DE LA RECHERCHE À L'INDUSTRIE

www.cea.fr

Writing portable behaviours implementations

— Thomas Helfer

Februar 2015

Purpose of this talk

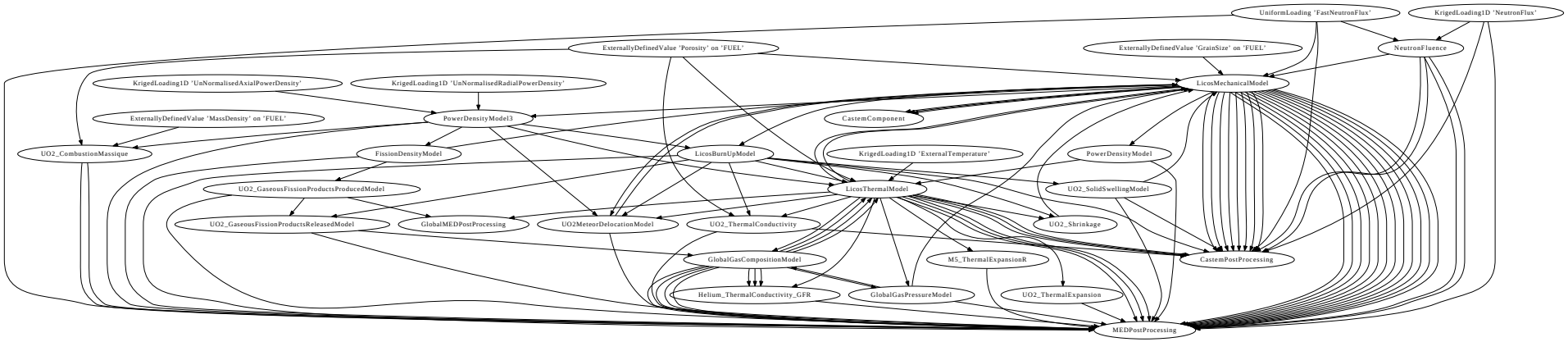
- It is possible to reuse an `MFront` among all the supported simulation codes :
  - However, one has to comply to all the specific requirements of all the supported simulation codes.
- This talks provides some guidelines and provides a comparison of the choices made in each solver.

Good practices

Units

- use SI units :
  - general purpose finite element solver don't put constraints on units : user are free to choose the most appropriate one (in his own view).
    - how many hours have you lost due to units inconsistencies ?
  - `PLEIADES` fuel performance codes uses SI units
    - consistent choices

Glossary names




```
@StateVariable evp;
evp.setGlossaryName("ViscoplasticStrain");
```

- glossary names make explicit the inputs and the outputs of material properties and behaviours :
  - in some solvers, those names are visible to the user (Code-Aster, ZeBuLoN, `licos`, `mtest`)
  - used by some solvers tp properly fill the input of dynamically loaded mechanical behaviours (`licos`, Cyrano3, `mtest` ZeBuLoN)
  - used internally for specific tasks :
    - the value of a variable whose glossary name is `AxialStrain` is used in finite strain strategies to compute the Cauchy stress

Material properties

- don't use material properties defined through `@MaterialProperties` :
  - are they evaluated at the beginning of the time step, at the end ?
  - part of physical information of your material is in the solver input file : this is bad.
  - extra work is requested in the finite element solver which may induce lot of overhead (*a priori* more than evaluating material properties within the behaviour).
  - use the `@MaterialLaw` keyword instead
  - **note :** elastic stiffness in orthotropic behaviours still requires some work

Portability issues and advices

Supported modelling hypotheses

Solver	Modelling hypothesis
`Cast3M`	Axisymmetrical generalised plane strain, plane strain, generalised plane strain, plane stress, axisymmetrical, tridimensional
Code-Aster	Plane strain, plane stress [1], axisymmetrical, tridimensional
`ZeBuLoN`	$1D$, $2D$, $3D$ [2]
Cyrano3	Axisymmetrical generalised plane strain, axisymmetrical generalised plane stress [3]

1. Can be supported by the behaviour or by the solver (De Borst)
2. Modelling hypothesis is not accessible within the behaviour. Plane stress is supported by the finite element (additional ddl)
3. The default.

Conventions (strains)

3D

Solver	Convention
`TFEL`	$\left(\epsilon^{to}_{xx}, \epsilon^{to}_{yy}, \epsilon^{to}_{zz}, \sqrt{2}\,\epsilon^{to}_{xy}, \sqrt{2}\,\epsilon^{to}_{xy}, \sqrt{2}\,\epsilon^{to}_{yz}\right)$
`Cast3M`	$\left(\epsilon^{to}_{xx}, \epsilon^{to}_{yy}, \epsilon^{to}_{zz}, 2\,\epsilon^{to}_{xy}, 2\,\epsilon^{to}_{xy}, 2\,\epsilon^{to}_{yz}\right)$
Code-Aster	$\left(\epsilon^{to}_{xx}, \epsilon^{to}_{yy}, \epsilon^{to}_{zz}, \sqrt{2}\,\epsilon^{to}_{xy}, \sqrt{2}\,\epsilon^{to}_{xy}, \sqrt{2}\,\epsilon^{to}_{yz}\right)$ [4]
`ZeBuLoN`	$\left(\epsilon^{to}_{xx}, \epsilon^{to}_{yy}, \epsilon^{to}_{zz}, \sqrt{2}\,\epsilon^{to}_{xy}, \sqrt{2}\,\epsilon^{to}_{yz}, \sqrt{2}\,\epsilon^{to}_{xz}\right)$

1D

Solver	Convention
`Cast3M`	$\left(\epsilon^{to}_{rr}, \epsilon^{to}_{zz}, \epsilon^{to}_{\theta}\right)$
Code-Aster	$\left(\epsilon^{to}_{rr}, \epsilon^{to}_{zz}, \epsilon^{to}_{\theta}\right)$
Cyrano3	$\left(\epsilon^{to}_{rr}, \epsilon^{to}_{\theta}, \epsilon^{to}_{zz}\right)$

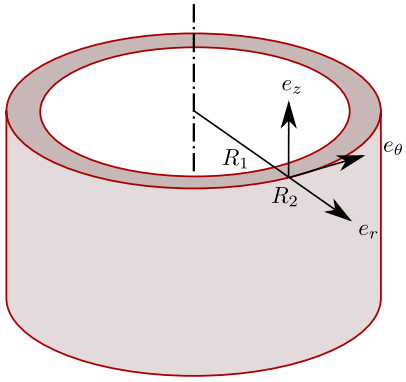
4. Requires "still to be done" modifications to the Code-Aster external behaviour handler and `MFront` Code-Aster interface

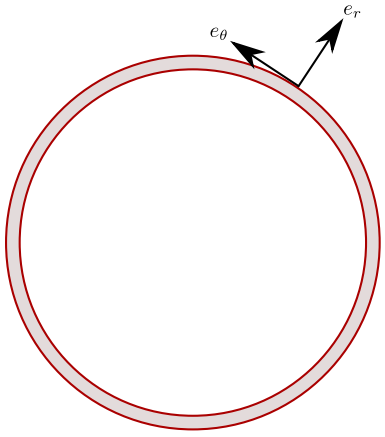
Conversions

- Conversions of strain, stresses and consistent tangent operators are handled by the interface
- Internal state variables are stored :
  - Using TFEL conventions
    - Except for ZeBuLoN to match native behaviours (this may induce some overhead)
  - In the material frame (for orthotropic behaviours)

It would be nice to have native conversion procedures in each solver for post-processings

Orthotropy





- Rotation to and from in the material frame must be handled by the behaviour in Code-Aster and `Cast3M`
- In `Cast3M` and Code-Aster, behaviour conventions and orthotropic axes definitions (in the input file) *must be consistent*
- no general purpose solver allow uniform definition of the orthotropic axes for all modelling hypotheses
  - example of pipes
    - the definition of the Hill tensor has to be different in $2D\,(r, z)$ and $2D\,(r, \theta)$
    - this is an issue in `ZeBuLoN`

Plane stress

- Support for plane stress and axisymmetrical generalised plane stress (Cyrano3) modelling hypotheses *shall* explicitly provided for performance reasons :
  - De Borst algorithm (Cyrano3 and Code-Aster) is slow !
  - the generic algorithm provided in the `Cast3M` interface is very slow !

Prediction operator

- only usable in Code-Aster !

Finite strain strategies

- finite strain strategies allows the user to reuse the small strain formalism to build consistent finite strain behaviours.
- the most portable (and the most appealing on physical grounds) finite strain strategy is the logarithmic strain formalism introduced by Miehe et al. [Miehe 02].
  - native in Code-Aster and `ZeBuLoN` (although this can be a problem, the user must understand what he is doing)
  - handled by `MFront` in the `Cast3M` interface
  - usable in $1D$ fuel performance codes without any modification of the code (requires special interface feature)

Consistent tangent operator

- most finite element solver preferably uses the consistent tangent operator :
  - `Cast3M` **does not** : fixed point iterations and (modified and specialised) Anderson acceleration algorithm ;
    - support for the true Newton is planned in COPL/INCREPL
  - `ZeBuLoN` only consider one type of operator : the consistent tangent operator (which can be hand-crafted to keep a positive definite matrix).
  - Code-Aster may use one the following operator :
    - elastic
    - secant
    - tangent operator
    - consistent tangent operator

Finite strain consistent tangent operator

Solver	Definition
`Cast3M`	None
Code-Aster	$\frac{\partial \underset{\sim}{\tau}}{\partial \Delta \underset{\sim}{\mathbf{F}}}$ [5]
`ZeBuLoN`	`DSIG_DD` [6]

- `MFront` handles automatically the conversion from the tangent operator defined by the user and the tangent operator expected by the solver.

5. $\Delta \underset{\sim}{\mathbf{F}} = \underset{\sim}{\mathbf{F}}\Big|_{t+\Delta t} . \left(\underset{\sim}{\mathbf{F}}\Big|_{t} \right)^{-1}$
6. The only finite strain formulation supported by `MFront` is `Updated_Lagrangian`

References

Miehe C., Apel N. et Lambrecht M.
Anisotropic additive plasticity in the logarithmic strain space : modular kinematic formulation and implementation based on incremental minimization principles for standard materials.
Computer Methods in Applied Mechanics and Engineering, Novembre 2002, vol 191, n° 47–48, p 5383–5425.