CentraleSupélec

EM2C

cnrs

université PARIS-SACLAY

Research field option

CASE STUDY 5
Comparative study of numerical tools to solve closure problems in porous media

Authors
Thomas GAUTHEY, Clément CÔTE, Emmanuel CASTIEL,
Yingjie ZHAO, Wassim BOURBIA

Supervisors
Morgan CHABANON, Bich-Lien DOAN, Cristina MANIU

Contents

1 Introduction and context

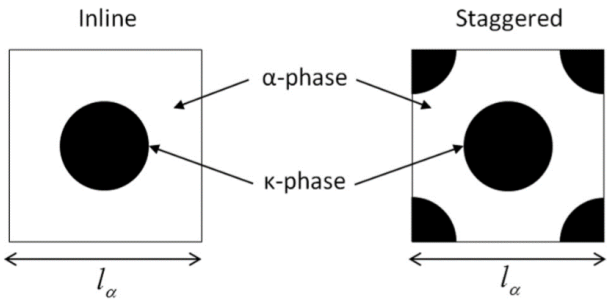
1.1 Transport homogenization in porous media

Porous media is a space occupied by multiphase materials.It must contain relatively small spaces, so-called pores or voids, free of solids, embedded in the solid or semisolid matrix, and it must be permeable to a variety of fluids, which means fluids should be able to penetrate through one face of a septum made of the material and emerge on the other side[1].

Porous materials are ubiquitous in engineering processes regarding energy and health, from petroleum industry to bioengineering. In such systems, studying the viscous flow mechanism is of great interest. However, the structure is too complex for direct analysis of transport equations within the pores. So the homogenization is performed to handle this difficulty.

The reason to use the homogenization is to rigorously derive continuum equations for multiphase systems[3]. And the key issue is to use local volume averaged equations which are valid everywhere to simplify the calculation. The chosen scale of the average volume must be representative enough to be applied to the whole media. To simplfy this process, we take the elementary volume as below, as well as performing the homogenization on it. After volume averaging, Darcy's law can be applied to such a scale[3].

Figure 1: Simplified elementary volume



The velocity field is determined with the continuity equation by incompressible, steady Stokes equations, and non-slip boundary[3].

$$\begin{cases} \nabla \cdot v_\alpha = 0 \\ 0 = -\nabla p_\alpha + \mu_\alpha \nabla^2 v_\alpha \\ v_\alpha = 0 \qquad\qquad \text{at} \quad \mathcal{A}_{\alpha\kappa} \end{cases}$$

The v_α, p_α and μ_α are the velocity, pressure and viscosity in α-phase. And $\mathcal{A}_{\alpha\kappa}$ is the interface between solid and fluid.

The homogenization is then performed as below[3],

$$\langle \psi \rangle = \frac{1}{V_\alpha + V_\kappa} \int_{V_\alpha + V_\kappa} \psi_\alpha dV$$

where ψ is the arbitrary physical quantity which can be v and p. V_α and V_κ are the volume of corresponding phase.

Then we can apply Darcy's law to this volume,

$$\langle v \rangle = -\frac{\underline{\underline{K}}}{\epsilon \mu_\alpha} \nabla \langle p \rangle$$

where $\langle v \rangle$ is the averaging velocity and $\langle p \rangle$ the averaging pressure. ϵ is the porosity defined as

$$\epsilon = \frac{V_\alpha}{V_\alpha + V_\kappa}$$

$\underline{\underline{K}}$ is the permeability tensor, the value of which is uniquely determined by the pore geometry and is independent of the properties of the penetrating fluid[1]. It's the function of closure variables, which lead to the closure problem.

1.2 Closure problem

The closure problem is characterized by equations below[3],

$$\begin{cases} 0 = -\nabla \underline{b} + \nabla^2 \underline{\underline{B}} + \underline{\underline{I}} \\ \nabla \cdot \underline{\underline{B}} = 0 \end{cases}$$

where $\underline{b}$ and $\underline{\underline{B}}$ are closure variables. The boundary conditions are defined as below.

$$\begin{cases} \underline{\underline{B}} = 0 \text{ at } \partial\Omega_\alpha \bigcap \partial\Omega_\kappa \\ \text{periodic at } \partial\Omega_\alpha \setminus \partial\Omega_\kappa \end{cases}$$

The permeability tensor can be then derived by the equation below,

$$\langle B \rangle^\alpha = \epsilon^{-1} \underline{\underline{K}}$$

where $\langle B \rangle^\alpha$ is the averaging closure variable.

1.3 Objectives of the study

- *Get started with different finite elements softwares :* The first objective is to get familiar with OpenFOAM, but we also purpose some other softwares to explore.
- *Solve the closure problem :*
- *Compare their performance :* We define metrics to evaluate these softwares, and

2 Softwares

2.1 OpenFOAM

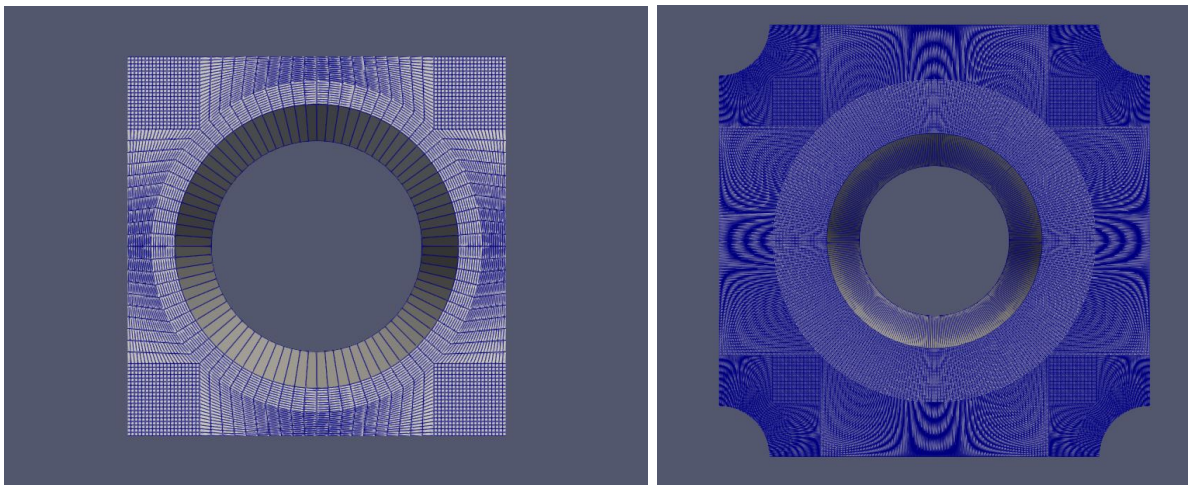
Presentation

OpenFOAM (Open-source Field Operation And Manipulation) is a Linux software encompassing several 3D numerical solvers for continuum mechanics, and mainly CFD (computational fluid dynamics). This solvers are coded in C++, but OpenFOAM comes with its own programming language, used for three tasks : creating the mesh, choosing the solver (and its options), and the post-processing functions.

Mesh

To create the mesh, the geometry must first be split into different blocks, each one with exactly 8 vertices. Then, each block is meshed, and the user can choose the form (hexahedron, prism, wedge...), the number and expansion ratio for each direction. The two geometries below have been created (only in 2D) :

Figure 2: Mesh created by openFoam and visualized by Paraview



Solvers

Five solvers are available to solve Navier-Stokes equations for incompressible flow : boundaryFoam, icoFoam, pimpleFoam, pisoFoam and simpleFoam. icoFoam doesn't allow for a source term in the equation, so it was discarded. Since simpleFoam is the only solver specialized for the steady-state equation, it was privileged. This solver is based upon the SIMPLE algorithm. This algorithm seeks to solve the equation :

$$\begin{cases} \nabla \cdot (\mathbf{u} \times \mathbf{u}) - \nabla \cdot \mathbf{R} = -\nabla p + \mathbf{S} \\ \nabla \cdot \mathbf{u} = 0 \end{cases} \tag{1}$$

where $\mathbf{u}$ is the velocity, p the pressure, $\mathbf{R}$ the stress tensor and $\mathbf{S}$ the source term.

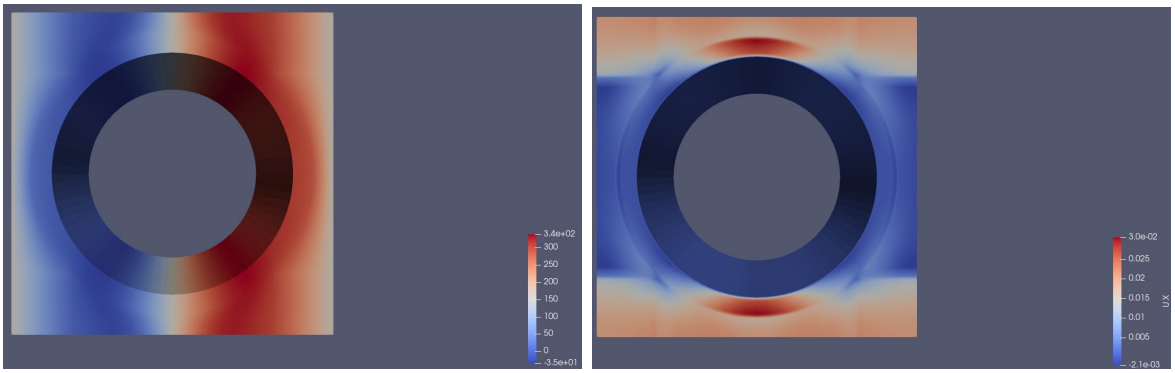
The algorithm works as follow :

1. Advance to the next time step t^{n+1}
2. Initialize $\mathbf{u}^{n+1}$ and p^{n+1} using latest available values
3. Discretize the momentum equation in the form $\mathrm{M}\mathbf{u} = -\nabla p$ (with M a matrix), and write $\mathrm{M}\mathbf{u} = \mathrm{A}\mathbf{u} - \mathbf{H}$ with A the matrix containing only the diagonal elements of M
4. Use the new expression $\mathbf{u} = A^{-1}\mathbf{H} - A^{-1}\nabla p$ in the continuity equation, which becomes $\nabla \cdot (A^{-1}\mathbf{H}) = \nabla \cdot (A^{-1}\nabla p)$
5. Solve the continuity equation for p, and update the value for $\mathbf{u}$ with the expression $\mathbf{u} = A^{-1}\mathbf{H} - A^{-1}\nabla p$
6. Go back to step 2 if the algorithm has not converged

The user can choose the value of the viscosity ν and the source term, the discretization schemes used by the solver to compute the derivatives, the tolerance value for the residuals below which the algorithm considers there is convergence and stops the loop, the boundary conditions (cyclic, Neumann-type, Dirichlet-type, etc.), the turbulence model (it was deactivated in our case)... Cyclic conditions were enforced on the external boundaries for both pressure and velocity, and the conditions $\mathbf{u} = \mathbf{0}$ and $\nabla p = \mathbf{0}$ were imposed at the interface between the two phases. Since p is defined up to an additive constant, the user must select a reference cell, that will be set to a reference value (also chosen by the user).

The results obtained with the source term (1 0 0) are shown below :

Figure 3: simpleFoam on a simple case : pressure and velocity field observed by Paraview



Pressure field : p Velocity field : $u.e_x$

2.2 FEniCS project

Presentation

FEniCS is a popular open-source computing platform for solving partial differential equations (PDEs). It enables users to quickly translate scientific models into efficient finite element code. With the high-level Python and C++ interfaces to FEniCS, it is easy to get started, but FEniCS offers also powerful capabilities for more experienced programmers. This software runs on a multitude of platforms ranging from laptops to high-performance clusters.

Mesh

Solvers

2.3 NGS-Py Finite Element Tool

Presentation

Netgen is an automatic 2D and 3D tetrahedral mesh generator. NGSolve is a finite element library which can be linked to Netgen and contains arbitrary order finite elements of all standard element geometries, scalar, vector-valued, hybrid DG finite element spaces. Program flow as well as geometry description and equation setup can be controlled from Python. Together they offer a rich Python interface inspired by the FEniCS project.

Mesh

Netgen allows for precise control of the meshing process while still providing automatic options. The geometry can be constructed either using a point-edge-shape system alike GMSH either using shapes primitives and Boolean operators (only in available in 3D).
How to create a mesh in 2D using Netgen :

- *Define the geometry object by :* geo = SplineGeometry()
- *Define the vertex of the geometry :* geo.AppendPoint(x,y) → PointObject
- *Define the edges of the geometry :*
  geo.Append(["line",PointObjA,PointObjB],leftdomain=(int),rightdomain=(int),bc="name"
)
  or for curved edges
  geo.Append(["spline3",PointObjA,PointObjB,PointObjC],leftdomain=(int),rightdomain=(int),bc="
)
  Some primitive also exist in 2D but don't allow Boolean operations

Solvers

3 Methods

3.1 Protocol

Quel est l'expérience commune aux trois softwares ?

3.2 Metrics

Sur quoi nous allons les comparer ?

4 Results

4.1 Mésocentre

Pourquoi on utilise mésocentre ?
Capacité de la ruche ? (puce, cb de coeurs,...)

4.2 Comparison

perméabilité (K_{xx},..) Convergence en temps
Convergence en mailles
Ergonomie,Flexibilité

5 Conclusion

5.1 Which one should you choose?

Dans quelles mesures prendre l'un ou l'autre

References

[1] Front matter. In F.A.L. DULLIEN, editor, *Porous Media (Second Edition)*, page iii. Academic Press, San Diego, second edition edition, 1992.

[2] V. V. Tuchin. Polarized light interaction with tissues. *Journal of biomedical optics, vol. 21, no. 7*, 2016.

[3] Stephen Whitaker. *The Method of Volume Averaging*. 01 1999.