

Un modèle objet avancé de splitting d'opérateur Application à l'élastoplasticité en géotechnique

Dominique Eyheramendy

JOURNEES SCIENTIFIQUES MOMAS
CIRM / 28-30 Novembre 2005

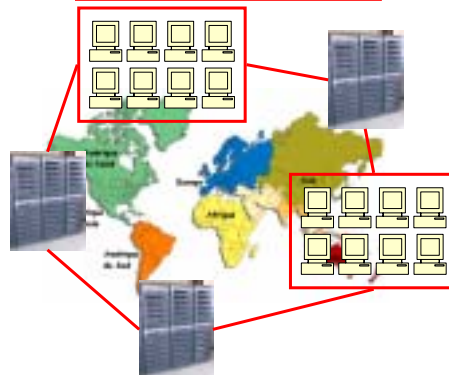
Institut Camille Jordan (UMR 5208) - Center for the Development of Parallel Scientific Computing
ISTIL – Université Claude Bernard Lyon 1 - France

Real life problem

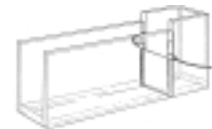


Debris flows

<http://vcmc.univ-lyon1.fr>

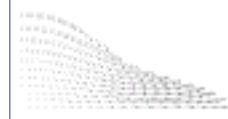


Experimentation



Martin & Moyce 1952

Simulation



Eyheramendy & Zimmermann 2001

Mathematical Model

IBVP

$$\begin{aligned} T_{ij} + f_j &= \rho (u_{i,j} + u_j \mu_{i,j}) && \text{on } \Omega \\ u_{i,j} &= 0 && \text{on } \Omega \\ T_{ij} \mu_j &= F_i && \text{on } \partial^e \Omega \\ u &= \bar{u} && \text{on } \partial^d \Omega \\ T_{ij} &= -p \delta_{ij} + \tau_{ij} && \text{on } \Omega \\ \tau_{ij} &&& \text{Constitutive relationship} \\ &&& + \text{I.C.} \end{aligned}$$

Numerical Model

Finite Elements discrete formulation


Given f , find $(u, q, p) \in (W_0 \times P_0)$ such that :

$$\begin{aligned} \int_{\Omega} \rho (u_i \delta_{ij} + u_j \mu_{ij}) v_i - \int_{\Omega} \rho (u_i \delta_{ij} + u_j \mu_{ij}) v_j + \int_{\Omega} p v_i \delta_{ij} &= \int_{\Omega} f_i v_i \\ + \sum_{\Gamma} \int_{\Gamma} (u_i \delta_{ij} + u_j \mu_{ij}) \tau_{ij} - \tau_{ij} (u_i \delta_{ij} + u_j \mu_{ij}) &= 0 \\ + \sum_{\Gamma} \int_{\Gamma} p \delta_{ij} v_i \delta_{ij} &= 0 \end{aligned}$$

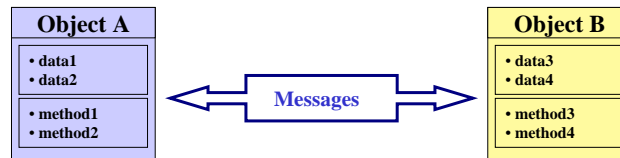
Outline

1. What are the needs for future scientific computing?
 - Problem position
 - Basic ideas
 - The object-oriented paradigm
2. Modeling and Algorithmic Complexity Handling
 - The elastoplastic problem: J_2 plasticity equations
 - An operator split model
 - The object model for material modeling
 - Inner class concept
 - Enhanced data organization to enforce modeling
 - Interface concept
 - Algorithmic consistency enforcement: explicit algorithm equations for material
 - Numerical example: bearing capacity of a strip footing
3. Illustration of the wide range of applicability
 - Stokes and Navier-Stokes flow through a porous media (periodic flow)
4. Conclusion

What are the needs for scientific computing?

- Today tendency in the research and industry
 - Solve complex problems on heterogeneous complex computer systems
- multi-physics** **multi-scale** **multi-processors**
- Assurance that a model as embodied in a computer code is a correct representation of the process or system for which it is intended (validation of code)
 - Assurance that a computer code correctly performs the operations specified in a numerical model (verification of code)
 - need for a high level of automation for the computations with a high reliability
 - Basic idea
 - introduction of homogeneous code capable to handle this global complexity: models, algorithms, hardware system (including multi-processing and networking)
 -  Advanced object-oriented model in Java to introduce mathematical modeling and numerical considerations in scientific computing tools

- Basic definition of object-oriented programming
 - Hierarchical organization of classes with inheritance
 - Encapsulation of data and behavior inside the Object
 - Communication between objects by sending messages (polymorphism)



- Application of the O.O.P. to the FEM ([Rehak 1989], [Zimmermann & al. 1990]...)
 - Enhancement of the modularity of FE codes: Better maintainability of FE codes
 - Better extensibility of FE codes
 - Natural representation of algorithms and easy manipulation of elemental contributions
 - Widely adopted today in the scientific computing community (applied mathematics, computational mechanics,)

Nonlinear Material Modeling: Enforcing numerical consistency

□ The Problem

Find u and σ with appropriate regularity conditions such that :

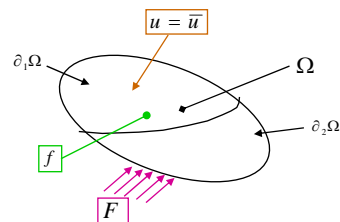
$$\sigma_{ij,j} + f_i = 0 \quad \text{on } \Omega$$

$$\sigma_{ij} n_j = F_i \quad \text{on } \partial_2 \Omega$$

$$u_i = \bar{u}_i \quad \text{on } \partial_1 \Omega$$

$$\mathcal{E}_{kl}(u) = \frac{1}{2}(u_{k,l} + u_{l,k}) \quad \text{on } \Omega$$

+ Constitutive Law : J_2 plasticity



□ The constitutive law: linear elastic / perfectly plastic

- Additive decomposition of the strain (see e.g. [Chaboche & Lemaître 1996]): $\dot{\epsilon} = \dot{\epsilon}^e + \dot{\epsilon}^p$
- Yield function – Mises' criterion: $f(\sigma) = \sqrt{J_2} - k$
- Flow rule: $\dot{\epsilon}^p = \dot{\gamma} r(\sigma)$ with flow: $r(\sigma) = \frac{dq}{d\sigma}$
- Associated constitutive law: $q(\sigma) = f(\sigma)$

- Expressions of plastic multiplier and elasto-plastic tangent operator:

$$\dot{\gamma} = \frac{df^T}{d\sigma} D^{el} d\epsilon \quad D^{ep} = D^{el} - \frac{(D^{el} \frac{dq}{d\sigma}) (D^{el} \frac{df}{d\sigma})^T}{\frac{df^T}{d\sigma} D^{el} \frac{dq}{d\sigma}}$$

Elastic-Plastic Operator Split

- Principle

– Consider the IBVP: find $x(t) \in \mathfrak{R}^N$

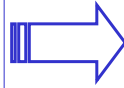
$$\begin{cases} \dot{x}(t) = Ax(t) \\ x(t_n) = x_n \end{cases}$$

– Where: $A: \mathfrak{R}^N \times \mathfrak{R}^N$

$$A = A^1 + A^2 \quad t_{n+1} = t_n + h$$

Solution

$$x(t_{n+1}) = \exp((A^1 + A^2)h) x_n$$



Problem 1

Problem 2

$$\begin{cases} \dot{\bar{x}}(t) = A^1 x(t) \\ \bar{x}(t_n) = x_n \end{cases} \quad \begin{cases} \dot{x}(t) = A^2 x(t) \\ x(t_n) = \bar{x}(t_{n+1}) \end{cases}$$

Approximated Solution

$$x_{n+1} = \exp(A^1 h) \exp(A^2 h) x_n$$

- Application to elastoplasticity

Elastoplastic

Elastic predictor

Plastic corrector

$$\dot{\varepsilon} = \nabla^S(\Delta \dot{u})$$

$$\dot{\varepsilon} = \nabla^S(\Delta \dot{u})$$

$$\dot{\varepsilon} = 0$$

$$\dot{\varepsilon}^p = \gamma \frac{\partial f}{\partial \sigma}$$

$$= \dot{\varepsilon}^p = 0$$

+

$$\dot{\varepsilon}^p = \gamma \frac{\partial f}{\partial \sigma}$$

Objects for constitutive modeling

Material Model

J2 plasticity

• properties:

- Yield stress σ_y
- Young modulus E
- Poisson ratio ν

- Properties management

Criterion: Mises

• value $f(\sigma) = \sqrt{J_2} - k$

• first order derivative value

$$\frac{df(\sigma)}{d\sigma}$$

Equations Model

Perfect plasticity

- Criterion: Mises function
- Integrator: radial return

Internal variables:

- plastic strain $d\varepsilon^p$
(plastic multiplier $d\gamma$ used here)

- plastic correction computation

$$d\sigma^p = -D^d d\gamma \frac{df(\sigma_{n+1}^t)}{d\sigma}$$

$$d\gamma = \frac{f(\sigma_{n+1}^t)}{\frac{df}{d\sigma} - D^d \frac{d\gamma}{d\sigma}}$$

- constitutive matrix (tangent elasto-plastic or elastic)

$$D^{\sigma} = D^e - \frac{(D^e \frac{d\gamma}{d\sigma}) (D^d \frac{df}{d\sigma})}{\frac{df}{d\sigma} - D^d \frac{d\gamma}{d\sigma}}$$

- determination of plastic range at a Gauss Point

- material integration at a Gauss Point (integrator)

Numerical integrator

Explicit Scheme

• Equations model: perfectly plastic

- Radial return

Problem at iteration i and step $n+1$:
Given σ_n and $d\varepsilon_{n+1} = B \Delta d^i$, find σ_{n+1}

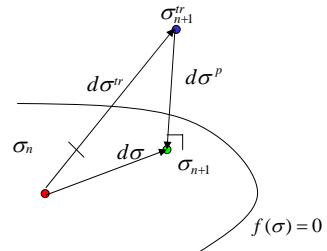
• Compute trial stress (elastic predictor)

$$\sigma_{n+1}^{tr} = \sigma_n + d\sigma^e = \sigma_n + D^e d\varepsilon_{n+1}$$

• If plastic behavior at Gauss Point

Then $\sigma_{n+1} = \sigma_{n+1}^{tr}$

Else compute plastic correction



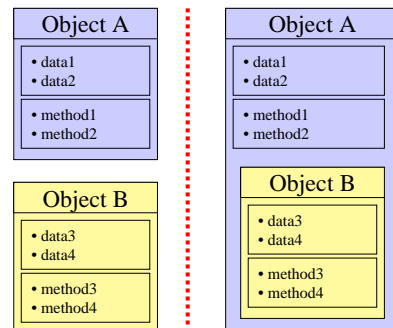
Partial view of the O.O. hierarchical architecture for constitutive law modeling

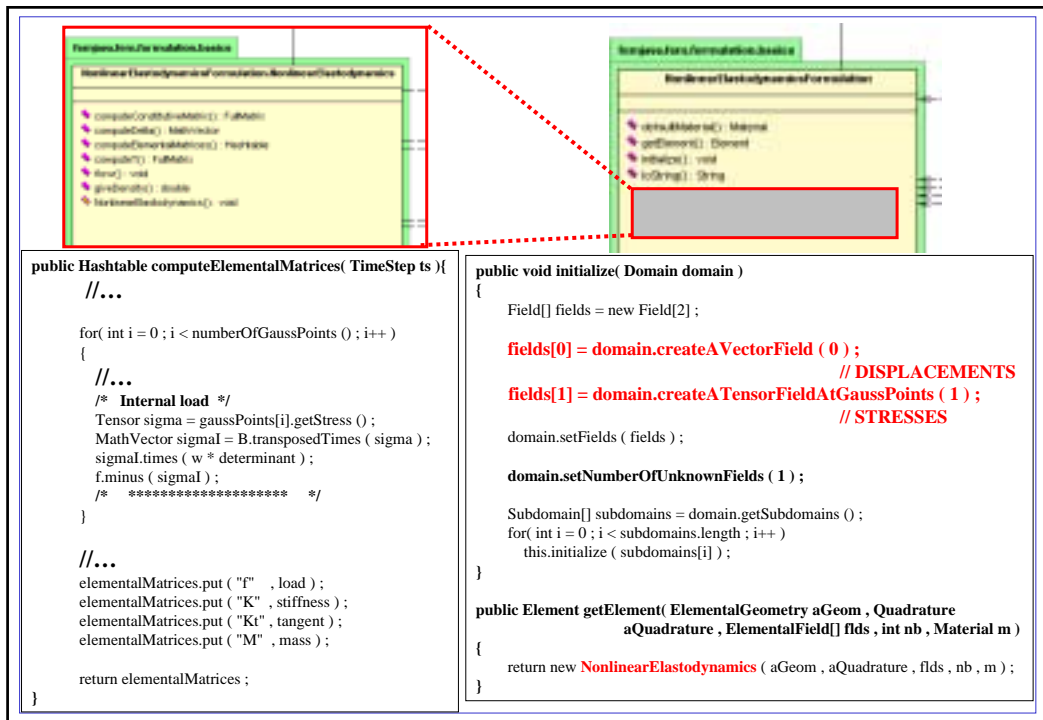


- The object Material and subclasses:
 - management of the physical data for a given material
 - The object Criterion and subclasses:
 - Computation of the criterion value and its partial derivative values
 - The object Behavior and subclasses:
 - management of the computations needed by the integration algorithm (plastic corrector computation, computation of the elastoplastic tangent operator...)
 - The object Integrator and subclasses:
 - numerical integration algorithm
- The behavior LinearElasticPerfectlyPlastic can be integrated by the integrator RadialReturn

Object-oriented Data Organization in Java

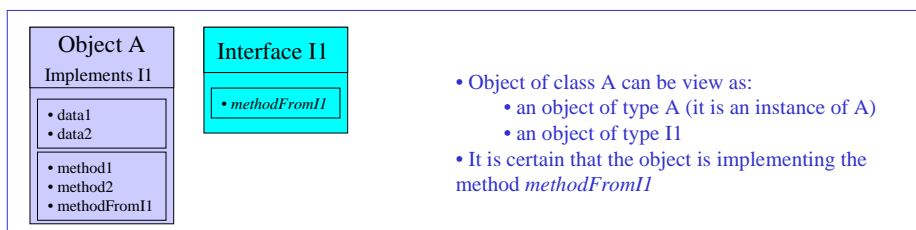
- A typical mechanism of O.O organization proposed in Java
 - Define a class inside another class
 - Local sharing of data
 - Define coherent data at the same physical place
- Application to Finite Elements
 - data organization for multi-fields computations
 - strong decoupling between data and algorithms
- Illustrated on elastoplasticity
 - The nonlinear formulation
 - Discrete variational formulation
 - The iterative solution scheme
 - The constitutive equations and the numerical integration





Objects Specification in Java: The interface mechanism

- A typical mechanism for objects proposed in Java
 - Interface concept proposed in Java
 - An interface is a user-defined data type which allows functionalities specification
 - May be view it an abstract class
 - No implementation, no attributes
 - All the implementation of an interface is done at the level of the class which implements it



- Application to elastoplasticity
 - The same data may be used in various algorithmic contexts
 - Enforcement of numerical consistency in the F.E code

Functionality specification: the interface concept

- The behavior: LinearElasticPerfectlyPlastic

LinearElasticPerfectlyPlastic implements the interface **RadialReturnIntegrable**



- The behavior: LinearElasticPerfectlyPlastic

- The method `computePlasticCorrection()` implemented of the interface `RadialReturnIntegrable` (explicit integration of the constitutive equations)

```

public void computePlasticCorrection( Tensor trialStress , GaussPoint gp ,
                                     Element element )
{
    Tensor DfDsigma = criterion.firstOrderDerivative ( trialStress );
    FullMatrix Del = element.computeConstitutiveMatrix ( );
    double k = material.getProperty ( "sigy" ) / Math.sqrt ( 3.0 );
    double f = criterion.value ( trialStress , k );

    double dGamma = f / ( DfDsigma.times ( Del.times ( DfDsigma ) ) );

    Tensor dEpsp = DfDsigma.times ( dGamma );

    trialStress.minus ( Del.times ( dEpsp ) );

    gp.plastic ( );
    gp.setDGamma ( dGamma );
}
    
```

$$d\gamma = \frac{f(\sigma_{n+1}^{tr})}{\frac{df}{d\sigma}^T D^{el} \frac{dq}{d\sigma}}$$

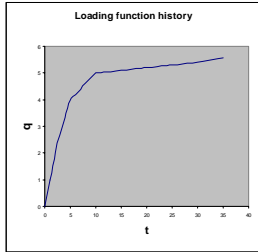
$$d\epsilon^p = d\gamma \frac{dq}{d\sigma}$$

$$\sigma_{n+1}^{tr} = \sigma_n^{tr} - D^{el} d\epsilon^p$$

where : $d\sigma^p = D^{el} d\epsilon^p$

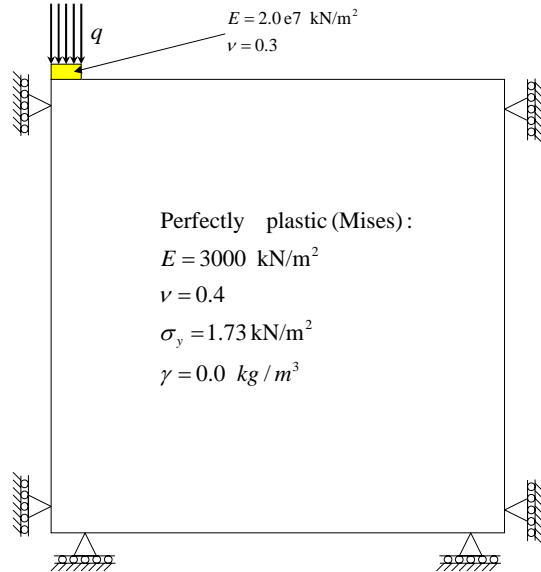
Application to Strip footing

- Footing problem
 - see [Commend & Zimmermann 2001]
 - Quadrilateral elements
 - Locking?



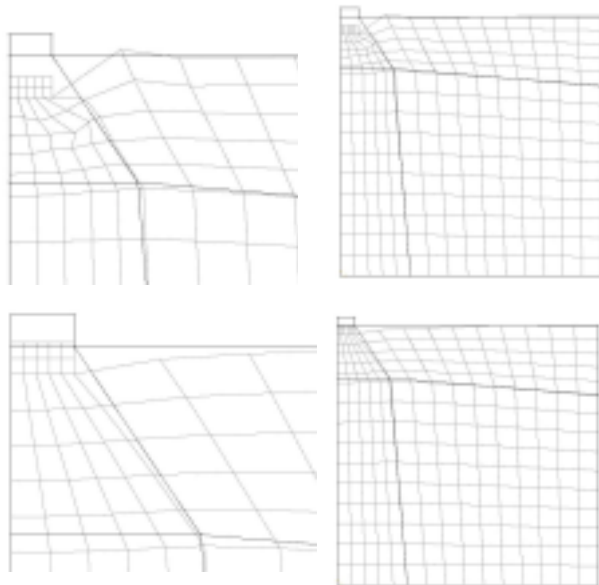
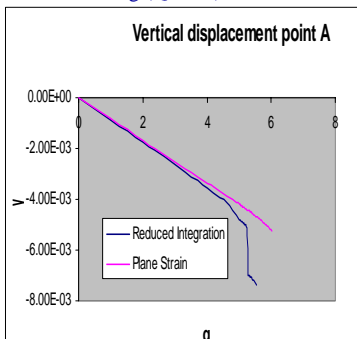
- Theoretical failure load based on a Mohr-Coulomb yield surface (Terzaghi 1951)

$$q_f \approx 5 \text{ kN / m}$$

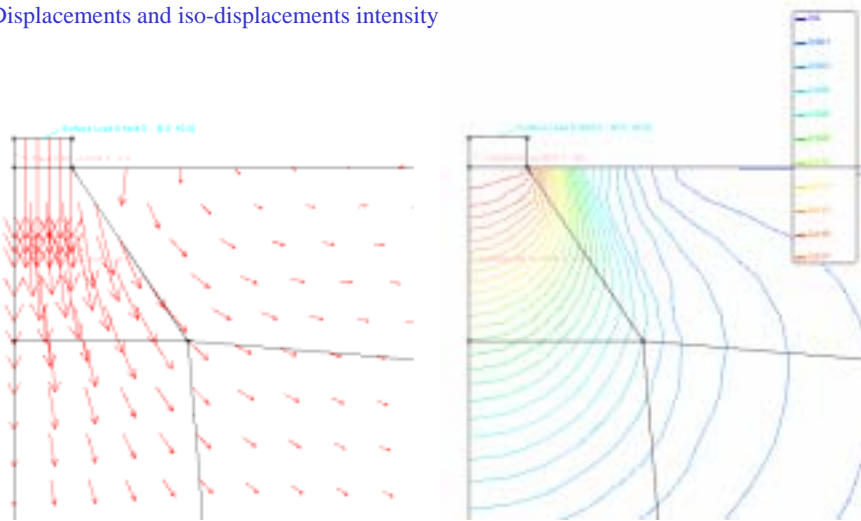


Volumetric locking for quadrilateral elements

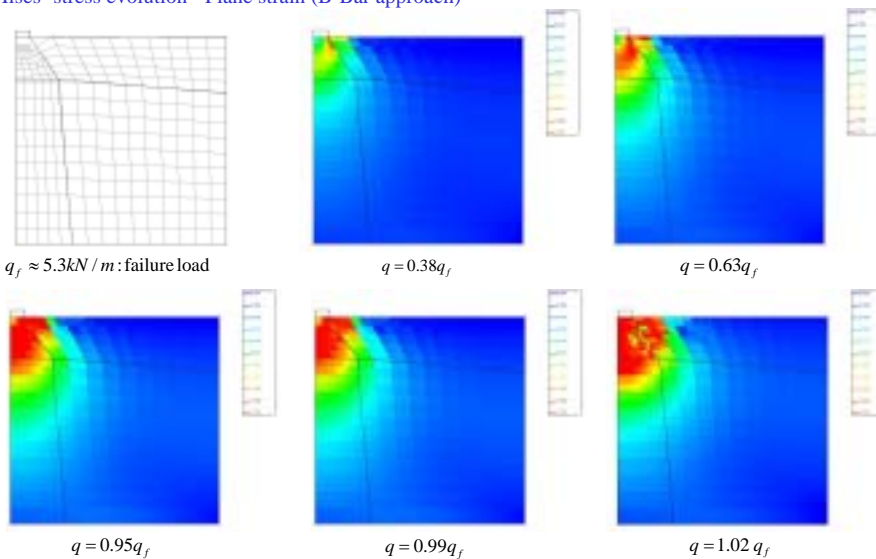
- Plane strain
 - Reduced integration (Dilatational part – B-Bar approach)
 - Locking (QUA 4)



- Displacements and iso-displacements intensity



- Mises' stress evolution - Plane strain (B-Bar approach)

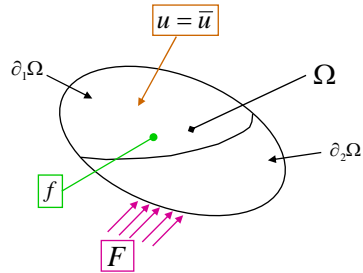


- see [Chiumenti & al. 2004] and [Commend & al. 2004] for more robust approaches to deal with incompressibility

Application to the Incompressible Navier-Stokes Problem

- Steady-state Incompressible Navier-Stokes equations

$$\begin{aligned} \sigma_{ij,j} + f_i &= \rho u_i u_{i,j} && \text{on } \Omega \\ u_{i,j} &= 0 && \text{on } \Omega \\ \sigma_{ij} n_j &= F_i && \text{on } \partial_2 \Omega \\ u_i &= u_i && \text{on } \partial_1 \Omega \\ \sigma_{ij} &= -p \delta_{ij} + 2\mu \varepsilon_{ij}(u) && \text{on } \Omega \\ \varepsilon_{ij}(u) &= \frac{1}{2}(u_{i,j} + u_{j,i}) && \text{on } \Omega \end{aligned}$$



- Formulation :

- Galerkin formulation, stabilized by adding least-squares type terms (PSPG/SUPG see [Tezduyar 2004] for more detail)

- Given f , find $(u^h, p^h) \in ((S^h)_n \times (P^h)_n)$ such that for each $(w^h, q^h) \in ((W^h)_n \times (P^h)_n)$, one has :

$$\begin{aligned} & \int_{\Omega} \rho u_j^h u_{i,j}^h dv - \int_{\Omega} 2\mu \varepsilon_{ij}(u^h) \varepsilon_{ij}(w^h) dv + \int_{\Omega} p_h w_{i,i}^h dv + \int_{\Omega} u_{i,i}^h q^h dv - \int_{\Omega} f_i w_i^h dv \\ & + \sum_{\Omega^r \in \Omega^h} \left[\int_{\Omega^r} (\rho u_j^h u_{i,j}^h - 2\mu \varepsilon_{ij,j}(u^h) + p_{,i}^h - f_i) \tau_{mom}(\rho u_j^h w_{i,j}^h + q_i^h) dv \right] = 0 \end{aligned}$$

• Equal order interpolation

$$\bullet \tau_{mom} = \left(\left(\frac{2|u|}{h} \right)^2 + \left(\frac{4\mu}{h^2} \right)^2 \right)^{-1/2}$$

```

public Hashtable computeElementalMatrices( ts ){
//...
for( int i = 0; i < numberOfGaussPoints(); i++)
{
//...
// Computation of the elemental
// contribution at each Gauss Point
}
//...
elementalMatrices.put( "F" , load );
elementalMatrices.put( "K" , stiffness );

return elementalMatrices ;
}
                
```

```

public void initialize( Domain domain )
{
Field[] fields = new Field[2];

fields[0] = domain.createAVectorField( 0 ); // VELOCITY
fields[1] = domain.createAScalarField( 1 ); // PRESSURE

domain.setFields( fields );

domain.setNumberOfUnknownFields( 2 );

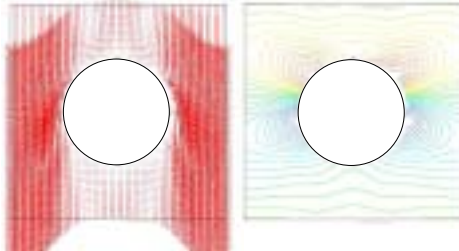
super.initialize();
}

public Element getElement( ElementalGeometry aGeom, Quadrature
aQuadrature, ElementalField[] flds, int nb, Material m )
{
return new UnsteadyStabilizedNavierStokes( g, gps, flds, nb, m );
}
                
```

Application to Periodic Flow

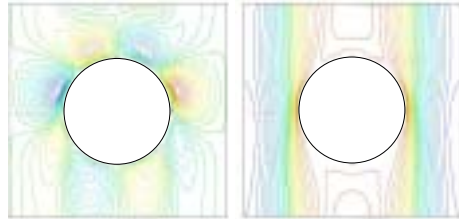
- Objective: derivation of the flow laws of Newtonian and non-Newtonian fluid through fibrous media
 - Collaboration : C. Geindreau & L. Orgéas, L3S-Grenoble

Velocity **-Re=78.3 - 20000 dofs-** Pressure

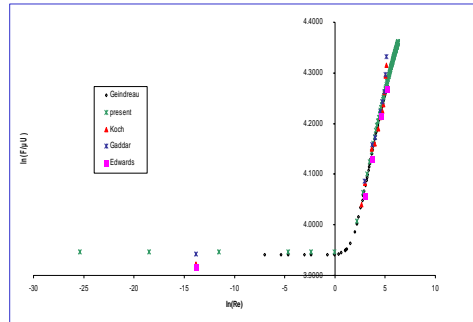


Velocity (x-dir)

Velocity (y-dir)

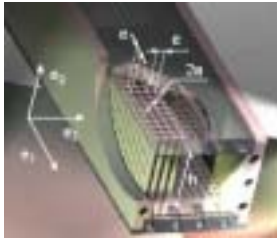


- a micro-macro technique: homogenization of periodic structures
- boundary value problems directly deduced from the upscaling process to obtain the macroscopic flow law (see [Auriault & al. 2002] and [Idris & al. 2004])

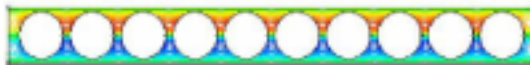
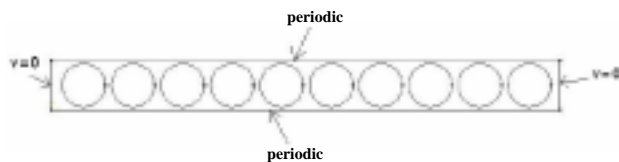


- Perspectives: power law constitutive law (viscosity) and unsteady flow computation

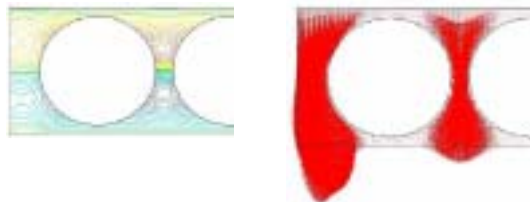
- Experimental supply (from [Idris & al. 2005]) Computational model (~80000 dofs)



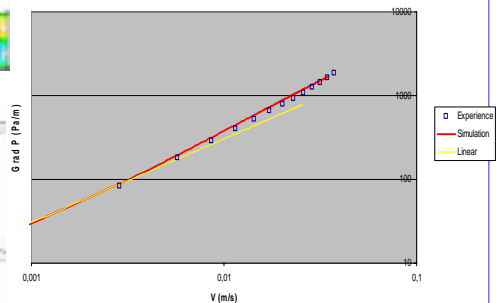
Pressure field



Pressure and velocities detail



Preliminary results:

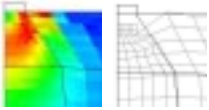
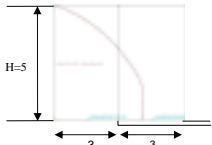
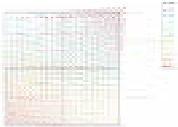



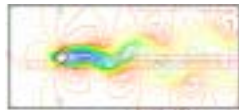


Conclusions 1/3

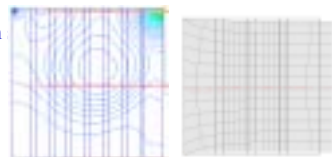
- High level programming concepts to integrate mathematical models and numerical algorithms
 - Advanced Object-oriented model based on a splitting operator solution scheme
 - Complexity handling, Consistency enforcement
 - Inner class concept
 - Enhanced data organization to enforce modeling
 - Interface concept
 - Algorithmic consistency enforcement: explicit algorithm equations for material
 - Application to Nonlinear material modeling: J2 plasticity
 - Numerical example: bearing capacity of a strip footing
 - Alternative example: flow through porous media

Conclusion 2/3

- Problems solved within the Java framework

 <p>Nonlinear elasticity (perfect plasticity, isotropic and kinematics hardening) [Eyheramendy 2004,2005]</p>	 <p>Darcy's flow (seepage problem with free surface tracking) [Eyheramendy & Guibert 2004]</p>	 <p>Darcy's flow (velocity/pressure, stabilized) [Guibert 2004]</p>	 <p>Heat conduction (steady, transient)</p>
 <p>Linear elasticity (static, dynamics)</p>	 <p>Periodic incompressible Newtonian and Non-Newtonian Navier-Stokes flow</p>		 <p>Incompressible Navier-Stokes (steady state, transient)</p>

- Parallel approaches in Java ([Eyheramendy 2003])
 - Multi-threaded Matrix/vector product (for any iterative linear system)
 - Schwarz Domain Decomposition method (Schwarz/Newton/Krylov)
 - Aitken acceleration techniques for DDM
 - Distributed computing (Grid computing)
 - Coupling with existing Fortran libraries (Blas, Lapack)

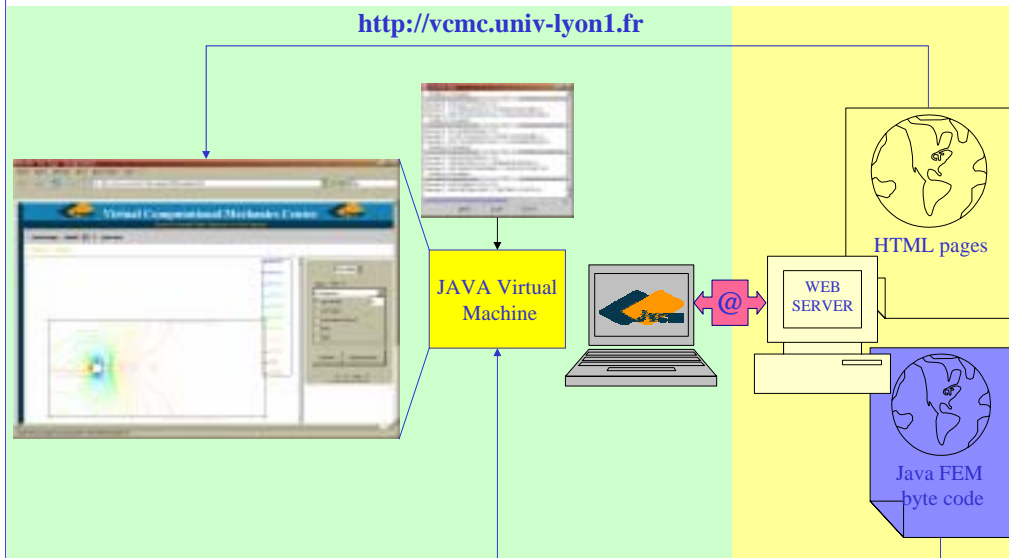


Conclusions 3/3

- Beyond the Java language: High level approaches to represent convenient frameworks for the developments in the future
 - Introduction of mathematical concepts into scientific computing tools
 - Advanced abstraction level in software tools: e.g. advanced object-oriented approach or functional programming
 - Including hardware aspects: Large scale portability, Simplicity, Code uniformity
- Convergence of different high level abstract approaches
 - Hybrid symbolic/numerical tools based on Java kind approaches ([Eyheramendy 2000])
 - Variational approach and automatic F.E coding
 - Ingredients of symbolic manipulations in classical numerical approach
 - Verification and validation of code
 - Integration of CAD kind approaches: new methods, CAD kinds systems...
- Need to get further in the portability of the application over the Internet
 - To enhance autonomy of codes over the network: components (independent objects)
 - Idea: independent components running in the same time
 - E.g.: Active objects (Actors): aims to enforce local consistency of objects

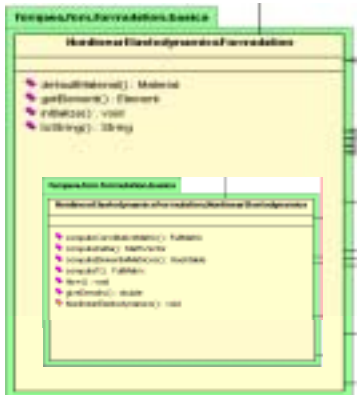
A first step towards Virtual Computational Mechanics

- Complete portability of the F.E application through the Internet



Data organization for Elastoplasticity

- Management of the F.E formulation:
 - Local and global level aspects managed at the same place
 - Problem fields definition:
 - displacements \mathbf{U} : vector defined at the nodes (unknown)
 - stress field $\boldsymbol{\sigma}$: tensor defined at Gauss Point
 - Finite element matrices computation and constitutive equations numerical integration



Consistency handling for the model