aligned}$$

where $\underline{\mathbf{B}}$ gives the relationship between $\Delta\,\underline{\epsilon}^{to}$ and $\Delta\vec{\mathbb{U}}$

Standard resolution scheme

- Mechanical equilibrium: find$\Delta\vec{\mathbb{U}}$ such as:

$$\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{O}}\quad\text{with}\quad\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{F}}_i\left(\Delta\vec{\mathbb{U}}\right)\Big|_{t+\Delta\,t}-\vec{\mathbb{F}}_e\Big|_{t+\Delta\,t}$$

- element contribution to inner forces:

$$\vec{\mathbb{F}}_i^e=\sum_{i=1}^{N^G}\left(\underline{\sigma}_{t+\Delta\,t}\left(\Delta\underline{\epsilon}^{to}\left(\vec{\eta}_i\right),\Delta\,t\right):\underline{\underline{\mathbf{B}}}\left(\vec{\eta}_i\right)\right)w_i$$

- Resolution using the Newton-Raphson algorithm:

$$\Delta\vec{\mathbb{U}}^{n+1}=\Delta\vec{\mathbb{U}}^{n}-\left(\left.\frac{\partial\vec{\mathbb{R}}}{\partial\Delta\vec{\mathbb{U}}}\right|_{\Delta\vec{\mathbb{U}}^{n}}\right)^{-1}.\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}^{n}\right)=\Delta\vec{\mathbb{U}}^{n}-\underline{\underline{\mathbb{K}}}^{-1}.\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}^{n}\right)$$

Standard resolution scheme

- Mechanical equilibrium: find$\Delta\vec{\mathbb{U}}$ such as:

$$\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{O}}\quad\text{with}\quad\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\right)=\vec{\mathbb{F}}_i\left(\Delta\vec{\mathbb{U}}\right)\Big|_{t+\Delta\,t}-\vec{\mathbb{F}}_e\Big|_{t+\Delta\,t}$$

- element contribution to inner forces:

$$\vec{\mathbb{F}}_i^e=\sum_{i=1}^{N^G}\left(\underline{\sigma}_{t+\Delta\,t}\left(\Delta\underline{\epsilon}^{to}\left(\vec{\eta}_i\right),\Delta\,t\right):\underline{\underline{\mathbf{B}}}\left(\vec{\eta}_i\right)\right)w_i$$

- Resolution using the Newton-Raphson algorithm:

$$\Delta\vec{\mathbb{U}}^{n+1}=\Delta\vec{\mathbb{U}}^{n}-\underline{\underline{\mathbb{K}}}^{-1}.\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}^{n}\right)$$

- element contribution to the stiffness:

$$\underline{\underline{\mathbb{K}}}^e=\sum_{i=1}^{N^G}{}^t\underline{\underline{\mathbf{B}}}\left(\vec{\eta}_i\right):\frac{\partial\Delta\underline{\sigma}}{\partial\Delta\underline{\epsilon}^{to}}\left(\vec{\eta}_i\right):\underline{\underline{\mathbf{B}}}\left(\vec{\eta}_i\right)w_i$$

$\dfrac{\partial\Delta\underline{\sigma}}{\partial\Delta\underline{\epsilon}^{to}}$ is the **consistent tangent operator**

An homotopy scheme

- The residual of an incremental elastic resolution is:

$$\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\right) = \underline{\underline{\mathbb{K}}}^{el}\,\Delta\vec{\mathbb{U}} - \left.\vec{\mathbb{F}}_{e}\right|_{t+\Delta\,t} + \left.\vec{\mathbb{F}}_{i}\right|_{t}$$

An homotopy scheme

- The residual of an incremental elastic resolution is:

$$\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\right) = \underline{\underline{\mathbb{K}}}^{el}\,\Delta\vec{\mathbb{U}} - \left.\vec{\mathbb{F}}_e\right|_{t+\Delta\,t} + \left.\vec{\mathbb{F}}_i\right|_t$$

  - This auxiliary problem is often used to provide a first estimate of the solution.

An homotopy scheme

- The residual of an incremental elastic resolution is:

$$\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\right) = \underline{\underline{\mathbb{K}}}^{el}\,\Delta\vec{\mathbb{U}} - \left.\vec{\mathbb{F}}_{e}\right|_{t+\Delta\,t} + \left.\vec{\mathbb{F}}_{i}\right|_{t}$$

  - This auxiliary problem is often used to provide a first estimate of the solution.

- Let define the residual $\vec{\mathbb{R}}^{\chi}$ as:

$$\vec{\mathbb{R}}^{\chi}\left(\Delta\vec{\mathbb{U}}\left(\chi\right),\chi\right) = \chi\,\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right) + (1-\chi)\,\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right)$$

  with $\chi \in [0, 1]$.

An homotopy scheme

- The residual of an incremental elastic resolution is:
$$\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\right)=\underline{\underline{\mathbb{K}}}^{el}\,\Delta\vec{\mathbb{U}}-\vec{\mathbb{F}}_{e}\Big|_{t+\Delta\,t}+\vec{\mathbb{F}}_{i}\Big|_{t}$$
    - This auxiliary problem is often used to provide a first estimate of the solution.
- Let define the residual $\vec{\mathbb{R}}^{\chi}$ as:
$$\vec{\mathbb{R}}^{\chi}\left(\Delta\vec{\mathbb{U}}\left(\chi\right),\chi\right)=\chi\,\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right)+\left(1-\chi\right)\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right)$$
with $\chi\in[0,1]$.
- χ defines an homotopy between the (trivial) incremental elastic solution $\Delta\vec{\mathbb{U}}\left(0\right)$ and the solution of the initial non linear problem $\Delta\vec{\mathbb{U}}\left(1\right)$

An homotopy scheme

- The residual of an incremental elastic resolution is:

$$\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\right) = \underline{\underline{\mathbb{K}}}^{el}\,\Delta\vec{\mathbb{U}} - \left.\vec{\mathbb{F}}_e\right|_{t+\Delta\,t} + \left.\vec{\mathbb{F}}_i\right|_t$$

  - This auxiliary problem is often used to provide a first estimate of the solution.

- Let define the residual $\vec{\mathbb{R}}^{\chi}$ as:

$$\vec{\mathbb{R}}^{\chi}\left(\Delta\vec{\mathbb{U}}\left(\chi\right),\chi\right) = \chi\,\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right) + \left(1-\chi\right)\,\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right)$$

  with $\chi \in [0, 1]$.

- χ defines an homotopy between the (trivial) incremental elastic solution $\Delta\vec{\mathbb{U}}\left(0\right)$ and the solution of the initial non linear problem $\Delta\vec{\mathbb{U}}\left(1\right)$

- At fixed χ, the jacobian is:

$$\frac{\partial\vec{\mathbb{R}}^{\chi}}{\partial\Delta\vec{\mathbb{U}}} = \chi\,\underline{\underline{\mathbb{K}}}^{el} + \left(1-\chi\right)\,\underline{\underline{\mathbb{K}}}$$

An homotopy scheme

- The residual of an incremental elastic resolution is:

$$\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\right) = \underline{\underline{\mathbb{K}}}^{el}\,\Delta\vec{\mathbb{U}} - \left.\vec{\mathbb{F}}_e\right|_{t+\Delta\,t} + \left.\vec{\mathbb{F}}_i\right|_{t}$$

  - This auxiliary problem is often used to provide a first estimate of the solution.
- Let define the residual $\vec{\mathbb{R}}^{\chi}$ as:

$$\vec{\mathbb{R}}^{\chi}\left(\Delta\vec{\mathbb{U}}\left(\chi\right),\chi\right) = \chi\,\vec{\mathbb{R}}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right) + \left(1-\chi\right)\,\vec{\mathbb{R}}^{el}\left(\Delta\vec{\mathbb{U}}\left(\chi\right)\right)$$

  with $\chi \in [0, 1]$.
- χ defines an homotopy between the (trivial) incremental elastic solution $\Delta\vec{\mathbb{U}}\left(0\right)$ and the solution of the initial non linear problem $\Delta\vec{\mathbb{U}}\left(1\right)$
- At fixed χ, the jacobian is:

$$\frac{\partial\vec{\mathbb{R}}^{\chi}}{\partial\Delta\vec{\mathbb{U}}} = \chi\,\underline{\underline{\mathbb{K}}}^{el} + \left(1-\chi\right)\,\underline{\underline{\mathbb{K}}}$$

- The elastic stiffness can stabilize the real consistent tangent operator in many cases, i.e.

$$\frac{\partial\vec{\mathbb{R}}^{\chi}}{\partial\Delta\vec{\mathbb{U}}} = \chi\,\underline{\underline{\mathbb{K}}}^{el} + \left(1-\chi\right)\,\underline{\underline{\mathbb{K}}}$$

Resolution

- By virtue of the implicit function theorem, assuming that $\dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}$ is invertible:

$$
\begin{aligned}
\frac{\partial \Delta \vec{\mathbb{U}}}{\partial \chi} &= -\left(\frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \chi} \\
&= -\left(\frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \left[\left.\vec{\mathbb{F}}_{i}\right|_{t+\Delta\, t}\left(\Delta \vec{\mathbb{U}}\left(\chi\right)\right) - \left.\vec{\mathbb{F}}_{i}\right|_{t} - \underline{\underline{\mathbb{K}}}^{el}\, \Delta \vec{\mathbb{U}}\left(\chi\right)\right]
\end{aligned}
$$

Resolution

- By virtue of the implicit function theorem, assuming that $\dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}$ is invertible:

$$
\begin{aligned}
\frac{\partial \Delta \vec{\mathbb{U}}}{\partial \chi} &= -\left(\frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \chi} \\
&= -\left(\frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \left[\left.\vec{\mathbb{F}}_i\right|_{t+\Delta\, t}\left(\Delta \vec{\mathbb{U}}\,(\chi)\right) - \left.\vec{\mathbb{F}}_i\right|_{t} - \underline{\underline{\mathbb{K}}}^{el}\, \Delta \vec{\mathbb{U}}\,(\chi)\right]
\end{aligned}
$$

- This can be solved by a RK method with automatic χ-stepping with the initial value:

$$
\Delta \vec{\mathbb{U}}\,(0) = \left(\underline{\underline{\mathbb{K}}}^{el}\right)^{-1} \left[\left.\vec{\mathbb{F}}_e\right|_{t+\Delta\, t} - \left.\vec{\mathbb{F}}_i\right|_{t}\right]
$$

Resolution

- By virtue of the implicit function theorem, assuming that $\dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}$ is invertible:

$$
\begin{aligned}
\frac{\partial \Delta \vec{\mathbb{U}}}{\partial \chi} &= -\left(\frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \chi} \\
&= -\left(\frac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \left[\left.\vec{\mathbb{F}}_i\right|_{t+\Delta\, t}\left(\Delta \vec{\mathbb{U}}\left(\chi\right)\right) - \left.\vec{\mathbb{F}}_i\right|_{t} - \underline{\underline{\mathbb{K}}}^{el}\, \Delta \vec{\mathbb{U}}\left(\chi\right)\right]
\end{aligned}
$$

- This can be solved by a RK method with automatic χ-stepping with the initial value:

$$
\Delta \vec{\mathbb{U}}\left(0\right) = \left(\underline{\underline{\mathbb{K}}}^{el}\right)^{-1} \left[\left.\vec{\mathbb{F}}_e\right|_{t+\Delta\, t} - \left.\vec{\mathbb{F}}_i\right|_{t}\right]
$$

- This provides an estimate of the solution $\Delta \vec{\mathbb{U}}\left(1\right)$ that can initialize the Newton resolution.

Resolution

- By virtue of the implicit function theorem, assuming that $\dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}$ is invertible:

$$\begin{aligned}
\dfrac{\partial \Delta \vec{\mathbb{U}}}{\partial \chi} &= -\left(\dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \chi} \\
&= -\left(\dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}\right)^{-1} \left[\left.\vec{\mathbb{F}}_i\right|_{t+\Delta\, t}\left(\Delta \vec{\mathbb{U}}\,(\chi)\right) - \left.\vec{\mathbb{F}}_i\right|_{t} - \underline{\underline{\mathbb{K}}}^{el}\,\Delta \vec{\mathbb{U}}\,(\chi)\right]
\end{aligned}$$

- This can be solved by a RK method with automatic χ-stepping with the initial value:

$$\Delta \vec{\mathbb{U}}\,(0) = \left(\underline{\underline{\mathbb{K}}}^{el}\right)^{-1}\left[\left.\vec{\mathbb{F}}_e\right|_{t+\Delta\, t} - \left.\vec{\mathbb{F}}_i\right|_{t}\right]$$

- This provides an estimate of the solution $\Delta \vec{\mathbb{U}}\,(1)$ that can initialize the Newton resolution.
- Assuming that $\dfrac{\partial \vec{\mathbb{R}}^{\chi}}{\partial \Delta \vec{\mathbb{U}}}$ is invertible, this method has strong convergence guarantees.

Discussion

- The implicit system is not solved directly, we "only" ensure that $\dfrac{\mathrm{d}\vec{\mathbb{R}}^{\chi}}{\mathrm{d}\chi}$ is null. This is why a final Newton resolution is required. This final Newton resolution may be limited to one evaluation of the residual.

Discussion

- The implicit system is not solved directly, we "only" ensure that $\dfrac{\mathrm{d}\vec{\mathbb{R}}^{\chi}}{\mathrm{d}\chi}$ is null. This is why a final Newton resolution is required. This final Newton resolution may be limited to one evaluation of the residual.
- Performances depends on:

Discussion

- The implicit system is not solved directly, we "only" ensure that $\dfrac{\mathrm{d}\vec{\mathbb{R}}^{\chi}}{\mathrm{d}\chi}$ is null. This is why a final Newton resolution is required. This final Newton resolution may be limited to one evaluation of the residual.
- Performances depends on:
  - the order of the RK method.

Discussion

- The implicit system is not solved directly, we "only" ensure that $\dfrac{\mathrm{d}\vec{\mathbb{R}}^{\chi}}{\mathrm{d}\chi}$ is null. This is why a final Newton resolution is required. This final Newton resolution may be limited to one evaluation of the residual.
- Performances depends on:
  - the order of the RK method.
  - the criteria value used to accept RK steps:

Discussion

- The implicit system is not solved directly, we "only" ensure that $\dfrac{\mathrm{d}\vec{\mathbb{R}}^{\chi}}{\mathrm{d}\chi}$ is null. This is why a final Newton resolution is required. This final Newton resolution may be limited to one evaluation of the residual.
- Performances depends on:
  - the order of the RK method.
  - the criteria value used to accept RK steps:
    - a loose value may provide a poor estimate

Discussion

- The implicit system is not solved directly, we "only" ensure that $\dfrac{\mathrm{d}\vec{\mathbb{R}}^{\chi}}{\mathrm{d}\chi}$ is null. This is why a final Newton resolution is required. This final Newton resolution may be limited to one evaluation of the residual.
- Performances depends on:
  - the order of the RK method.
  - the criteria value used to accept RK steps:
    - a loose value may provide a poor estimate
    - a strigent value is not required: accuracy is provided by the final Newton resolution.