

DE LA RECHERCHE À L'INDUSTRIE



Paris - Saclay

www.cea.fr

AMITEX_FFT training

-

Practice

-

V8.17.13

-

10/10/2023

Prerequisites for training

Pre-requisites for amitex_fftp (gcc/openmpi/fftw)

Octave

Paraview

Mfront

Meld (if possible)

(Solution to read .pptx and .pdf files)

Lionel Gélébart

Installation

Geometry description for AMITEX

-> reading vtk file *paraview* + generation (with *matlab/octave*)

amitex_fftp command line

-> First simple linear simulation

Xml files material, loading-output, algorithms

-> Linear simulations

-> First non linear simulation (ex : viscoelasticity)

-> Finite strain simulations (ex : rotation)

Behaviors

-> umat –user behaviors

-> MFRONT-user behaviors

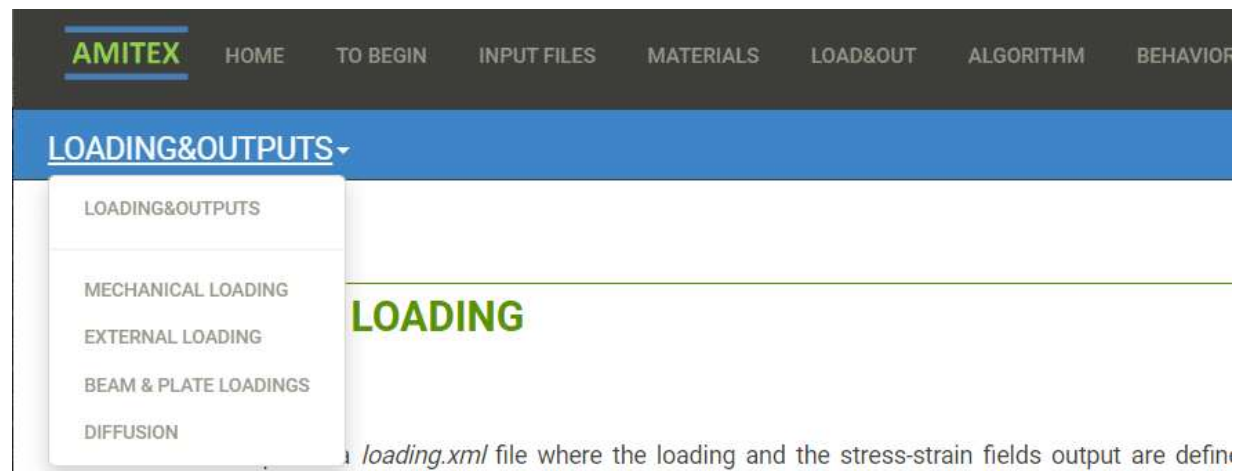
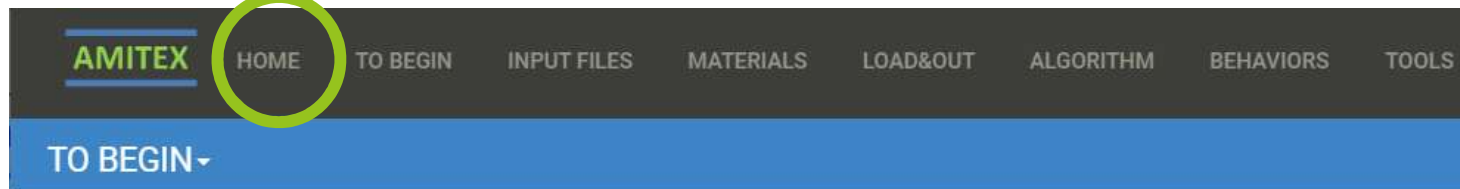
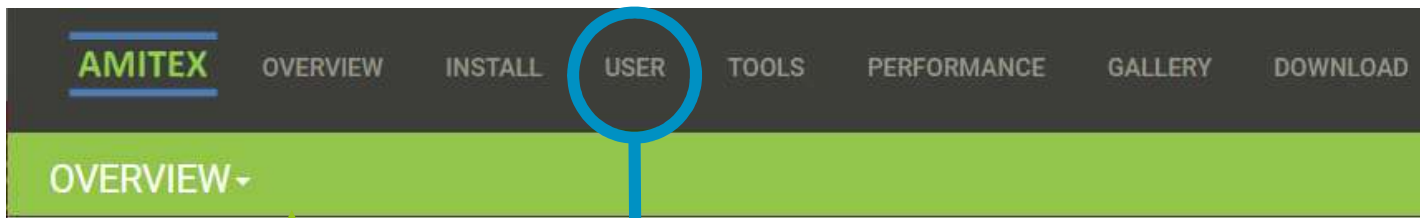
Composite voxels

Practice : - proposed application

- free application

THE WEBSITE IS THE REFERENCE DOCUMENT!

http://www.maisondelasimulation.fr/projects/amtex/general/_build/html/index.html



http://marquises/wiki/_build/html/usage.html#softwares



➤ Linux-based OS

➤ Pre-requisites

compiler : gfortran (>v5.0.0) OR ifort (and nothing else!)
 MPI library : openMPI OR intelMPI
 fftw3 : simple **AND** double precision, development version
 Mfront (optionnal)

ifort+intelMPI > gfortran+openMPI

Installations :

If root or super user => 1. Prefer 'package installer' (apt-get, yum install etc...)
 => 2. Install by yourself (see amitex_ffp's website)
 Else => **Ask your administrator !**
 Or => Install by yourself on your home (see amitex_ffp's website)

➤ Installation

1 In file *install* :
 Adjust FC, MPIFC, FFT, FFT_inc, FFT_lib

2 Launch *install*

```
$ ./install
```

-> **Executable : (amitex)/libAmitex/bin/amitex_fftp**

```
# Compiler : ifort or gfortran
FC=ifort
FC=gfortran

# MPI compiler
MPIFC=mpif90
#MPIFC=mpiifort
...
FFT=fftw3_f03
...
FFT_inc=<path_to_fftw_include>
FFT_lib=<path_to_fftw_lib>
```

install

❑ In *<my_home>/* :

1 - Install amitex_fftp

Untar the archive

```
$ tar xvf amitex_fftp-vx.x.x.tar
```

In *amitex_fftp*: Adjust *install* -> FC, MPIFC, FFT, FFT_inc, FFT_lib

Launch *install* :

```
$ ./install  
$ ls /libAmitex/bin
```

On linux ubuntu:
FC=gfortran
MPIFC=mpif90
FFT=fftw3_f03
FFT_inc=/usr/include
FFT_lib=/usr/lib64

2 – Launch validation tests

Can be long if the number of available cores is small (typically 20min for 6 cores).

Tip : add an « exit 0 » in *script_tests.sh* after a few tests.

In *amitex_fftp/validation* :

```
$ ./script_test.sh
```

or

```
$ lsubmit script_tests_poincare
```

or

```
$ qsub script_tests_marquises
```

or ...

On a standalone PC

On a cluster

AMITEX_TRAINING DIRECTORY : OVERVIEW

❑ In `<my_home>/` :

amitex_training/

➤ `env_amitex_training.sh` : define environment variables

- To be adjusted to your own configuration
- Before using `amitex_fftp` :



\$ `source env_amitex_training.sh myenv`

➤ 7 directories

- ❑ Scripts : script shell to launch simulation
- ❑ Microstructures : vtk images + matlab/octave files
- ❑ Materials : xml files for material properties
- ❑ Loadings_outputs : xml files for loading and output description
- ❑ Algorithms : xml files for algorithm parameters
- ❑ Behaviors : umat and MFRONT user behaviors
- ❑ Matlab_octave : tools (pre-post processing)

+ `solutions` directories in each directory!

GEOMETRY : UNIT-CELL DESCRIPTION

- **Material** = a set of voxels with same behavior law
(isotropic elasticity, orthotropic elasticity, VonMises plasticity, crystal plasticity etc...)
- **Zone** = **within a material** a set of voxel with the same material coefficients

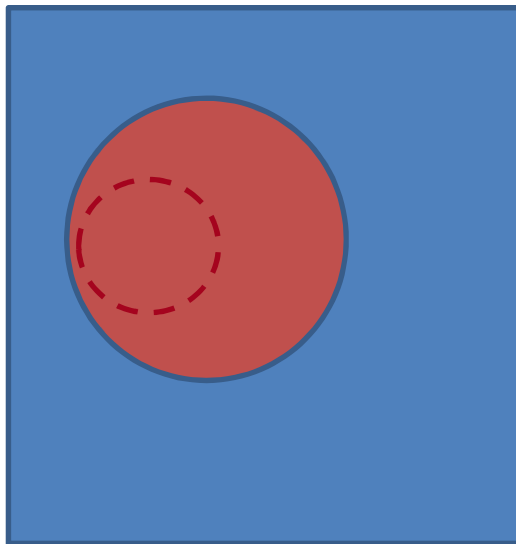
Example

a unit cell with :

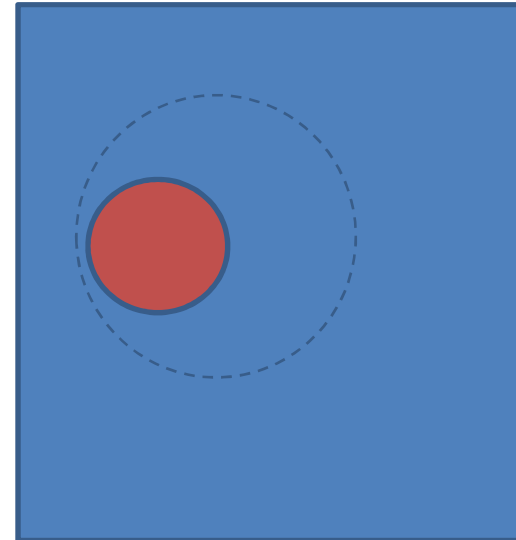
2 materials (1 is plastic and 2 is elastic)

1 zone in material 1

2 zones in material 2 (with different Young moduli)



Material - field

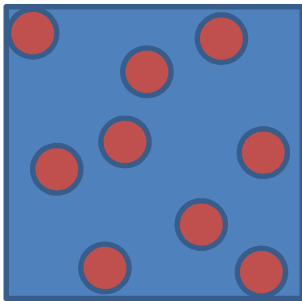


Zone - field

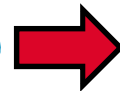
GEOMETRY : UNIT-CELL DESCRIPTION

➤ A non-unique description : an example

Elastic composite (isotropic elasticity)
N inclusions (same elastic coeff.)
Matrix



5 possible
descriptions for
AMITEX



To choose :

- Think 'output'!
 - > Per material average (by default)
 - > Per zone average (on demand)
- Think 'input' (*material.xml* file)!
 - > Avoid large number of materials

❑ 1 material	-	2 zones
❑ 1 material	-	(N+1) zones
❑ 2 materials	-	1 zone / material
❑ 2 materials	-	1 zone / 'inclusion'
❑ N+1 materials	-	1 zone / material

GEOMETRY : INPUT VTK FILES

➤ Exemple 2 materials with Nzones in material 'inclusions'

❑ In *Microstructures/vtk* :

\$ more zone_al2p5_64.vtk

WARNING

Values in vtk file

- Between 1 and N (or 0 and (N-1))
- 1 (or 0) associated to numM=1 in *material.xml*
- 2 (or 1) associated to numM=2 in *material.xml*
- No skipping values (ex : 1,2,4,5)

\$ paraview mat_al2p5_64.vtk &

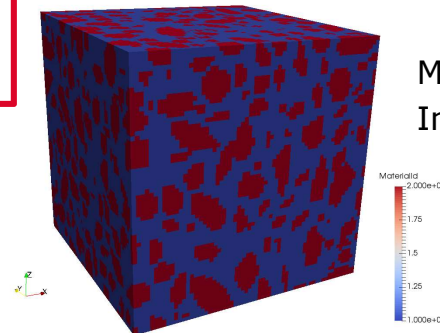
- + Apply
- + Representation menu : « surface »
- + Coloring menu : « MaterialID »

File / open / zone_al2p5_64.vtk

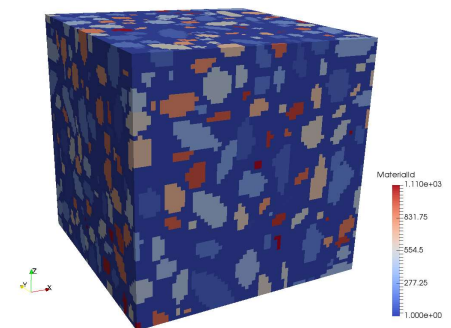
- + Apply
- + Representation menu : « surface »
- + Coloring menu : « MaterialID »

vtk header

```
# vtk DataFile Version 4.5
Material
BINARY
DATASET STRUCTURED_POINTS
DIMENSIONS 65 65 65
ORIGIN 0.000 0.000 0.000
SPACING 1.000000 1.000000 1.000000
CELL_DATA 262144
SCALARS MaterialId short
LOOKUP_TABLE default
```



Matrice : 1
Inclusions : 2



Matrice : 1
Inclusion1 : 1
Inclusion2 : 2
Inclusion3 : 3
Etc...

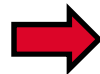
GEOMETRY : INPUT VTK FILES GENERATION

➤ GENERATE .vtk files for amitex_fftp with matlab/octave script files

❑ In *Microstructures/*



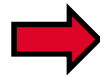
```
$ «my_editor»  
two_spheres.m
```



uses functions in Microstructures/matlab_octave :
gen_grid
savefieldvtk

OCTAVE

```
$ octave  
> two_spheres  
> exit
```

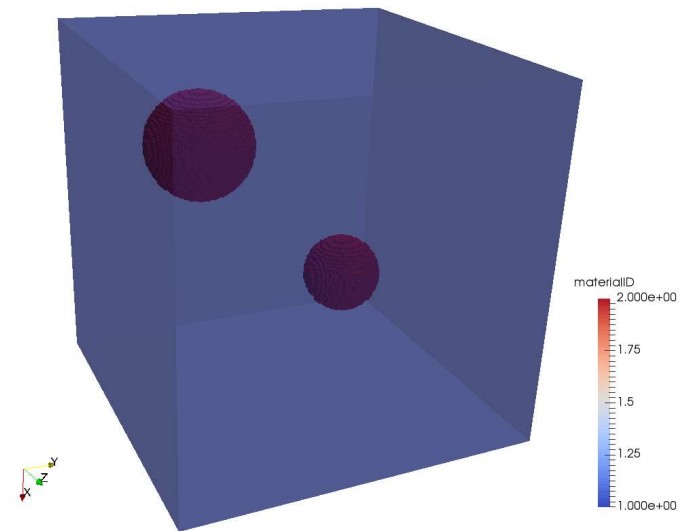
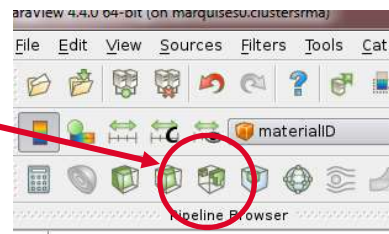


vtk/2spheres.vtk

PARAVIEW

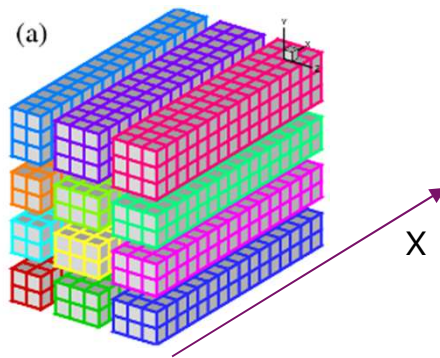
File / Open / *vtk/2spheres.vtk*
+ Apply
+ Representation menu : « surface »
+ Coloring menu : « MaterialID »
+ Opacity menu : 0,5

Filter « threshold »
+ Minimum menu : 1,5
+ Apply



GEOMETRY : 2D INPUT VTK FILES GENERATION

➤ **WARNING** for 2D unit-cells (1 voxel thickness)

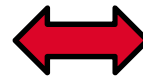


AMITEX uses a 2D MPI decomposition with X-pencils

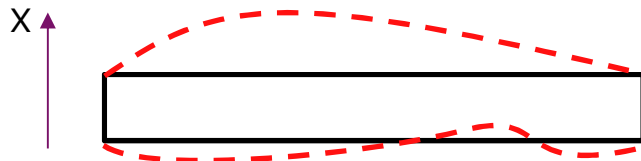
2D unit-cell (1voxel thickness)

➤ **MUST BE DEFINED IN THE (YZ) PLANE!**

2D unit-cell (1voxel thickness) in AMITEX



PLANE STRAIN
or **GENERALIZED PLANE STRAIN**
or **PLANE STRESS** (if simulation
with an additionnal void layer)



Exx non uniform -> violate Periodicity Condition

=> Exx uniform (idem Exy, Exz)

```
$ amitex_fftp -help
```

0/ help (this message)

mpirun (-np n) amitex_fftp -help OR SIMPLY : amitex_fftp -help

1/ general case

mpirun (-np n) amitex_fftp -nm <num_mat.vtk> -nz <num_zone.vtk> -a <algo.xml> -c <load.xml> -m <mat.xml> -s <output>

2/ one material (assumes num_mat.vtk full of 1)

mpirun (-np n) amitex_fftp -nz <num_zone.vtk> -a <algo.xml> -c <load.xml> -m <mat.xml> -s <output>

3/ one zone per material (assumes num_zone.vtk full of 1)

mpirun (-np n) amitex_fftp -nm <num_mat.vtk> -a <algo.xml> -c <load.xml> -m <mat.xml> -s <output>

4/ one material with one zone per voxel (assumes num_zone.vtk varying from 1 to the number of voxels)

assumes also $dx=dy=dz=1$.

mpirun (-np n) amitex_fftp -a <algo.xml> -c <load.xml> -m <mat.xml> -NX nx -NY ny -NZ nz -s <output>

5/ one material with one zone per voxel (assumes num_zone.vtk varying from 1 to the number of voxels)

mpirun (-np n) amitex_fftp -a <algo.xml> -c <load.xml> -m <mat.xml> -NX nx -NY ny -NZ nz -DX dx -DY dy -DZ dz -s <output>

FIRST SIMPLE LINEAR SIMULATION

❑ In *Scripts/*

\$ «my_editor» *script_linear1.sh* &



```
#!/bin/bash
```

```
# Environment variables
```

```
source ../env_amitex.sh
```

Same env. as used to compile *amitex_fftp*

```
MPIRUN="mpirun -np 12"
```

Not always necessary

```
MPIRUN="mpirun"
```

```
# INPUT FILES
```

```
MATEVTK="../Microstructures/vtk/mat_al2p5_64.vtk"
```

```
MATEXML="../Materials/mat_lin.xml"
```

```
LOADXML="../Loadings_outputs/def_imp_mini.xml"
```

```
ALGOXML="../Algorithms/algo_default.xml"
```

```
# LAUNCH SIMU WITH 2 MATERIALS (ONE ZONE/MATERIAL)
```

```
mkdir res_linear1 #don't forget!
```

Crash if directory does not exist

```
$MPIRUN amitex_fftp -nm $MATEVTK -a $ALGOXML -c $LOADXML -m $MATEXML -s res_linear1/res
```

➔ Inputs

\$ «my_editor» /open/ *mat_lin.xml*, *def_imp_mini.xml*, *algo_default.xml*



FIRST SIMPLE LINEAR SIMULATION

❑ In *Scripts/*

Launch simulation

```
$ ./script_linear1.sh
```

➔ Outputs

standard output on screen
systematically :

.log file

.std file

.mstd file

+copy of xml input files



Error, Warning messages

(on demand only :

.vtk files

.zstd files

Uncomment 'Output_vtkList'
in *def_imp_mini.xml*

.log file

To check the number of process

To check the correct reading of vtk and xml files

Error/warning messages

.std file

Unit-cell averages and standard deviations

.mstd file

Per material averages and standard deviations

.zstd files

Per zone averages and standard deviations

.vtk files

Stress, strain, internal variables fields

FIRST SIMPLE LINEAR SIMULATION

❑ Plot macroscopic behavior

➤ In *Scripts* :

```
$ gnuplot
```

```
gnuplot> load "plot1.gp"
```

→ plot "res_linear1/res.std" u 8:2 w l

❑ Apply simple modifications

➤ In *Loadings_outputs/def_imp_mini.xml* :

```
$ cp def_imp_mini.xml def_imp_vtk.xml
```

Uncomment 'Output_vtkList'



vizualize stress/strain fields

➤ In *Loading_outputs/*

```
$ cp def_imp_mini.xml traction.xml
```

In *traction.xml* :

except for xx component

replace "Strain" by "Stress"

assign *Value*="0"

In *script_linear1.sh* and launch simulation



Check results

FIRST SIMPLE LINEAR SIMULATION

□ Apply simple modifications

➤ In *Materials/*

```
$ cp mat_lin.xml mat_pores.xml
```

In *mat_pores.xml* : assign nul elastic properties
do not touch reference material



Effect on the number of iterations
Visualize axial stress field (component 1) -> null stress in inclusions

```
$ cp mat_lin.xml mat_poresb.xml
```

In *mat_poresb.xml* : adjust the reference material properties
Lambda0="2.e10" Mu0="1.35e10"



Effect on the number of iterations

Number of iterations
def_imp_mini 23
traction 25
def_imp_mini + pores ~200
def_imp_mini + poresb ~150

❑ PARSER : FoX library <http://homepages.see.leeds.ac.uk/~earawa/FoX/>

❑ SYNTAX :

Mandatory
Begin Node

Attributes

Input Datas

End Node

```
<?xml version="1.0" encoding="UTF-8"?>
<Loading_Output>

  <!-- OUPUT QUANTITIES -->
  <Output>
    <vtk_StressStrain Strain = "1" Stress = "1"/>
  </Output>

  <!-- SUCCESSIVE LOADINGS AND OUTPUT TIMES -->
  <Loading_Tag="1">
    <Time_Discretization Discretization="User" Nincr="4" />
    <Time_List>2 4.1 8.3 16.1</Time_List>

    <Output_vtkList>
      4
    </Output_vtkList>

    <xx Driving="Strain" Evolution="Linear" Value="0.05"/>
    <yy Driving="Strain" Evolution="Linear" Value="0.015" />
    <zz Driving="Strain" Evolution="Linear" Value="0.015" />
    <xy Driving="Strain" Evolution="Linear" Value="0" />
    <xz Driving="Strain" Evolution="Linear" Value="0" />
    <yz Driving="Strain" Evolution="Linear" Value="0" />
  </Loading>
</Loading_Output>
```

➤ XML (nodes, attributes)
is case sensitive

➤ Input datas are case
insensitive,
except 'path' names

XML FILES : MATERIAL PROPERTIES

- ❑ Behavior evaluation : a UMAT-type procedure called on every voxel, at every iteration

Initial State at time t

$$\sigma^t, \varepsilon^t, \underline{\alpha}^t, T^t, P_{ext}^t$$

Loading increment

$$dt, d\varepsilon, dT, dP_{ext}$$

Material coefficients

$$\underline{c}$$

UMAT

$$\sigma^{t+dt}, \underline{\alpha}^{t+dt}$$

UMAT format



compatibility with CAST3M/ABAQUS and MFRONT

```
SUBROUTINE my_behavior( STRESS, STATEV, DDSDE, SSE, SPD, SCD,&
  RPL, DDSDDT, DRPLDE, DRPLDT,&
  STRAN, DSTRAN, TIME, DTIME,&
  TEMP, DTEMP, PREDEF, DPRED,&
  CMNAME, NDI, NSHR, NTENS, NSTATV,&
  PROPS, NPROPS, COORDS,&
  DROT, PNEWDT, CELENT, DFGRD0, DFGRD1,&
  NOEL, NPT, LAYER, KSPT, KSTEP, KINC )
```

loading.xml

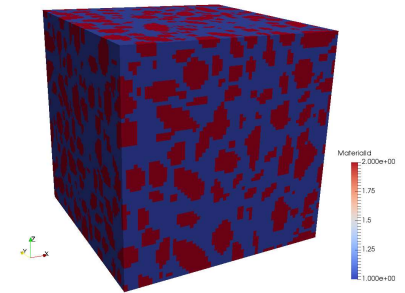
Initialized by *material.xml*

☐ See website

http://www.maisondelasimulation.fr/projects/amitex/general/_build/html/index.html

WARNING : the numM values are going from 1 to N associated
respectively to values 1 to N / or 0 to (N-1) in *mate.vtk*

➡ Prefer defining *mate.vtk* from 1 to N.



15' then correction

➤ In *Materials/*

```
$ cp mat_lin.xml mat_lin_2.xml
```

In *mat_lin_2.xml* : define 1 material with 2 zones, same material properties as *mat_lin.xml* values defined in the xml file

```
$ cp mat_lin_2.xml mat_lin_2b.xml
```

In *mat_lin_2b.xml* : idem
values defined in a ascii files (create *Lambda.txt* and *Mu.txt* in *Microstructures*)

```
$ cp mat_lin_2.xml mat_lin_2c.xml
```

In *mat_lin_2c.xml* : idem
values defined in binary files (use *lambda_mu_bin.m* in *Microstructures*)

➤ In *Scripts/* : *script_linear2.sh* launch simulations

➤ To compare results (should be identical) :

```
$ meld res_linear1/res.std res_linear2/res.std
```

❑ See website

http://www.maisondelasimulation.fr/projects/amitex/general/_build/html/index.html

Exercise

15' then correction

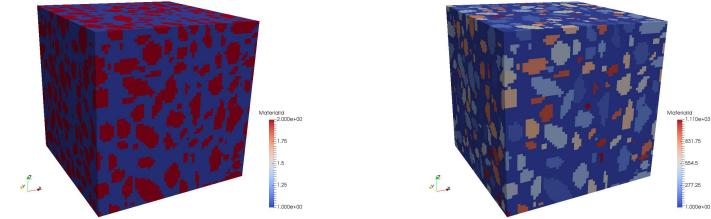
Microstructures/vtk/mat_al2p5_64.vtk and zone_al2p5_64.vtk

Behavior :

Matrix : elasiso

Inclusions : thermoelasiso, $S=C:(E-a(T-T_0))$

$a=PROPS(3)=10^{-5}K^{-1}$ and $T_0=PROPS(4)=0K$



Additional outputs :

stress field (.vtk)

per inclusion average (.zstd)

Loading & output :

Initial temperature 0K

Time 0 to 1000s :

Uniaxial (xx) tensile test : 0->1% (Linear evolution)

Temperature 0K ->1000K (Linear evolution)

100 steps (useless in linear problem, used for exercise!)

Output : vtk stress field at 500s and 1000s

per inclusion average : 3 steps /100

per unit-cell and per material average : 10 steps / 100

Time 1000s to 2000s :

Uniaxial (xx) tensile test : fix 1%

Temperature 1000K -> 500K (Linear evolution)

100 steps (useless in linear problem, used for exercise!)

Output : vtk stress field at 1500s and 2000s

per inclusion average (10 steps / 100)

per unit-cell and per material average : 15 steps / 100

In Scripts : use *script_thermoelas.sh*

In Materials : `$ cp mat_lin.xml mat_thermoelas.xml`

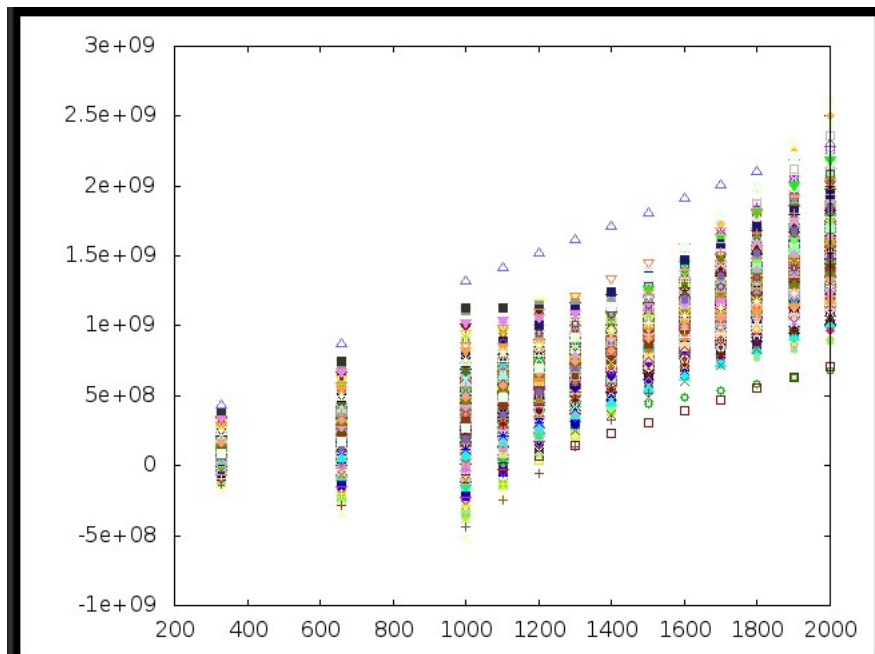
In Loadings_Outputs : `$ cp traction.xml char_thermoelas.xml`

Adjust mat_thermoelas.xml and char_thermoelas.xml

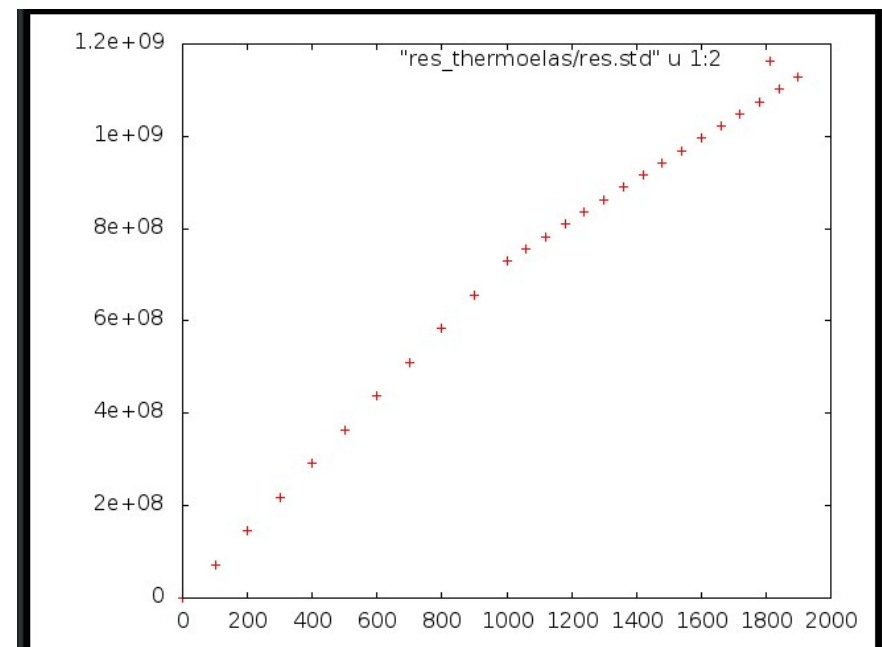
Exercise

15' then correction

- In *Scripts/* use gnuplot
(plot_thermoelas.gp)



#Macroscopic behavior ($s_{11}=f(t)$)
plot "res_thermoelas/res.std" u 1:2



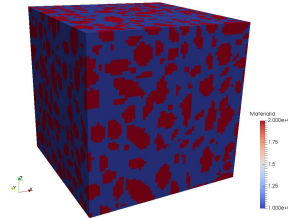
#Per inclusion behavior ($s_{11}=f(t)$)
plot for [j=1:1110] "res_thermoelas/res_2.zstd" every 1110::j-1 u 1:2 notitle

❑ See website

http://www.maisondelasimulation.fr/projects/amitex/general/_build/html/index.html

Exercise

- In *Scripts/* use *script_algo.sh*



Inclusions poreuses (contraste infini!)

- In *Algorithm/*

```
$ cp algo_default.xml algo_nofilter.xml
```



Use the classical Green operator (~5200 iterations vs 157)

```
$ cp algo_default.xml algo_noacv.xml
```



Remove convergence acceleration (~7800 iterations vs 157)

```
$ cp algo_default.xml algo_nofilter_noacv.xml
```



Use the classical Green Operator
&
Remove convergence acceleration
(>10000 iterations!)

❑ See website

http://www.maisondelasimulation.fr/projects/amtex/general/_build/html/index.html

BEHAVIORS : UMAT USER DEFINED

- ❑ Behavior evaluation : a UMAT-type procedure called on every voxel, at every iteration

Initial State at time t

$$\sigma^t, \varepsilon^t, \underline{\alpha}^t, T^t, P_{ext}^t$$

Loading increment

$$dt, d\varepsilon, dT, dP_{ext}$$

Material coefficients

$$\underline{c}$$

UMAT

$$\sigma^{t+dt}, \underline{\alpha}^{t+dt}$$

UMAT format



compatibility with CAST3M/ABAQUS and MFRONT

```
SUBROUTINE my_behavior( STRESS, STATEV, DDSDE, SSE, SPD, SCD,&
  RPL, DDSDDT, DRPLDE, DRPLDT,&
  STRAN, DSTRAN, TIME, DTIME,&
  TEMP, DTEMP, PREDEF, DPRED,&
  CMNAME, NDI, NSHR, NTENS, NSTATV,&
  PROPS, NPROPS, COORDS,&
  DROT, PNEWDT, CELENT, DFGRD0, DFGRD1,&
  NOEL, NPT, LAYER, KSPT, KSTEP, KINC )
```

loading.xml

Initialized by *material.xml*

BEHAVIORS : UMAT USER DEFINED

❑ Behavior evaluation : a UMAT-type procedure is called by each voxel, at each iteration

$\sigma^t = \text{STRESS}$

$\underline{\alpha}^t = \text{STATEV}$

$\varepsilon^t = \text{STRAN}$

$T^t = \text{TEMP}$

$\underline{P}_{ext}^t = \text{PREDEF}$

$dt = \text{DTIME}$

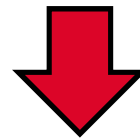
$d\varepsilon = \text{DSTRAN}$

$dT = \text{DTEMP}$

$\underline{dP}_{ext} = \text{DPRED}$

$\underline{c} = \text{PROPS}$

```
SUBROUTINE my_behavior( STRESS, STATEV, DDSDE, SSE, SPD, SCD,&
    RPL, DDSDDT, DRPLDE, DRPLDT,&
    STRAN, DSTRAN, TIME, DTIME,&
    TEMP, DTEMP, PREDEF, DPRED,&
    CMNAME, NDI, NSHR, NTENS, NSTATV,&
    PROPS, NPROPS, COORDS,&
    DROT, PNEWDT, CELENT, DFGRD0, DFGRD1,&
    NOEL, NPT, LAYER, KSPT, KSTEP, KINC )
```



$\sigma^{t+dt} = \text{STRESS}$

$\underline{\alpha}^{t+dt} = \text{STATEV}$

WARNING : for Finite Strain behavior, the mechanical Load increment is given by DFGRD0 and DFGRD1
Instead of STRAN and DSTRAN

➤ Very few 'native' behaviors in AMITEX_FFTP

Why ?

Compatibility with the umat interface (ABAQUS, CAST3M compatible)

-> user-defined behaviors

-> MFRONT defined behaviors

-> A very large collection of standard behaviors (CAST3M compatible)

➤ Native behaviors as simple examples and tests : in **libAmitex/src/materiaux**

Linear behaviors

elasiso.f90

linear isotropic elasticity (small strains – linearized strain)

elasaniso.f90

linear orthotropic elasticity

elasiso_GD.f90

linear isotropic elasticity (finite strains – Green-Lagrange strain)

elasiso_eigs.f90

linear elasticity with eigenstrain

paramextelasiso.f90

linear elasticity + dependance with external parameters

thermoelasiso.f90

linear elasticity + dependance with temperature

Applied stress 'behavior' (for porosity, pressured porosity etc...)

contrainte_imposee.f90

applied stress (6 components)

Non-linear behavior

viscoelas_maxwell.f90

Maxwell visco-elasticity

Exercise : I - **building** your own behavior (fortran)

➤ In *Behaviors/*

1

```
$ cp ../../amitex_fftp-vX.Y.Z/libAmitex/src/materiaux/Makefile ./
```



Adjust **Makefile** (if necessary):

FC=gfortran or ifort

ldecomp=path_to_2decomp

2

```
$ cp ../../amitex_fftp-vX.Y.Z/libAmitex/src/materiaux/elasiso.f90 ./elasiso2.f90
```



Adjust **elasiso2.f90** :

rename function : elasiso-> elasiso2

3

```
$ make clean  
$ make
```




Generate : *libUmatAmitex.so*

Verify that subroutine elasiso2 is contained in *libUmatBehavior.so*

```
$ nm -D libUmatAmitex.so
```

Exercise : II - **testing** your own behavior (fortran) on a single voxel

➤ In *Scripts/*  `script_testelasiso2.sh` ➡ A single voxel submitted to uniaxial tensile test

➤ In *Materials/*

1

```
$ cp mat_lin.xml mat_test_elasiso2.xml
```



Adjust `mat_test_elasiso2.xml`

Lib="path_to_libUmatAmitex.so"

Law="elasiso2"

remove material numM=2

➤ In *Scripts/* `$./script_test_elasiso2.sh`

2

Check results

Young modulus : $E = \mu(3\lambda + 2\mu)/(\lambda + \mu) = 2,7 \cdot 10^{10} (3 \cdot 4 + 2 \cdot 2,7)/(4 + 2,7) = 7,011940 \cdot 10^{10} \text{ Pa}$

=> Uniaxial stress at 0,05 strain : $E \cdot 0,05 = 0,35059701 \cdot 10^{10} \text{ Pa}$

Exercise 1 : I - **building** your own behavior (mfront)

➤ In *Behaviors/*



<http://tfel.sourceforge.net/StandardElastoViscoPlasticityBrick.html>

Example : Standard Elasto-Plastic behavior : *SEVP.mfront*



$$\left\{ \begin{array}{l} \underline{\sigma} = \underline{D} : \underline{\epsilon}^{\text{el}} \\ \underline{\epsilon}^{\text{to}} = \underline{\epsilon}^{\text{el}} + \sum_{i_p=0}^{n_p} \underline{\epsilon}_{i_p}^p + \sum_{i_{vp}=0}^{n_{vp}} \underline{\epsilon}_{i_{vp}}^{vp} \\ \dot{\underline{\epsilon}}^p = \dot{\lambda} \frac{\partial g}{\partial \underline{\sigma}} \end{array} \right.$$

```
@Brick "StandardElastoViscoPlasticity" {
  stress_potential : "Hooke" {young_modulus : 200e3, poisson_ratio : 0.3},
  inelastic_flow : "Plastic" {
    criterion : "Mises",
    isotropic_hardening : "Voce" {R0 : 300, Rinf : 900, b : 1},
    isotropic_hardening : "Voce" {R0 : 0, Rinf : 300, b : 10},
    kinematic_hardening : "Armstrong-Frederick" {C : 1.5e3, D : 5}
  }
};
```

Plasticity criterion : Von Mises

$$g(\underline{\sigma}, p) = \phi \left(\underline{\sigma} - \sum_i \underline{X}_i \right) - \sum_i R_i(p)$$

Isotropic Hardening (Voce)

$$R(p) = R_{\infty} + (R_0 - R_{\infty}) \exp(-bp)$$

Kinematic Hardening (Armstrong-Frederick)

$$\left\{ \begin{array}{l} \underline{X} = \frac{2}{3} C \underline{a} \\ \dot{\underline{a}} = \dot{p} \underline{n} - D \dot{p} \underline{a} \end{array} \right.$$

The von Mises stress is defined by:

$$\sigma_{\text{eq}} = \sqrt{\frac{3}{2} \underline{s} : \underline{s}} = \sqrt{3 J_2}$$

where: - \underline{s} is the deviatoric stress defined as follows:

$$\underline{s} = \underline{\sigma} - \frac{1}{3} \text{tr}(\underline{\sigma}) \underline{I}$$

Exercise 1 : I - **building** your own behavior (mfront)

➤ In *Behaviors/*

- 1 Compile the behavior and store it in a dynamic library (.so)

```
$ mfront --obuild --interface=umat SEVP.mfront
```



Generate :

src/libUmat.Behaviour.so
castem/SEVP.dgibi
include/...

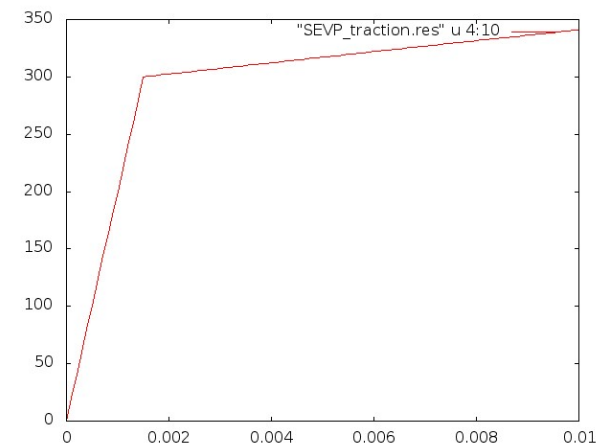
- 2 Test your implementation with « mtest »

```
$ mtest SEVP_traction.mtest
```



Generate : *SEVP_traction.res*

In gnuplot : `plot "SEVP_traction.res" u 2:8 w l`



BEHAVIORS : MFRONT USER DEFINED

Exercise 1 : II - **testing** your own behavior (mfront) on a single voxel

➤ In *Scripts/*  `script_testsevp.sh` ➔ A single voxel submitted to uniaxial tensile test

➤ In *Materials/*

1

`$ cp mat_test_elasiso2.xml mat_test_sevp.xml`



Adjust *mat_test_sevp.xml*

Lib="path_to_libUmatAmitex.so"
Law="umatsevp"

Coeff

IntVar

Ref. Material

(lambda=115384,mu=76923)

** Tridimensional example

** 'OPTION' 'DIMENSION' 3 'MODELISER' 'TRID' ;

coel = 'MOTS' 'YOUN' 'NU' 'RHO' 'ALPH';

statev = 'MOTS' 'EEXX' 'EYY' 'EEZZ' 'EEXY' 'EEXZ' 'EYZ' 'KHXX'
'KHYY' 'KHZZ' 'KHXY' 'KHxz' 'KHYZ' 'P';

➤ In *Behaviors/*

`$ more castem/SEVP.dgibi`



WARNING : The first 4 coefficients are usefull for CAST3M, not used by AMITEX

➤ In *Loadings_Outputs/* `$ cp traction.xml traction_sevp.xml`

2

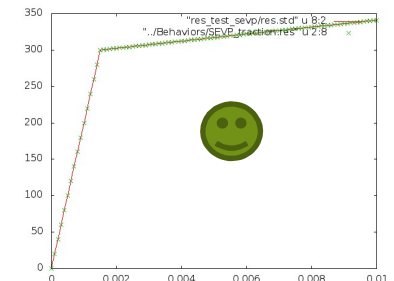


Adjust *traction_sevp.xml*

➤ In *Scripts/* `$./script_testsevp.sh`


3

Gnuplot `plot "res_test_sevp/res.std" u 8:2 w l, "../Behaviors/SEVP_traction.res" u 2:8`



BEHAVIORS : MFRONT USER DEFINED

Exercise 2 : idem with *Behaviors/SEVP_matcoeff.mfront*

 Material coeff. can be modified in .xml files

➤ In *Behaviors/*

1



SEVP_matcoeff.mfront and *SEVP_matcoeff_traction.mtest*


Compile *SEVP_matcoeff.mfront*

Test *SEVP_matcoeff_traction.mtest*

➤ In *Materials/*

2

```
$ cp mat_test_sevp.xml mat_test_sevp2.xml
```

Adjust *mat_test_sevp2.xml* 

➤ In *Scripts/*

3

```
$ ./script_testsevp.sh
```

Gnuplot `plot "res_test_sevp2/res.std" u 8:2 w l, "../Behaviors/SEVP_traction.res" u 2:8`

BEHAVIORS : MFRONT USER DEFINED

Exercise 3 : Application – elasto-plastic matrix with elastic inclusions

(=matrix elasticity)

- In *Scripts/*  *script_sevp2_incl.sh*

1

- In *Materials/*

```
$ cp mat_test_sevp2.xml mat_sevp2_incl.xml
```

2

Adjust *mat_sevp2_incl.xml*



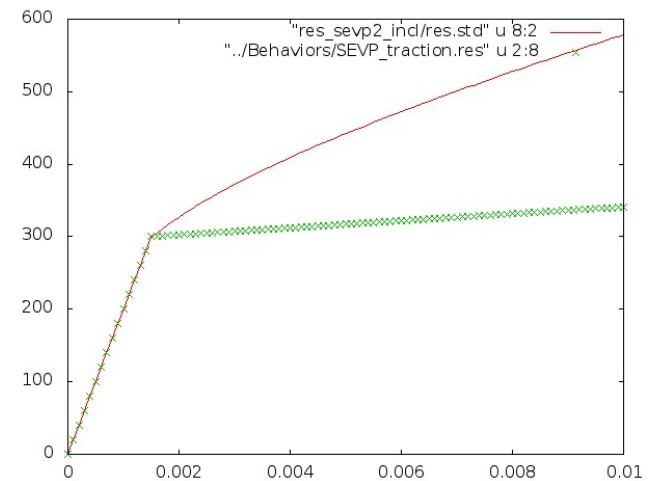
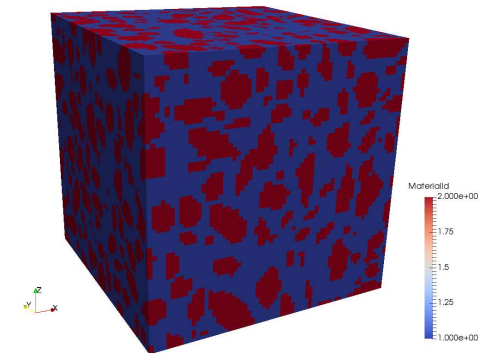
- In *Scripts/*

```
$ ./script_sevp2_incl.sh
```

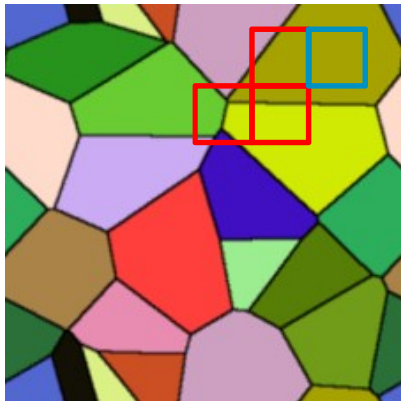
3

Gnuplot

```
plot "res_sevp2_incl/res.std" u 8:2 w l,"../Behaviors/SEVP_traction.res" u 2:8
```



COMPOSITE VOXELS



Homogeneous Voxel



Composite Voxels with N 'phases' (here 2, 2 and 6)

Composite voxel = mixture of different 'phases', characterized by :

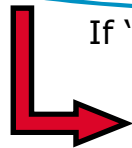
- Its position in the grid (linear indice in the 3D grid)

- The number of phases

- The volume fraction of phases

- The properties (coeff and initial int. Variables) of the phases

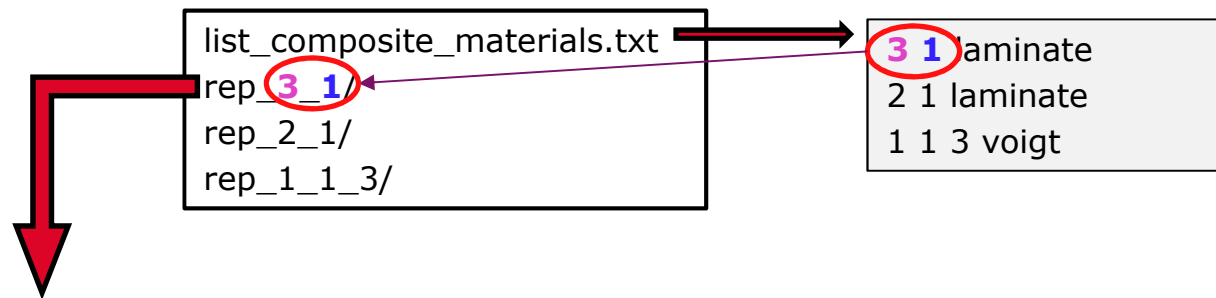
- If 'laminate' homogenization rule : the « normal » and « tangent » vectors



Each phase is associated to a « material » and a
« zone » (in the sense of AMITEX)

COMPOSITE VOXELS

For amitex, the complete definition is given in a folder containing in the example below :



fv1.bin fv2.bin zone1.bin zone2.bin N12x.bin N12y.bin N12z.bin pos.bin Tx.bin Ty.bin Tz.bin

Each binary file collects (one value per composite voxel)

- The volume fraction of phases 1 and 2 (1 for **material *i***, 2 for **material *j*** in *rep_*i*_j*)
- The **zone number** for material *i* and *j*, for respectively phases 1 and 2
- The normal and tangent vector (for 'laminar' homogenization)
- The position (linear index in the 3D grid)

In general, « material » and « zone » numbers correspond to the materialID.vtk and zoneID.vtk used for simulation without composite voxels

Possibility to add « interphase » material or zone (which are too thin to appear in materialID.vtk or zoneID.vtk)

Example : No « interphase » material or zone

1 material with 27 zones + composite voxels NO « INTERPHASE » MATERIAL OR ZONE

```
<?xml version="1.0" encoding="UTF-8"?>
<Materials>
```

```
<!-- REFERENCE MATERIAL -->
```

```
<Reference_Material Lambda0="5.76923076923077e8" Mu0="3.84615384615385e8"/>
```

```
<!-- MATERIAL 1 -->
```

```
<Material numM="1" Lib="/home/gelebart/amitex_fftp/libAmitex/src/materiaux/libUmatAmitex.so" Law="elasiso" >
```

```
  <Coeff Index="1" Type="Constant_Zone" File="materiaux/coefficients/Lambda1_polyx27G_R21.bin" Format="binary"/>
```

```
  <Coeff Index="2" Type="Constant_Zone" File="materiaux/coefficients/Mu1_polyx27G_R21.bin" Format="binary"/>
```

```
  <Coeff_composite Index="1" Type="Constant_Zone" File="materiaux/coefficients/Lambda1_polyx27G_R21.bin" Format="binary" />
```

```
  <Coeff_composite Index="2" Type="Constant_Zone" File="materiaux/coefficients/Mu1_polyx27G_R21.bin" Format="binary"/>
```

```
</Material>
```

Isotropic behavior for modified Newton-Raphson algorithm

~ initial elastic behavior

```
<!-- DIRECTORY FOR THE DEFINITION OF COMPOSITE VOXELS -->
```

```
<Material_composite>
```

```
  <Coeff_composite directory="microstructures/voxcomp/polyX_27G_R21_reuss"/>
```

```
</Material_composite>
```

```
</Materials>
```

Removing this section -> Simulation without any composite voxels

Example 2 : « interphase » material and/or zone

See website

❑ See website

http://www.maisondelasimulation.fr/projects/amitex/user_guide/_build/html/materials.html

Exercise : Generate composite voxels for AMITEX

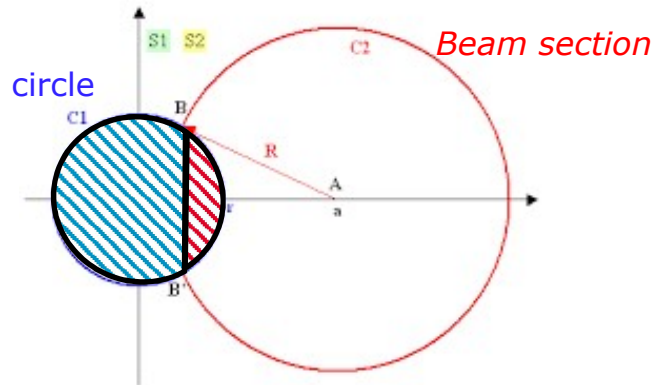
➤ In *Microstructures/*

`cube_cylinder_composite.m`



Volume fraction approximation : intersection disk (around a composite voxel) / disk (beam cross-section)
analytical solution

Voxel circumscribed circle



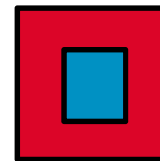
➤ In *Microstructures/*

octave

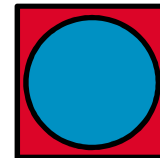
`cube_cylinder_composite`



`Cube64_cylinder_zone.vtk`



`Cube64_cylinder_mate.vtk`



`Cube64_cylinder_composite/` : Directory describing **composite voxels** for



Exercise : use composite voxels in AMITEX

➤ In *Scripts/*  *script_voxcomp.sh*

➤ In *Materials/* `$ cp mat_lin.xml mat_voxcomp.xml`

Adjust *mat_voxcomp.xml*

Add `<Coeff_composite Index=.../>`

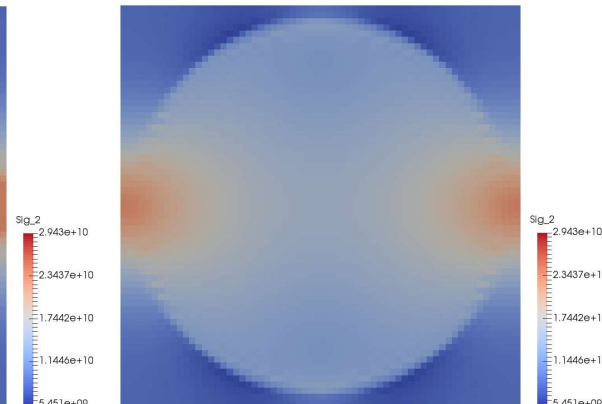
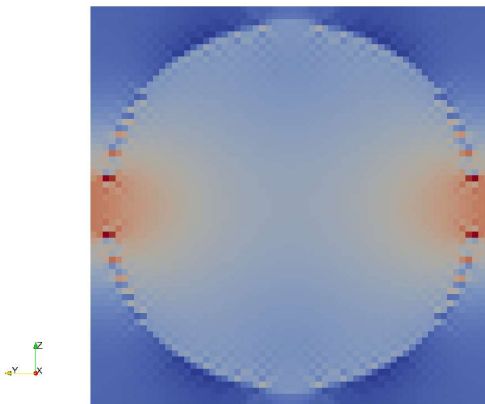
Add `<Material_composite>...<Material_composite/>`

`$ cp mat_lin.xml mat_voxcomp0.xml`

Adjust *mat_voxcomp0.xml*

Comment `<Material_composite>...<Material_composite/>`

➤ In *Scripts/* `$./script_voxcomp.sh`

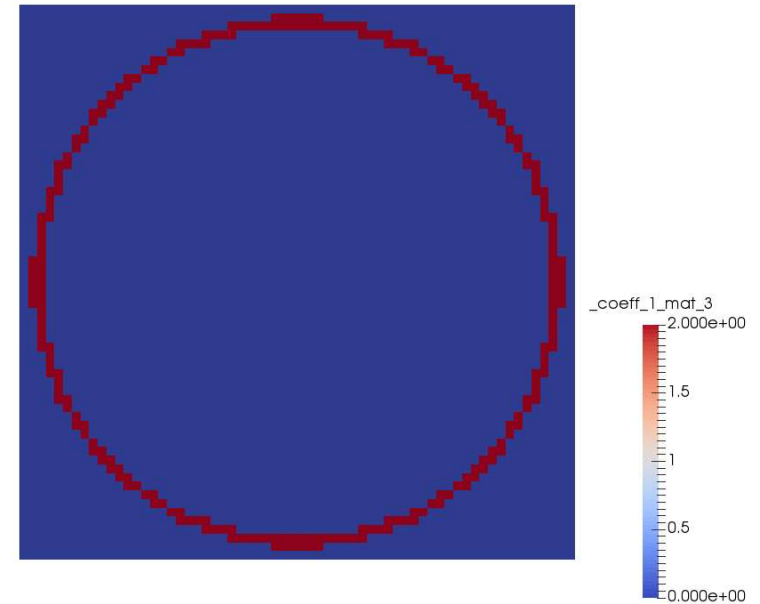


Exercise : use composite voxels in AMITEX

➤ In *Scripts/*  *script_voxcomp.sh*

```
mpirun amitex_fftp -nm $MATEVTK ..... -coeff2print i
```

One vtk file per material with coefficient i



Pour un matériau composite à quoi correspond le coefficient (ou variable interne) i ?
Ici : coeff 1 = nombre de phases



COMPOSITE VOXELS

```

=====
!>    FONCTION UMAT_VOIGT
!!-----
!! integration d'un modele composite de Voigt à N phases (déformation homogène)
!! formalisme HPP
!!
!!-----
!!
!! \param[in]  umatTab Tableau du type dérivé Umatptr
!!             - contient les pointeurs de fonction vers le comportement de chaque phase
!!
!! =====
!! COEFFICIENTS
!! -----
!!
!! Coeff(ncoeff) tableau de coefficients, ordre defini ci-dessous
!!             avec ncoeff = 2N+sum(ncoeffi)
!!
!! 1 - nPhase  nombre de phases dans le matériau composite
!! 2 - ncoeff1  nombre de coeff de chaque loi
!! 3 - ncoeff2
!! .....
!! 1+N - ncoeffN
!!
!! 2+N - FV 1   fractions volumiques de chaque phase
!! 3+N - FV 2
!! ....
!! 2N - FV N-1
!!
!! 1+2N,2N+ncoeff1
!!      Coeff1      coefficients pour le materiau 1
!! 1+2N+ncoeff1, 2N+ncoeff1+ncoeff2
!!      Coeff2      coefficients pour le materiau 2
!! .....
!! .....
!! 1+2N+ncoeff1+...+ncoeffn-1, 2N+ncoeff1+.....+ncoeffN
!!      CoeffN      coefficients pour le materiau n
!!
!! =====

```

```

!! =====
!! VARIABLES INTERNES
!! -----
!!
!! Varint(nvarint) tableau de variables internes, ordre defini ci-dessous
!!             avec nvarint= 7N + sum(nVari)
!!
!! 1 - nvar1      nombre de var. internes de chaque loi
!! 2 - nvar2
!! .....
!! N - nvarN
!!
!! 1+N,6+N
!!      Sig1      Contraintes dans la phase 1 (notation Voigt)
!! 7+N,6+N+nVar1
!!      VarInt1    Variables internes de la phase 1
!! .....
!! .....
!! 1+6*(i-1)+N+nVar1+...+nVari-1,6*i+N+nVar1+....+nVari-1
!!      Sigi      Contraintes dans la phase i (notation Voigt)
!! 1+6*i+N+nVar1+...+nVari-1,6*i+N+nvar1+...+nVari
!!      VarintI    Variables internes de la phase i
!! .....
!! .....
!! 1+6*(N-1)+N+nVar1+...+nVarn-1,6*N+N+nVar1+.....+nVarn-1
!!      SigN      contraintes dans la phase N
!!
!! 1+6*N+N+nVar1+....+nVarn-1,6*N+N+sum(nVari)
!!      VarintN    Variables internes dans la phase N
!!
!! =====
!!
!! \param[out] Sig, Varint
!!
!!-----
!! ATTENTION - ATTENTION - ATTENTION : notation de Voigt,
!!             pour la deformation : x2 sur les termes de cisaillement
!!             soit ici NT0,NT1 et TOT1
!!             pour la contrainte : pas de facteur 2
!!-----
!!
!! Notation CAST3M (Voigt, ordre 11 22 33 12 13 23)
!! =====

```

COMPOSITE VOXELS

```

=====
!>      FONCTION UMATREUSS
!!-----
!! integration d'un modele composite de REUSS a N phases
!!
!! formalisme HPP
!!
!!-----
!! \param[in] dt      Vecteur des pas de temps nouveau et ancien (dt_new,dt_old)
!! \param[in] umatTab Tableau du type dérivé Umatptr contenant les pointeurs
!!                de fonction vers le comportement de chaque phase
!!
!! =====
!!      COEFFICIENTS
!!      -----
!!
!!      Coeff(ncoeff) tableau de coefficients, ordre defini ci-dessous
!!      avec ncoeff = 4*N+ncoeff1+.....+ncoeffn
!!
!!      1  - nPhase          nombre de phases dans le matériau composite
!!      2  - ncoeff1         nombre de coeff de chaque loi
!!      3  - ncoeff2
!!      .....
!!      1+N - ncoeffN
!!
!!      2+N - FV 1           fractions volumiques de chaque phase (n-1 données la nieme vérifiant FV N = 1 - sum(FV I)
!!      3+N - FV 2
!!      ....
!!      2N - FV N-1
!!
!!      2N+1,2N+2
!!      (/ Lambdaeq, Mueq /)      coefficients de Lamé équivalents au comportement élastique initial du matériau 1
!!      2N+3,2N+2+ncoeff1
!!      Coeff1                    coefficients pour le materiau 1
!!      2N+3+ncoeff1,2N+4+ncoeff1
!!      (/ Lambdaeq, Mueq /)      coefficients de Lamé équivalents au comportement élastique initial du matériau 2
!!      2N+5+ncoeff1, 2N+4+ncoeff1+ncoeff2
!!      Coeff2                    coefficients pour le materiau 2
!!      ....
!!      ....
!!      2N+2*(N-1)+ncoeff1+.....+ncoeffn-1+1,4*N+ncoeff1+.....+ncoeffn-1
!!      (/ Lambdaeq, Mueq /)      coefficients de Lamé équivalents au comportement élastique initial du matériau N
!!      4*N+ncoeff1+.....+ncoeffn-1+1,4*N+ncoeff1+.....+ncoeffn
!!      CoeffN                    coefficients pour le materiau n
!!
=====

```

```

=====
!!      VARIABLES INTERNES
!!      -----
!!
!!      Varint(nvarint) tableau de variables internes, ordre defini ci-dessous
!!      avec nvarint= 13N + sum(nVari)
!!
!!      1 - nvar1          nombre de var. internes de chaque loi
!!      2 - nvar2
!!      .....
!!      N - nvarN
!!
!!      1+N,6+N
!!      Def1              composantes de la deformation, phase 1
!!
!!      7+N,12+N
!!      Def1_old          composantes de la deformation pas precedent, phase 1
!!
!!      13+N,12+N+nvar1
!!      VarInt1           variables internes, phase 1
!!
!!      13+N+nvar1,18+N+nvar1
!!      Def2              composantes de la deformation, phase 2
!!
!!      19+N+nvar1,24+N+nvar1
!!      Def2_old          composantes de la deformation pas precedent, phase 2
!!
!!      25+N+nvar1,24+N+nvar1+nvar2
!!      VarInt2           variables internes, phase2
!!      .....
!!
!!      N+12*(N-1)+1+nvar1+...+nvarn-1,N+12*(N-1)+6+nvar1+...+nvarn-1
!!      Defn              composantes de la deformation, phase N
!!
!!      N+12*(N-1)+7+nvar1+...+nvarn-1, N+12*(N-1)+12+nvar1+...+nvarn-1
!!      Defn_old          composantes de la deformation pas precedent, phase N
!!
!!      13*N+nvar1+...+nvarn-1+1,13*N+nvar1+...+nvarn
!!      VarIntn           variables internes, phaseN
!!
=====
!!
!!      autres paramètres d'entrée : voir formalisme UMAT
!!
!! \param[out] Sig, Varint
!!
!!-----
!! ATTENTION - ATTENTION - ATTENTION : notation de Voigt,
!!      pour la deformation : x2 sur les termes de cisaillement
!!      pour la contrainte : pas de facteur 2
!!-----
!!
!! Notation CAST3M (Voigt, ordre 11 22 33 12 13 23)
=====

```

COMPOSITE VOXELS

```

=====
!!>      FONCTION UMATLAMINATE
!!-----
!! integration d'un modele composite LAMINATE a N phases
!!
!! formalisme HPP
!!
!!-----
!!
!! \param[in]  dt      Vecteur des pas de temps nouveau et ancien (dt_new,dt_old)
!! \param[in]  umatTab Tableau du type dérivé Umatptr contenant les pointeurs
!!              de fonction vers le comportement de chaque phase
!!
!!
!! =====
!! COEFFICIENTS
!! -----
!!
!! Coeff(ncoeff) tableau de coefficients, ordre defini ci-dessous
!! avec ncoeff = 4*N+6+ncoeff1+.....+ncoeffn
!!
!!
!! 1 - nPhase          nombre de phases dans le matériau composite
!! 2 - ncoeff1          nombre de coeff de chaque loi
!! 3 - ncoeff2
!! .....
!! 1+N - ncoeffN
!!
!!
!! 2+N - FV 1          fractions volumiques de chaque phase (n-1 données la nieme vérifiant FV N = 1 - sum(FV I)
!! 3+N - FV 2
!! ....
!! 2N - FV N-1
!!
!!
!! 2N+1 - Nx           Composantes du vecteur normal à l'interface
!! 2N+2 - Ny
!! 2N+3 - Nz
!! 2N+4 - Tx           Composantes d'une direction tangeant à l'interface
!! 2N+5 - Ty
!! 2N+6 - Tz
!!
!!
!! 2N+7,2N+8
!! (/ Lambdaeq, Mueq /) coefficients de Lamé équivalents au comportement élastique initial du matériau 1
!! 2N+9,2N+8+ncoeff1
!! Coeff1              coefficients pour le materiau 1
!! 2N+9+ncoeff1,2N+10+ncoeff1
!! (/ Lambdaeq, Mueq /) coefficients de Lamé équivalents au comportement élastique initial du matériau 2
!! 2N+11+ncoeff1, 2N+10+ncoeff1+ncoeff2
!! Coeff2              coefficients pour le materiau 2
!! ....
!! ....
!!
!! 2N+6+2*(N-1)+ncoeff1+.....+ncoeffn-1+1,4*N+6+ncoeff1+.....+ncoeffn-1
!! (/ Lambdaeq, Mueq /) coefficients de Lamé équivalents au comportement élastique initial du matériau N
!! 4*N+6+ncoeff1+.....+ncoeffn-1+1,4*N+6+ncoeff1+.....+ncoeffn
!! CoeffN              coefficients pour le materiau n
!! =====

```

```

=====
!! VARIABLES INTERNES
!! -----
!!
!! Varint(nvarint) tableau de variables internes, ordre defini ci-dessous
!! avec nvarint= 10N + sum(nVari)
!!
!!
!! 1 - nvar1          nombre de var. internes de chaque loi
!! 2 - nvar2
!! .....
!! N - nvarN
!!
!!
!! 1+N,3+N
!! DefA1              composantes de la deformation Anti-Plane
!!                      dans la phase 1 (ordre NN,NT0,NT1, notation Voigt)
!!
!! 4+N,6+N
!! SigP1              composantes de la contrainte Plane
!!                      dans la phase 1 (ordre T0T0,T1T1,T0T1, notation Voigt)
!!
!! 7+N,9+N
!! DefA1_old          composantes de la deformation Anti-Plane
!!                      du pas de temps précédent
!!
!! 10+N,9+N+nvar1
!! VarInt1             variables internes phase 1
!!
!!
!! 10+N+nvar1,12+N+nvar1
!! DefA2              composantes de la deformation Anti-Plane
!!                      dans la phase 2 (ordre NN,NT0,NT1, notation Voigt)
!!
!! 13+N+nvar1,15+N+nvar1
!! SigP2              composantes de la contrainte Plane
!!                      dans la phase 2 (ordre T0T0,T1T1,T0T1, notation Voigt)
!!
!! 16+N+nvar1,18+N+nvar1
!! DefA2_old          composantes de la deformation Anti-Plane
!!                      du pas de temps précédent
!!
!! 19+N+nvar1,18+N+nvar1+nvar2
!! VarInt2             variables internes phase2
!!
!! .....
!! .....
!! .....
!!
!! N+9*(N-1)+1+nvar1+...+nvarn-1,N+9*(N-1)+3+nvar1+...+nvarn-1
!! DefAn              composantes de la deformation Anti-Plane
!!                      dans la phase N (ordre NN,NT0,NT1, notation Voigt)
!!
!! N+9*(N-1)+4+nvar1+...+nvarn-1,N+9*(N-1)+6+nvar1+...+nvarn-1
!! SigPn              composantes de la contrainte Plane
!!                      dans la phase N (ordre T0T0,T1T1,T0T1, notation Voigt)
!!
!! N+9*(N-1)+7+nvar1+...+nvarn-1, N+9*(N-1)+9+nvar1+...+nvarn-1
!! DefAn_old          composantes de la deformation Anti-Plane
!!                      du pas de temps précédent
!!
!! 10*N+nvar1+...+nvarn-1+1,10*N+nvar1+...+nvarn
!!
!! =====
!! ATTENTION - ATTENTION - ATTENTION => voir Voigt, Reuss
!! =====

```


Exercise : use composite voxels in AMITEX

➤ In *Scripts/*  *script_voxcomp.sh*

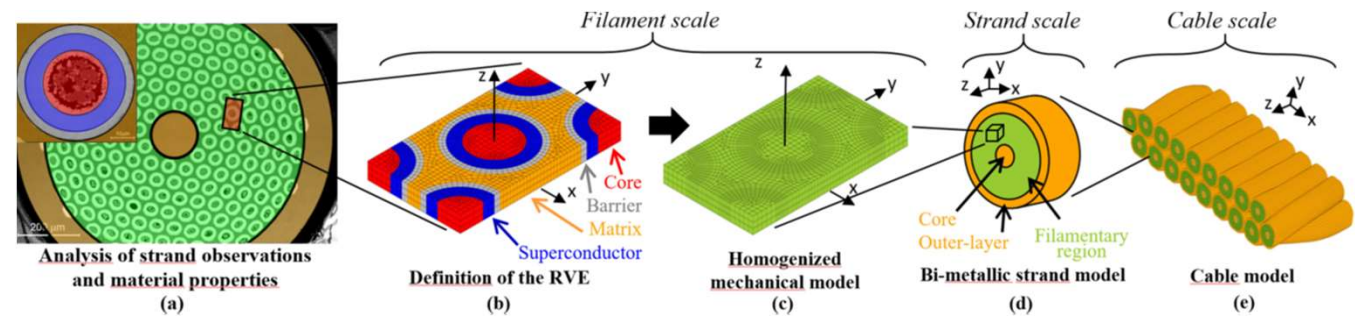
BEAM / PLATE APPLICATION

Homogeneous beam with circular cross-section

Homogeneous beam with circular cross-section

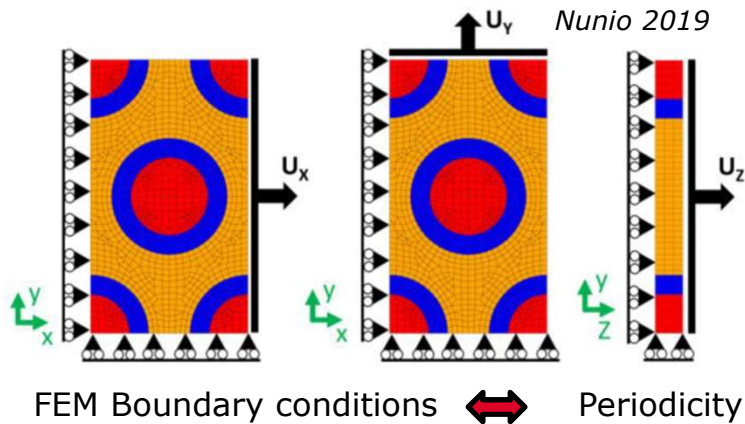
TP – SUPERCONDUCTING CABLES

- Context : superconducting cables (for magnets)



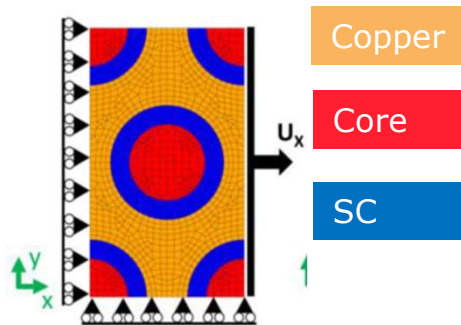
Nunio 2019

- Filament scale



- 1 - Create the microstructure
- 2 - Perform elastic simulation with arbitrary properties
 - uniaxial tensile test in X
 - applied axial strain 0,01
- 3- Perform elasto-plastic simulations

TP – SUPERCONDUCTING CABLES



COMPONENTS	$E(\text{GPa})$	$\sigma_y(\text{MPa})$	$C(\text{MPa})$	γ
Copper	129	39	35960	310
Filament Core	3	-	-	-
SC region (Nb ₃ Sn SG)	171	-	-	-

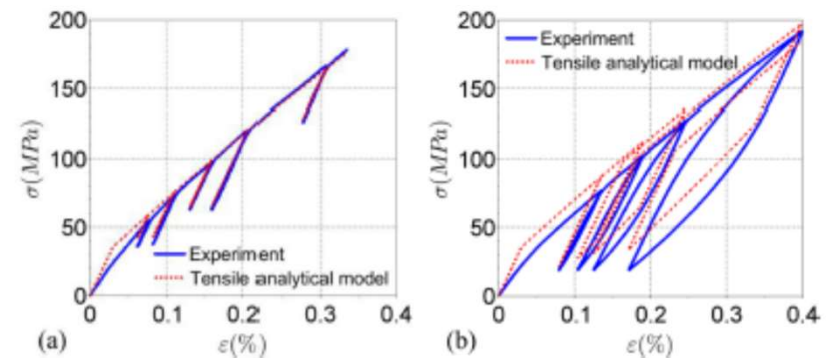
Kinematic Hardening

$$\dot{X} = C\dot{\epsilon}_p - \gamma X\dot{p}$$

Lenoir 2019

1 – Introduce non-linear behavior with the given properties

2 – Simulate loading-unloading



Lenoir 2019

DE LA RECHERCHE À L'INDUSTRIE



TP – PARALLEL COMPUTING

Pour ces 3 exercices : utiliser `elasiso.f90` comme point de départ (pour chaque comportement : renommer le fichier et le nom de la fonction)

Dégradation du module élastique avec le temps = $E = E0.\exp(-t/\tau)+E1$

Gonflement d'une phase avec un paramètre extérieur `Pext` (ex : fluence ~qté d'irradiation reçue par le matériau)

`Sig = c(esp-eps_g)`

`Eps_g = Pext.Eps0.Id`

Appliquer un chargement `Pext` (0->1) + plateau + rampe (1 -> 0)

Même simulation avec la Température :

`Sig = c(esp-eps_g)`

`Eps_g = Pext.Eps0.Id`

Appliquer un chargement `Pext` (0->1) + plateau + rampe (1 -> 0)

Changement de trajet de chargement (avec comportement cinématique 'induit') :

application traction/pression interne tube

loi MFRONT : plasticité avec faible écrouissage isotrope

chargement 1 : pression interne avec effet de fond : pilotage `eps_yy` 0%->1%
+ maintien `sig_zz=0,5*sig_yy` (autres composantes : contrainte = 0)

chargement 2 : traction uniaxiale : `eps_zz` 0->1% (autres composantes : contrainte = 0)
traction uniaxiale : `eps_zz` 1% -> 0% (autres composantes : contrainte = 0)

pression interne avec effet de fond : pilotage `eps_yy` 0%->1%
+ maintien `sig_zz=0,5*sig_yy` (autres composantes : contrainte = 0)

Tube sollicité en pression interne : effet des voxels composites

Composite à N inclusions avec propriétés variables de chaque inclusion

Propriétés (coeff. Ou var. internes) continuellement variables
coeff. Continuellement variable (cas 1 matériau / cas 2 matériaux)
cas 2 matériaux variable (cas 1 matériau / cas 2 matériaux)

Grandes transformations : rotation pure

Mise en évidence écrouissage cinématique induit par des inclusions élastiques