The Multiscale Mixed Finite-Element Method

Knut–Andreas Lie[1,2,3]

[1]SINTEF, Department of Applied Mathematics, Norway

[2]University of Bergen, Department of Mathematics

[3]Center of Mathematics for Applications, University of Oslo

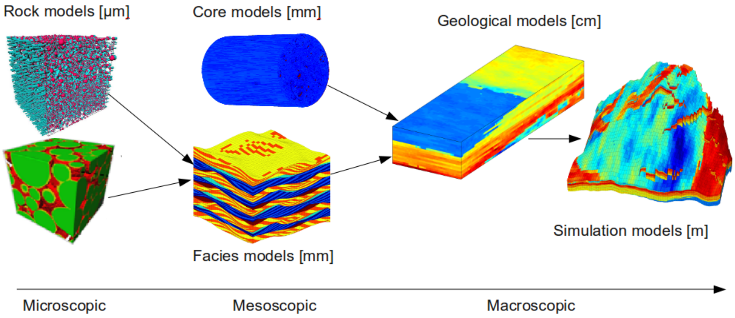
eVITA Winter School, Dr. Holms, Geilo, Jan 23–28, 2011

Physical scales in porous media flow

... one cannot resolve them all at once

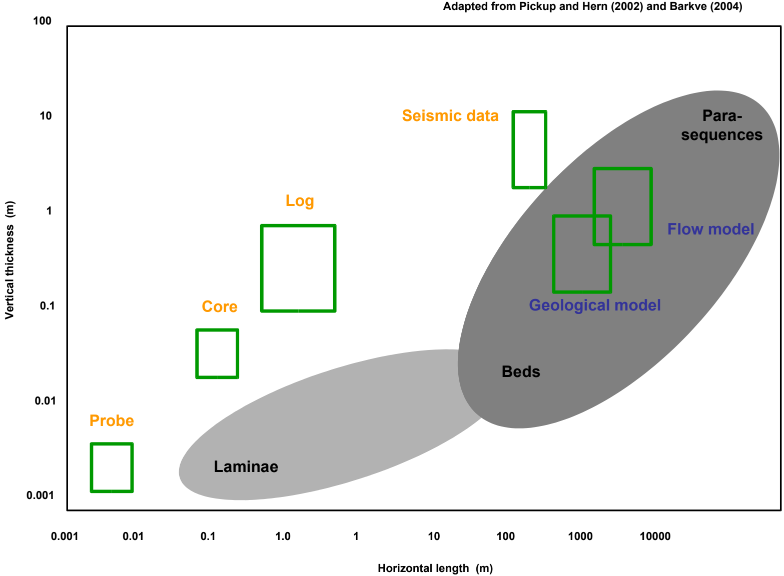
The scales that impact fluid flow in oil reservoirs range from

- the micrometer scale of pores and pore channels
- via dm–m scale of well bores and laminae sediments
- to sedimentary structures that stretch across entire reservoirs.



Physical scales in porous media flow

... and even measuring them is hard



Geological models

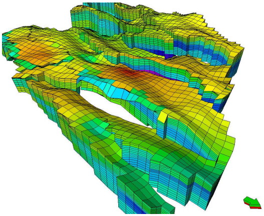
Articulation of the geologists' perception of the reservoir

Geological models:

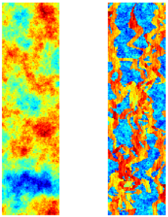
- here: geo-cellular models
- describe the reservoir geometry (horizons, faults, etc)
- typically generated using geostatistics
- give rock parameters (permeability and porosity)

Rock parameters:

- have a multiscale structure
- details on all scales impact flow
- permeability spans many orders of magnitude



Ex: Brent sequence



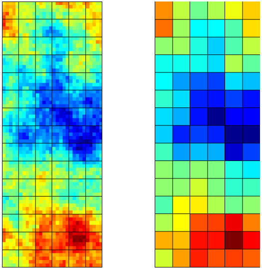
Building a coarse-scale model

Upscaling: geological $\longrightarrow$ simulation model

Gap in resolution:

- Geomodels: $10^7 - 10^9$ cells
- Simulators: $10^5 - 10^6$ cells

$\longrightarrow$ upscaling of parameters



Many alternatives:

- Harmonic, arithmetic, geometric, ...
- Local (K or T) methods
- Global methods
- Local-global methods
- Wavelet, multi-resolution, renormalization, ...
- Ensemble methods

Multiphase flow:

- Pseudo methods
- Steady-state methods

Fractional formulation (no gravity or capillary forces):

$$-\nabla\big(\mathbf{K}\lambda(S)\nabla p\big) = q, \qquad v = -\mathbf{K}\lambda(S)\nabla p,$$
$$\phi\partial_t S + \nabla\cdot(vf(S)) = 0$$

Numerical solution by operator splitting (each equation by a specialised numerical method):

pressure: multiscale or upscaling-downscaling method

saturation: finite volumes or streamlines

Iterated implicit (+ domain decomposition) converges within a few iterations and is therefore an alternative to fully implicit

Purpose:

Derive effective petrophysical parameters that produces the same flow response on a coarser model.

Elliptic pressure equation

$$-\nabla \cdot \mathbf{K}\nabla p = f, \qquad \text{in } \Omega$$

For each coarse grid block B, we seek a tensor $\mathbf{K}^*$ such that

$$\int_B \mathbf{K}\nabla p \, dx = \mathbf{K}^* \int_B \nabla p \, dx,$$

i.e., the net flow rate $\bar{v}$ through B is related to the average pressure gradient $\overline{\nabla p}$ in B through Darcy's law $\bar{v} = -\mathbf{K}^*\overline{\nabla p}$.

One-dimensional pressure equation:

$$-\big(K(x)p'(x)\big)' = 0, \qquad p(a) = p_0,\ p(b) = p_1$$

Integration gives that the velocity $v = -K(x)p'(x)$ is constant. Hence

$$K^* \int_a^b p'(x)\,dx = \int_a^b K(x)p'(x)\,dx$$

$$K^* \int_a^b \frac{v}{K(x)}\,dx = \int_a^b v\,dx$$

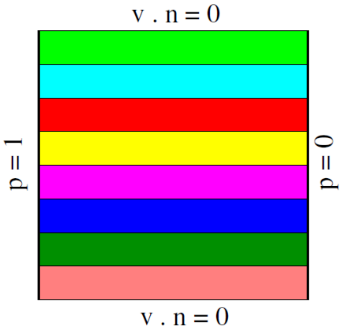
$$\implies \quad K^* = (b-a)\Big[\int_a^b \frac{1}{K(x)}\,dx\Big]^{-1}$$

In other words, K^* is identical to the harmonic average.

Upscaling

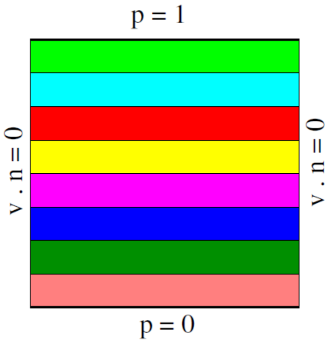
Two multi-dimensional cases

Flow from left to right



$\mathbf{K}^* =$ arithmetic average

Flow from top to bottom



$\mathbf{K}^* =$ harmonic average

Conclusion: correct upscaling depends on the flow

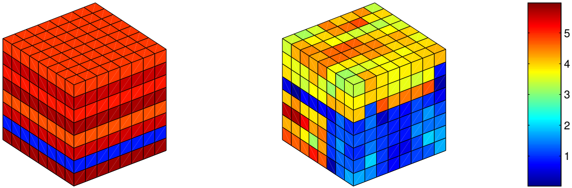
Harmonic-arithmetic averaging

To model flow in more than one direction, define a diagonal permeability tensor with the following diagonal components:

$$k_{xx} = \mathcal{A}_a^z \mathcal{A}_a^y \mathcal{A}_h^x \mathbf{K}, \quad k_{yy} = \mathcal{A}_a^z \mathcal{A}_a^x \mathcal{A}_h^y \mathbf{K}, \quad k_{zz} = \mathcal{A}_a^x \mathcal{A}_a^y \mathcal{A}_h^z \mathbf{K}.$$

Here, $\mathcal{A}_a^\xi$ and $\mathcal{A}_h^\xi$ represent the arithmetic and harmonic mean operators in the ξ-coordinate direction.

Harmonic-arithmetic averaging gives correct upscaling for perfectly stratified media with flow parallel to, or perpendicular to the layers



BC1: $p = 1$ at $(x, y, 0)$, $p = 0$ at $(x, y, 1)$, no-flow elsewhere.

BC2: $p = 1$ at $(0, 0, z)$, $p = 0$ at $(1, 1, z)$, no-flow elsewhere.

BC3: $p = 1$ at $(0, 0, 0)$, $p = 0$ at $(1, 1, 1)$, no-flow elsewhere.

	Model 1			Model 2		
	BC1	BC2	BC3	BC1	BC2	BC3
Q_H/Q_R	1	2.31e−04	5.52e−02	1.10e−02	3.82e−06	9.94e−04
Q_A/Q_R	4.33e+03	1	2.39e+02	2.33e+04	8.22	2.13e+03
Q_{HA}/Q_R	1	1	1.14	8.14e−02	1.00	1.55e−01

For each grid block B, solve the homogeneous equation

$$-\nabla \cdot \mathbf{K} \nabla p = 0 \quad \text{in } B,$$

with three sets of boundary conditions, one for each coordinate direction. Compute an upscaled tensor $\mathbf{K}^*$ with components

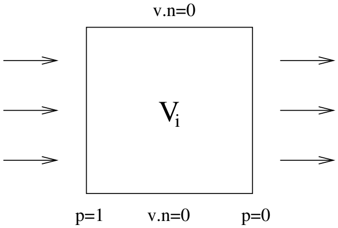
$$k_{x\xi} = -Q_\xi L_\xi / \Delta P_x, \quad k_{y\xi} = -Q_\xi L_\xi / \Delta P_y, \quad k_{z\xi} = -Q_\xi L_\xi / \Delta P_z.$$

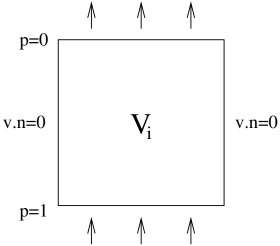
Here, Q_ξ, L_ξ and ΔP_ξ are the net flow, the length between opposite sides, and the pressure drop in the ξ-direction inside B.

Fundamental problem:

What kind of boundary conditions should be imposed?

Fixed boundary conditions $\longrightarrow$ diagonal tensor





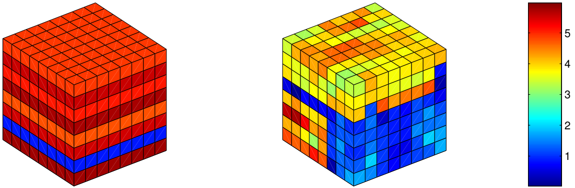
Periodic boundary conditions (in x-direction)

$$\begin{aligned} p(1,y) &= p(0,y) - \Delta p, \qquad & p(x,1) &= p(x,0), \\ v(1,y) &= v(0,y), \qquad & v(x,1) &= v(x,0) \end{aligned}$$

yeld a symmetric and positive-definite tensor $\mathbf{K}^*$

Flow-based upscaling

Example



BC1: $p = 1$ at $(x, y, 0)$, $p = 0$ at $(x, y, 1)$, no-flow elsewhere.

BC2: $p = 1$ at $(0, 0, z)$, $p = 0$ at $(1, 1, z)$, no-flow elsewhere.

BC3: $p = 1$ at $(0, 0, 0)$, $p = 0$ at $(1, 1, 1)$, no-flow elsewhere.

	Model 1			Model 2		
	BC1	BC2	BC3	BC1	BC2	BC3
Q_{HA}/Q_R	1	1	1.143	0.081	1.003	0.155
Q_D/Q_R	1	1	1.143	1	1.375	1.893
Q_P/Q_R	1	1	1.143	0.986	1.321	1.867

Flow-based upscaling

More advanced techniques

- Transmissibility: use two blocks to derive effective T^*

$$v_{ij} = T_{ij}(p_i - p_j)$$

- Extended local: use larger domain to reduce influence of b.c.
- Global: use global flow solution to set b.c.
- Local-global: bootstrapping procedure

Flow-based upscaling

More advanced techniques

- Transmissibility: use two blocks to derive effective T^*

$$v_{ij} = T_{ij}(p_i - p_j)$$

- Extended local: use larger domain to reduce influence of b.c.
- Global: use global flow solution to set b.c.
- Local-global: bootstrapping procedure

However,

- upscaling is a bottleneck in workflow,
- gives loss of information/accuracy,
- is not sufficiently robust (dependent on flow regime),
- is not consistent with governing PDE(s),
- extensions to multiphase flow are somewhat shaky

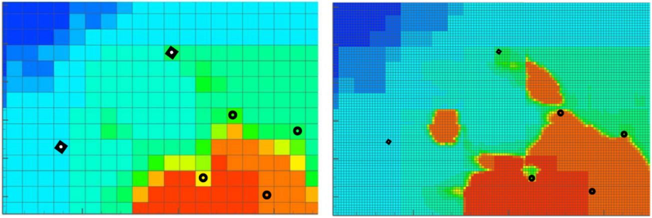
Simulation on seismic/geologic grid

Why do we want/need it?

Simulation on seismic/geologic grid:

- best possible resolution of the physical processes
- faster model building and history matching
- makes inversion a better instrument to find remaining oil
- better estimation of uncertainty by running alternative models

Example: Giant Middle-East field (10 million vs 1 billion cells)



From Dogru et al., SPE 119272

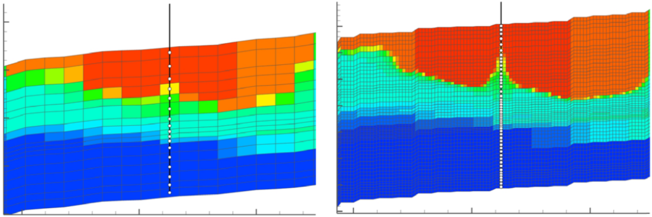
Simulation on seismic/geologic grid

Why do we want/need it?

Simulation on seismic/geologic grid:

- best possible resolution of the physical processes
- faster model building and history matching
- makes inversion a better instrument to find remaining oil
- better estimation of uncertainty by running alternative models

Example: Giant Middle-East field (10 million vs 1 billion cells)



From Dogru et al., SPE 119272

Million-cell models on desktop computers

How to get there..?

Simplified flow physics

Can often tell a lot about the fluid movement. "Full physics" is typically only required towards the end of a workflow

Operator splitting

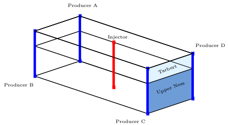
Fully coupled solution is slow.. Subequations often have different time scales. Splitting opens up for tailor-made methods

Simplified flow physics

Can often tell a lot about the fluid movement. “Full physics” is typically only required towards the end of a workflow

Operator splitting

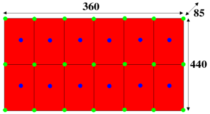
Fully coupled solution is slow.. Subequations often have different time scales. Splitting opens up for tailor-made methods



SINTEF inhouse code:

- $60 \times 220 \times 85 = 1.1$ million cells, 25 time steps
- Intel 2.4 GHz with 2 GB RAM:

multigrid:	8 min 36 sec
multiscale:	2 min 22 sec



FrontSim:

- $360 \times 440 \times 85 = 13.5$ million cells
- Intel Xeon 5482, 64 Gb, 3.2 GHz, single thread
- Computing time: 1 h 55 min

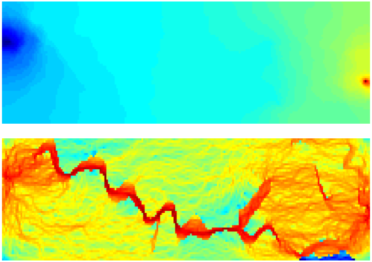
Million-cell models on desktop computers

How to get there..?

Use of sparsity / (multiscale) structure

- effects resolved on different scales
- small changes from one step to next
- small changes from one simulation to next

Example: SPE10, Layer 36



Multiscale idea:

- Pressure on coarse grid
- Velocity on fine grid

Incorporate impact of subgrid heterogeneity in approximation spaces

Advantages: utilize more geological data, more accurate solutions, geometrical flexibility

Million-cell models on desktop computers

Prerequisites for real-field studies

More efficient than standard solvers:

- easy to parallelise,
- less memory requirements than fine-grid solvers.

Ability to handle industry-standard grids:

- (highly) skewed and degenerate grid cells,
- non-matching cells,
- unstructured connectivities.

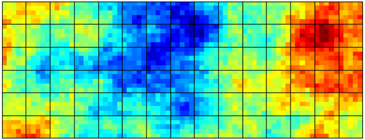
Compatible with current solvers:

- can be built on top of commercial/inhouse solvers,
- must be able to use existing linear solvers.

From flow-based upscaling to multiscale methods

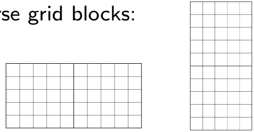
Utilizing the same computations more efficiently

Standard upscaling:



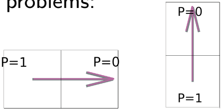
$\Downarrow$

Coarse grid blocks:

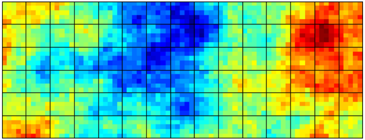


$\Downarrow$

Flow problems:

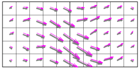


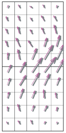
Standard upscaling:



⇓

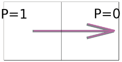
Coarse grid blocks:

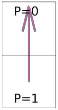




⇓ ⇑

Flow problems:

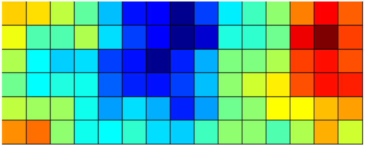




From flow-based upscaling to multiscale methods

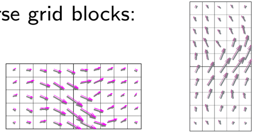
Utilizing the same computations more efficiently

Standard upscaling:



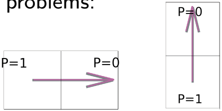
⇓ ⇑

Coarse grid blocks:



⇓ ⇑

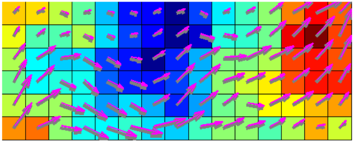
Flow problems:



From flow-based upscaling to multiscale methods

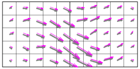
Utilizing the same computations more efficiently

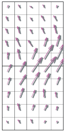
Standard upscaling:



⇓ ⇑

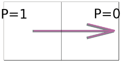
Coarse grid blocks:

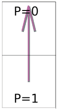




⇓ ⇑

Flow problems:

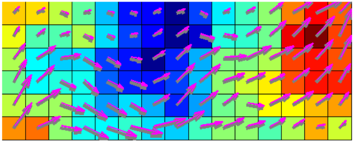




From flow-based upscaling to multiscale methods

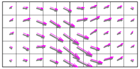
Utilizing the same computations more efficiently

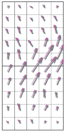
Standard upscaling:



⇓ ⇑

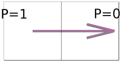
Coarse grid blocks:

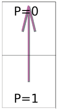




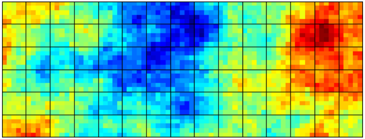
⇓ ⇑

Flow problems:



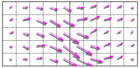


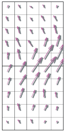
Multiscale method:



⇓

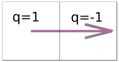
Coarse grid blocks:

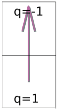




⇓ ⇑

Flow problems:

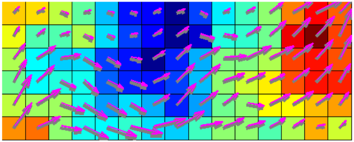




From flow-based upscaling to multiscale methods

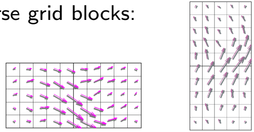
Utilizing the same computations more efficiently

Standard upscaling:



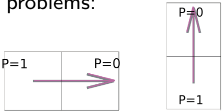
⇓ ⇑

Coarse grid blocks:

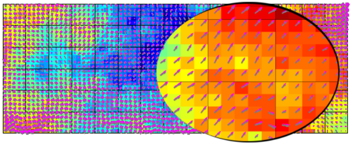


⇓ ⇑

Flow problems:

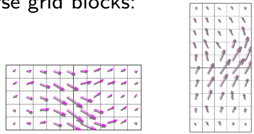


Multiscale method:



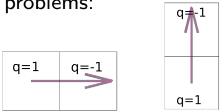
⇓ ⇑

Coarse grid blocks:



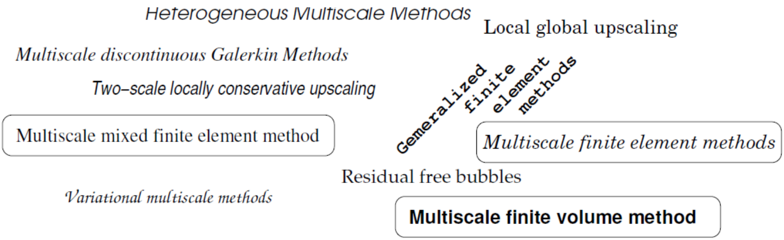
⇓ ⇑

Flow problems:



Multiscale methods

Numerical methods that attempt to model physical phenomena on coarse grids while honoring small-scale features that impact the coarse grid solution in an appropriate way



The multiscale finite-element (MsFE) method

In one spatial dimension

Model problem

Consider the Poisson-type problem

$$\partial_x(K(x)\partial_x p) = f, \qquad x \in \Omega = [0,1], \quad p(0) = p(1) = 0,$$

where $f, k \in L^2(\Omega)$ and $0 < \alpha < K(x) < \beta$ for all $x \in \Omega$

Variational formulation

Find $p \in H_0^1(\Omega)$ such that

$$a(p,v) = (f,v) \qquad \text{for all } v \in H_0^1(\Omega,$$

where $(\cdot,\cdot)$ is the L^2 inner-product and

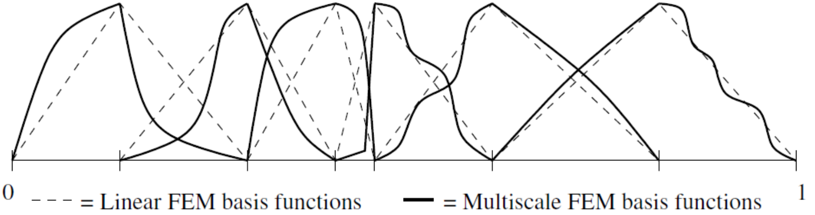
$$a(p,v) = \int_\Omega K(x)\partial_x p \partial_x v \, dx$$

Let $\mathcal{N}_B = \{0 = x_0 < x_1 < \cdots < x_n = 1\}$ be a set of nodal points and define $B = (x_{i-1}, x_i)$. For $i = 1, \ldots, n-1$, we define a basis function $\phi^i \in H_0^1(\Omega)$ by

$$a(\phi^i, v) = 0 \qquad \text{for all } v \in H_0^1(B_i \cup B_{i+1}), \qquad \phi^i(x_j) = \delta_{ij},$$

where δ_{ij} is the Kronecker delta.

Basis functions



The MsFE method

Find the unique function p_0 in

$$\begin{aligned} V^{\mathsf{ms}} &= \operatorname{span}\{\phi^i\} \\ &= \{u \in H_0^1(\Omega) : a(u,v) = 0 \text{ for all } v \in H_0^1(\cup_i B_i)\} \end{aligned}$$

satisfying

$$a(p_0, v) = (f, v) \quad \text{for all } v \in V^{\mathsf{ms}}$$

Theorem

Assume that p solves the variational formulation. Then $p = p_0 + \sum_{i=1}^n p_i$, where $p_i \in H_0^1(B_i)$ is defined by

$$a(p_i, v) = (f, v) \quad \text{for all } v \in H_0^1(B_i)$$

Assume that p solves the variational formulation and that $v \in V^{\text{ms}}$. Then

$$\begin{aligned} a(p - p_0, v) &= a(p, v) - a(p_0, v) \\ &\quad (f, v) - (f, v) = 0 \end{aligned}$$

Hence, p_0 is the orthogonal projection of p onto V^{ms}

Since $H_0^1(\Omega) = V^{\text{ms}} \otimes H_0^1(\cup_i B_i)$ it follows that

$$p_0(x_i) = p(x_i) \quad \text{for all } i$$

In other words, p_0 is the interpolant of p in V^{ms}

Let p_I be the interpolant of p in V^{ms}. Then $p - p_I \in H_0^1(\cup_i B_i)$ and it follows from the mutual orthogonality of V^{ms} and $H_0^1(\cup_i B_i)$ with respect to $a(\cdot,\cdot)$ that

$$a(p - p_I, v) = 0 \quad \text{for all } v \in V^{\mathsf{ms}}$$

Hence, for all $v \in V^{\mathsf{ms}}$

$$a(p_I, v) = a(p, v) = (f, v) = a(p_0, v) \quad \Longrightarrow a(p_I - p_0, v) = 0$$

Thus, in particular, by choosing $v = p_I - p_0$ we obtain

$$a(p_I - p_0, p_I - p_0) = 0,$$

which implies that $p_0 = p_I$

Super-convergence property

Solution of the variational problem is decomposed into the MsFE solution and solutions of independent local subgrid problems.

MsFEM in 1D = a Schur complement decomposition

Does the result extend to higher dimensions?

Super-convergence property

Solution of the variational problem is decomposed into the MsFE solution and solutions of independent local subgrid problems.

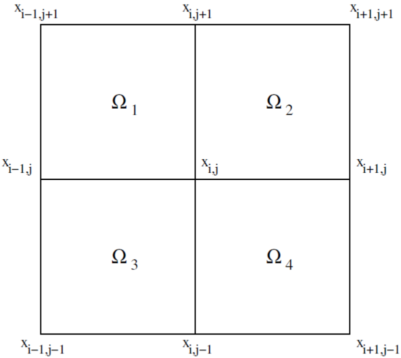
MsFEM in 1D = a Schur complement decomposition

Does the result extend to higher dimensions?

No, but the basic construction applies and helps us understand how subgrid features of the solution can be embodied into a coarse grid approximation space.

The MsFE method

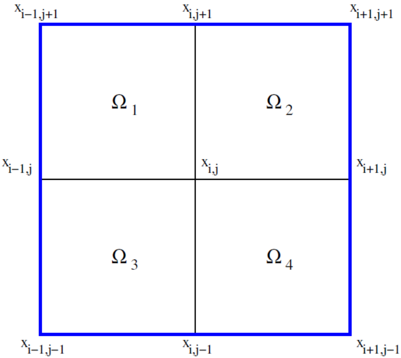
Basis functions in 2D



- $p \in V^{\mathsf{ms}}$ implies that $\nabla \cdot \mathbf{K}\nabla\phi^{ij} = 0$ in all Ω_m

The MsFE method

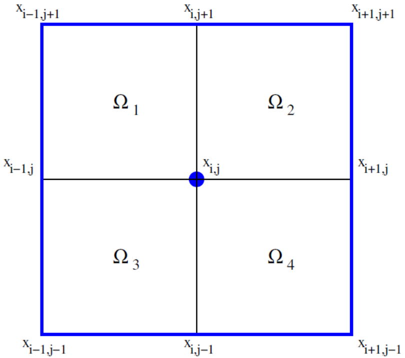
Basis functions in 2D



- $p \in V^{\text{ms}}$ implies that $\nabla \cdot \mathbf{K}\nabla\phi^{ij} = 0$ in all Ω_m
- $\phi^{ij} = 0$ on edges not emanating from $x_{i,j}$

The MsFE method

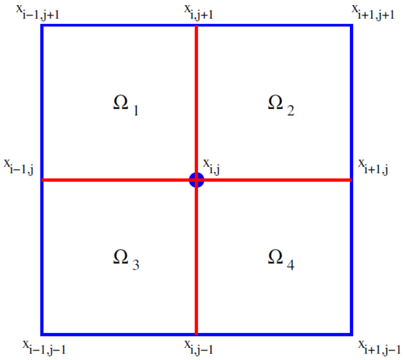
Basis functions in 2D



- $p \in V^{\text{ms}}$ implies that $\nabla \cdot \mathbf{K} \nabla \phi^{ij} = 0$ in all Ω_m
- $\phi^{ij} = 0$ on edges not emanating from $x_{i,j}$
- $\phi^{ij}(x_{m,n}) = \delta_{i,m}\delta_{j,n}$

The MsFE method

Basis functions in 2D



- $p \in V^{\text{ms}}$ implies that $\nabla \cdot \mathbf{K}\nabla\phi^{ij} = 0$ in all Ω_m
- $\phi^{ij} = 0$ on edges not emanating from $x_{i,j}$
- $\phi^{ij}(x_{m,n}) = \delta_{i,m}\delta_{j,n}$
- Boundary conditions on edges emanating from $x_{i,j}$?

Unfortunately, the MsFE method is not locally mass-conservative in higher dimensions

The multiscale mixed finite-element (MsMFE) method

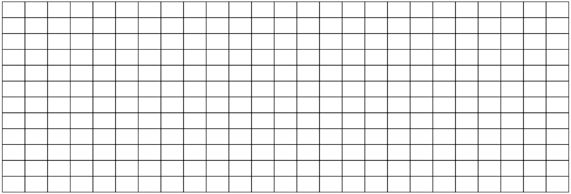
Mixed formulation for incompressible flow

Find $(v, p) \in H_0^{1,\text{div}} \times L^2$ such that

$$\begin{aligned} \int (\lambda K)^{-1} u \cdot v \, dx - \int p \nabla \cdot u \, dx = 0, & \qquad \forall u \in H_0^{1,\text{div}}, \\ \int \ell \nabla \cdot v \, dx = \int q \ell \, dx, & \quad \forall \ell \in L^2. \end{aligned}$$

Standard MFE method

- Seek solution in $\mathbf{V}_h \times W_h \subset H_0^{1,\text{div}} \times L^2$
- Approximation spaces: piecewise polynomials



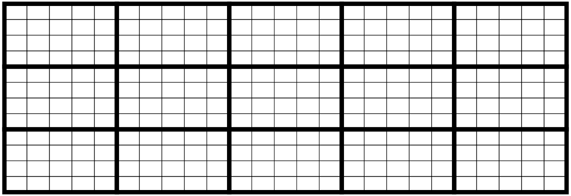
Find $(v,p) \in H_0^{1,\text{div}} \times L^2$ such that

$$\int (\lambda K)^{-1} u \cdot v \, dx - \int p \nabla \cdot u \, dx = 0, \qquad \forall u \in H_0^{1,\text{div}},$$

$$\int \ell \nabla \cdot v \, dx = \int q \ell \, dx, \quad \forall \ell \in L^2.$$

Multiscale MFE method

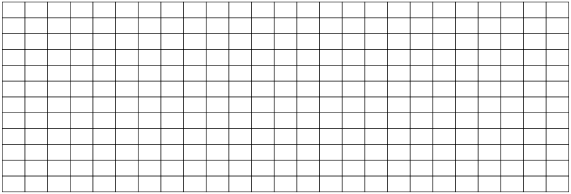
- Seek solution in $\mathbf{V}_{H,h} \times W_{H,h} \subset H_0^{1,\text{div}} \times L^2$
- Approximation spaces: local numerical solutions



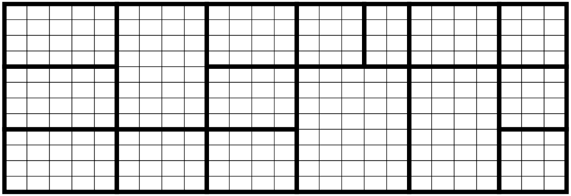
The MsMFE method

Grids and basis functions in general

Fine grid with petrophysical parameters cell

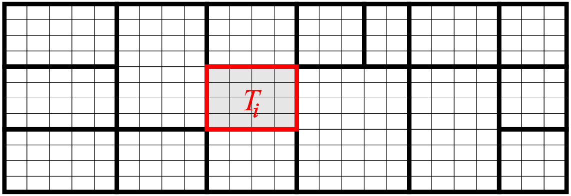


Fine grid with petrophysical parameters cell



Construct a *coarse* grid, and choose the discretisation spaces V and U^{ms} such that:

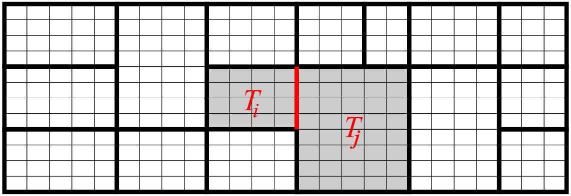
Fine grid with petrophysical parameters cell



Construct a *coarse* grid, and choose the discretisation spaces V and U^{ms} such that:

- For each coarse block T_i, there is a basis function $\phi_i \in V$.

Fine grid with petrophysical parameters cell



Construct a *coarse* grid, and choose the discretisation spaces V and U^{ms} such that:

- For each coarse block T_i, there is a basis function $\phi_i \in V$.
- For each coarse edge Γ_{ij}, there is a basis function $\psi_{ij} \in U^{ms}$.

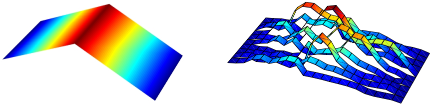
Decomposition:

- $p(x,y) = \sum_i p_i \phi_i(x,y)$ – sum over all coarse blocks
- $v(x,y) = \sum_{ij} v_{ij} \psi_{ij}(x,y)$ – sum over all block faces

Basis ϕ_i for pressure:

$$\phi_i = \begin{cases} 1 & \text{in } T_i, \\ 0 & \text{otherwise.} \end{cases}$$

Basis ψ_{ij} for velocity:



homogeneous (RT0) heterogeneous

The MsMFE method

Local flow problems

Velocity basis function ψ_{ij} solves a local system of equations in Ω_{ij}:

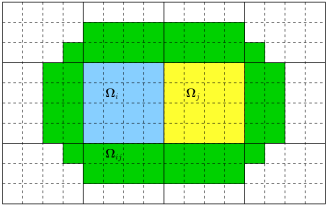
$$\vec{\psi}_{ij} = -\mu^{-1}\mathbf{K}\nabla\varphi_{ij}$$

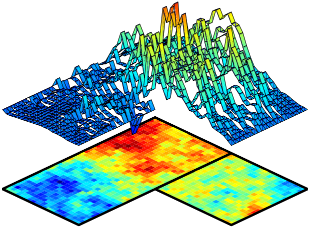
$$\nabla \cdot \vec{\psi}_{ij} = \begin{cases} w_i(\vec{x}), & \text{if } \vec{x} \in \Omega_i, \\ -w_j(\vec{x}), & \text{if } \vec{x} \in \Omega_j, \\ 0, & \text{otherwise.} \end{cases}$$

with no-flow conditions on $\partial\Omega_{ij}$

Source term: $w_i \propto \text{trace}\,(K_i)$ drives a unit flow through Γ_{ij}.

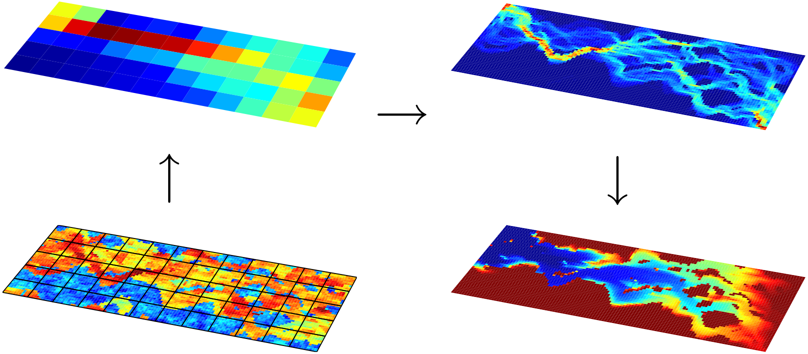
If there is a sink/source in T_i, then $w_i \propto q_i$.





Compute coarse-scale velocity

Reconstruct fine-scale velocity



Geomodel with petrophysical parameters from fine scale

Evolve fine-scale saturations using the fine-scale fluxes

Mixed form:

$$\begin{bmatrix} B & C \\ C^T & 0 \end{bmatrix} \begin{bmatrix} v \\ p \end{bmatrix} = \begin{bmatrix} 0 \\ g \end{bmatrix},$$

$$b_{ij} = \int_\Omega \psi_i \big(\lambda K\big)^{-1} \psi_j \, dx,$$

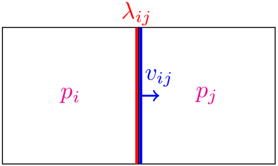
$$c_{ik} = \int_\Omega \phi_k \nabla \cdot \psi_i \, dx$$

Indefinite, saddle-point problem. Requires special numerical linear algebra

The MsMFE method

Linear system: mixed hybrid form

$$\begin{bmatrix} B & C & D \\ C^T & 0 & 0 \\ D^T & 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ -p \\ \pi \end{bmatrix} = \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix},$$



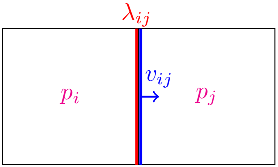
Here,

$$b_{ij} = \int_\Omega \psi_i \left(\lambda K\right)^{-1} \psi_j \, dx, \quad c_{ik} = \int_\Omega \phi_k \nabla \cdot \psi_i \, dx, \quad d_{ik} = \int_{\partial\Omega} |\psi_i \cdot n_k| \, dx$$

The MsMFE method

Linear system: mixed hybrid form

$$\begin{bmatrix} B & C & D \\ C^T & 0 & 0 \\ D^T & 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ -p \\ \pi \end{bmatrix} = \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix},$$



Here,

$$b_{ij} = \int_\Omega \psi_i \big(\lambda K\big)^{-1} \psi_j \, dx, \quad c_{ik} = \int_\Omega \phi_k \nabla \cdot \psi_i \, dx, \quad d_{ik} = \int_{\partial\Omega} |\psi_i \cdot n_k| \, dx$$

Reduced to a positive-definite form based using a Schur-complement

$$\big(D^T B^{-1} D - F^T L^{-1} F\big)\pi = F^T L^{-1} g,$$
$$F = C^T B^{-1} D, \quad L = C^T B^{-1} C.$$

Reconstruct cell pressures and fluxes by back-substition,

$$Lp = q + F^T \pi, \qquad Bv = Cp - D\pi.$$

The MsMFE method

Algebraic formulation

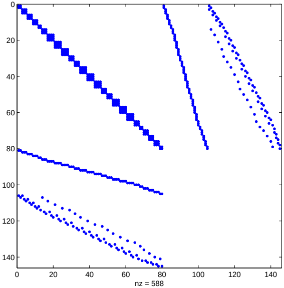
Split the basis functions, $\boldsymbol{\psi}_{ij} = \boldsymbol{\psi}_{ij}^H - \boldsymbol{\psi}_{ji}^H$

$$\boldsymbol{\psi}_{ij}^H(E) = \begin{cases} \boldsymbol{\psi}_{ij}(E), & \text{if } E \in T_{ij} \setminus T_j \\ 0, & \text{otherwise} \end{cases} \qquad \boldsymbol{\psi}_{ji}^H(E) = \begin{cases} -\boldsymbol{\psi}_{ij}(E), & \text{if } E \in T_j \\ 0, & \text{otherwise} \end{cases}$$

Hybrid basis functions ψ_{ij}^H as columns in a matrix $\boldsymbol{\Psi}$

Coarse-scale hybrid mixed system

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{B}\boldsymbol{\Psi} & \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{C}\boldsymbol{\mathcal{I}} & \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{D}\boldsymbol{\mathcal{J}} \\ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{C}^\mathsf{T}\boldsymbol{\Psi} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{\mathcal{J}}^\mathsf{T}\boldsymbol{D}^\mathsf{T}\boldsymbol{\Psi} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^c \\ -\boldsymbol{p}^c \\ \boldsymbol{\lambda}^c \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{g}^c \\ \boldsymbol{0} \end{bmatrix}$$

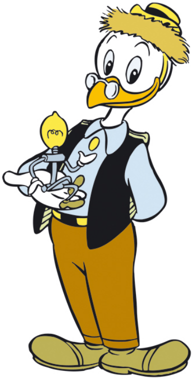


$\boldsymbol{\Psi}$ – matrix with basis functions
$\boldsymbol{\mathcal{I}}$ – prolongation from blocks to cells
$\boldsymbol{\mathcal{J}}$ – prolongation from block faces to cell faces

Reconstruction of fine-scale velocity $\boldsymbol{v}^f = \boldsymbol{\Psi}\boldsymbol{v}^c$
(Pressure bases may also have fine-scale structure if necessary)

The MsMFE method

This sounds interesting — where do I get it?



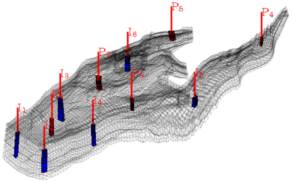
MRST Version 2010a

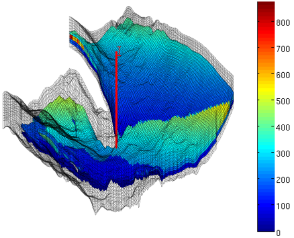
- routines and data structures for reading, representing, processing and visualizing unstructured grids
- corner-point grids / Eclipse input
- standard flow and transport solvers for one and two phases
- multiscale flow solvers

Inhouse version:

- black-oil models
- adjoint methods, reordering, flow-based grids, etc.

`http://www.sintef.no/MRST`





Multiscale versus upscaling methods

Comparison of accuracy and efficiency

Upscaling methods

- Harmonic-arithmetic averaging
- Flow-based upscaling with unit pressure drop
- Adaptive local-global upscaling (Chen & Durlofsky)

Fine-grid solution: downscaling using nested gridding

Multiscale methods

- The multiscale finite-volume (MsFV) method
- Numerical subgrid-upscaling (NSU) method (Arbogast et al.)
- The mixed finite-element (MsMFE) method

From: V. Kippe, J. E. Aarnes, and K.-A. Lie. A comparison of multiscale methods for elliptic problems in porous media flow. Comput. Geosci., Special issue on multiscale methods. Vol. 12, No. 3, pp. 377-398, 2008. DOI: 10.1007/s10596-007-9074-6

Multiscale versus upscaling methods

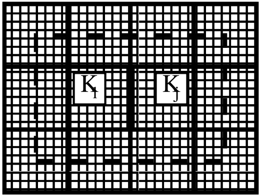
Adaptive local-global upscaling / nested gridding

Global upscaling + fine-grid reconstruction = a 'multiscale' method:

- Compute initial T^*_{lj}'s using standard upscaling
- Solve global coarse-scale pressure equation with T^*_{lj}'s
- Until convergence (in v and p)
  - Interpolate between pressures to get BC for local flow problems
  - Compute new T^*_{lj}'s from local flow problems
  - Solve global coarse-scale pressure equation with new T^*_{lj}'s
- Solve coarse-scale problem (wells and BC) with upscaled T^*_{lj}'s
- Reconstruct fine-scale velocity field with nested gridding

Upscale transmissibility:

$$-\nabla \cdot K \nabla p = 0 \quad \text{in} \quad \Omega_{lj}$$
$$p = I p^* \quad \text{in} \quad \partial\Omega_{lj}$$



$$T^*_{lj} = \frac{\int_{\partial K_l \cap \partial K_j} v \cdot n_{lj} \, ds}{\int_{K_l} p \, dx - \int_{K_j} p \, dx}$$

Solve coarse-scale problem:

$$\sum_j T^*_{lj}(p_l - p_j) = \int_{K_l} q \, dx \quad \forall K_l$$

Construct fine-scale velocity:

$$v = -K \nabla p, \quad \nabla \cdot v = q \quad \text{in } K_l$$
$$v \cdot n = \frac{T_{ki}(v^* \cdot n_{lj})}{\sum_{\gamma_{ki} \subset \Gamma_{lj}} T_{ki}} \quad \text{on } \partial K_l$$

(Here i runs over the underlying fine grid)

Instead of generalizing standard MFEM basis functions, NSUM includes localized subgrid variations in the approximation spaces:

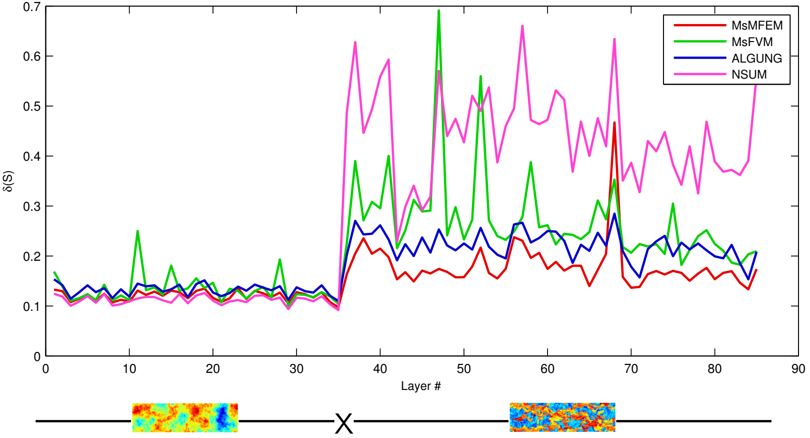
$$W_{H,h} = W_H \bigoplus_{T_i \in \mathcal{T}_H(\Omega)} W_h(T_i) = W_H \oplus W_h,$$

$$\mathbf{V}_{H,h} = \mathbf{V}_H \bigoplus_{T_i \in \mathcal{T}_H(\Omega)} \mathbf{V}_h(T_i) = \mathbf{V}_H \oplus \mathbf{V}_h.$$

- Both the coarse- and fine-scale spaces can be any standard MFEM spaces.
- The most common choices are BDM1 on the coarse scale and RT0 on the fine scale.
- Localization, $\mathbf{v_h} \cdot \mathbf{n} = 0, \ \forall \mathbf{v}_h \in \mathbf{V}_h(T_i)$, limits inter-element flow to be determined by the coarse-scale basis only.

Multiscale versus upscaling methods
SPE 10, individual layers

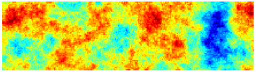
Saturation errors at 0.3 PVI on 15×55 coarse grid

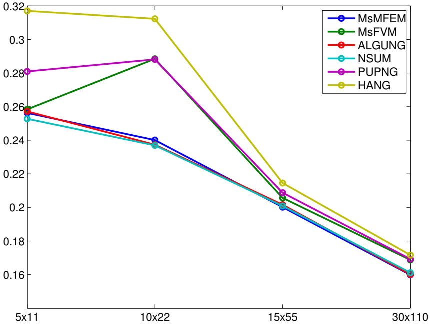


Multiscale versus upscaling methods

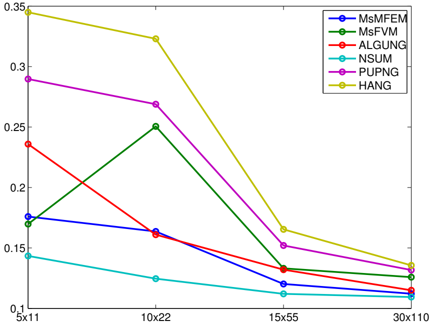
Average saturation errors on Tarbert formation (Layers 1–35)

Cartesian coarse grids:
Multiscale methods give enhanced accuracy only when subgrid information is exploited





Coarse-grid simulation

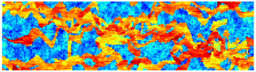


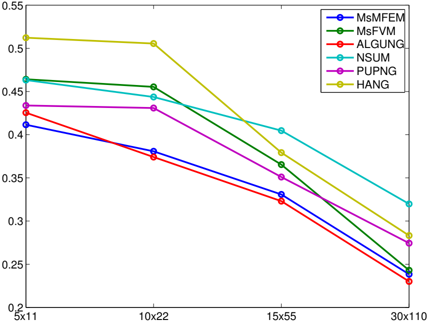
Fine-grid simulation

Multiscale versus upscaling methods

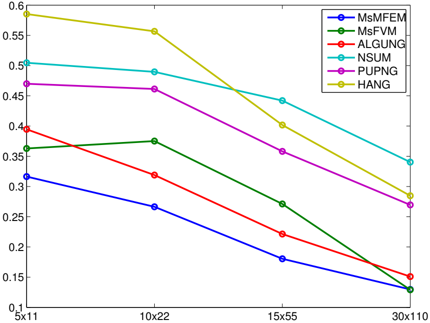
Average saturation errors on Upper Næss formation (Layers 36–85)

Cartesian coarse grids:
Multiscale methods give enhanced accuracy only when subgrid information is exploited





Coarse-grid simulation



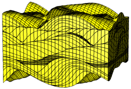
Fine-grid simulation

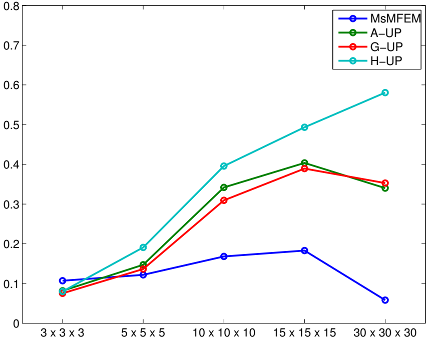
Multiscale versus upscaling methods

Up-gridded $30 \times 30 \times 333$ corner-point grid with layered log-normal permeability

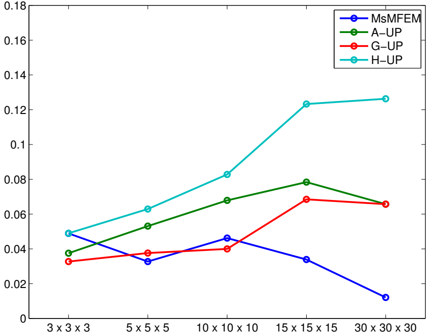
Complex coarse grid-block geometries:

MsMFEM is more accurate than upscaling, also for coarse-grid simulation.





Coarse-grid velocity errors

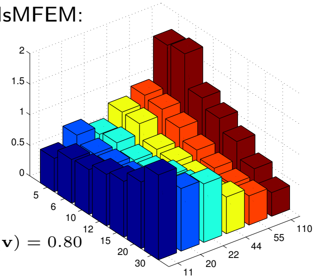


Coarse-grid saturation errors

Multiscale versus upscaling methods

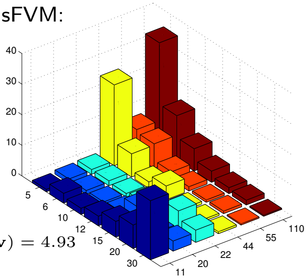
Velocity errors for Layer 85

MsMFEM:



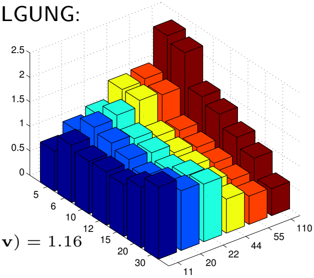
$\overline{\delta}(\mathbf{v}) = 0.80$

MsFVM:



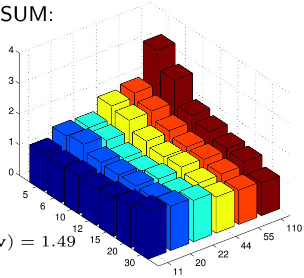
$\overline{\delta}(\mathbf{v}) = 4.93$

ALGUNG:



$\overline{\delta}(\mathbf{v}) = 1.16$

NSUM:

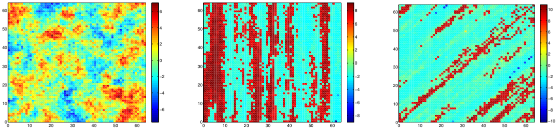


$\overline{\delta}(\mathbf{v}) = 1.49$

Multiscale versus upscaling methods

Synthetic test suite: permeability generated by `sgsim` from GSLIB

100 realisations of three different scenarios



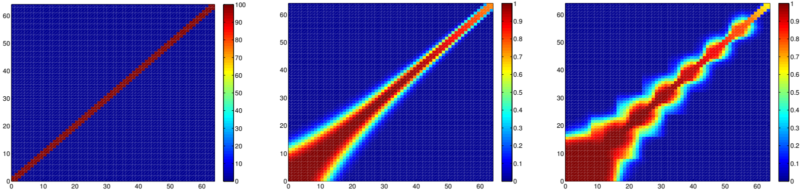
Scenario 1 Scenario 2 Scenario 3

Summary of observations:

- All methods give good results on log-normal permeability
- Long correlation lengths:
  - MsFVM sometimes gives very inaccurate velocity fields
  - NSUM has limited ability to model variations across coarse-mesh interfaces
  - MsMFEM has reduced accuracy for strong diagonal channels
- MsMFEM most accurate in terms of saturation errors
- ALGUNG is very robust, but uses *global information*

An idealized, but illustrative special case

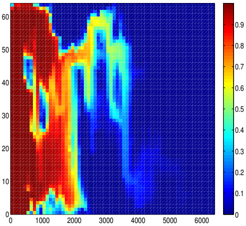
- MsMFEM looses accuracy for cases with strong diagonal channels hitting corners of coarse grid blocks
- Flow 45° to grid faces must take a detour into neighbouring coarse element
- If the channel crosses element faces (dual-grid corners), the problem disappears for MsMFEM but appears for MsFVM ...



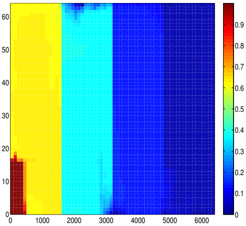
Permeability Reference MsMFEM

Scenario 1 with $\Delta x = 100\Delta y$

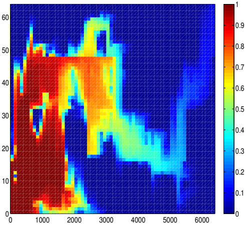
- MsFVM is highly inaccurate on fine scale, but acceptable on coarse scale
- Fine-grid fluxes large relative to coarse-grid fluxes $\longrightarrow$ oscillatory boundary conditions that introduce circular currents
- Problems reduced by using nested gridding to reconstruct fine-scale velocity



Reference saturation



MsFVM saturation



MsFVM + NG

Multiscale versus upscaling methods

Computational complexity: order-of-magnitude argument

Assume:

- Grid model with $N = N_s * N_c$ cells:
  - N_c number of coarse cells
  - N_s number of fine cells in each coarse cell
- Linear solver of complexity $\mathcal{O}(m^\alpha)$ for $m \times m$ system
- Negligible work for determining local b.c., numerical quadrature, and assembly (can be important, especially for NSUM)

Direct solution

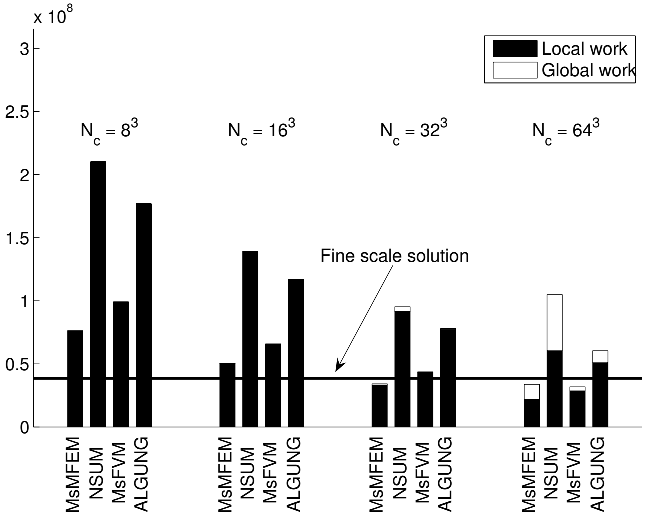
N^α operations for a two-point finite volume method

MsMFEM

Computing basis functions:	$D \cdot N_c \cdot (2N_s)^\alpha$ operations
Solving coarse-scale system:	$(D \cdot N_c)^\alpha$ operations

Multiscale versus upscaling methods

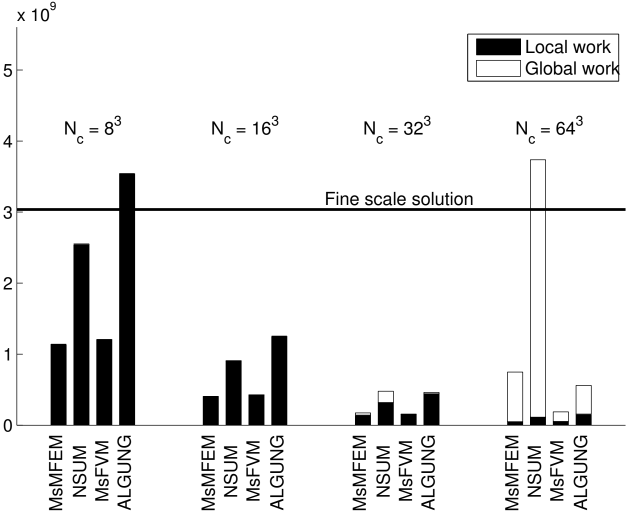
Example: $128 \times 128 \times 128$ fine grid



Comparison with algebraic multigrid, $\alpha = 1.2$

Multiscale versus upscaling methods

Example: $128 \times 128 \times 128$ fine grid



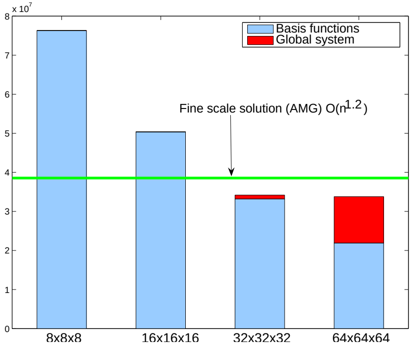
Comparison with less efficient solver, $\alpha = 1.5$

Direct solution may be more efficient, so why bother with multiscale?

- Full simulation: $\mathcal{O}(10^2)$ time steps.
- Basis functions need not be recomputed

Also:

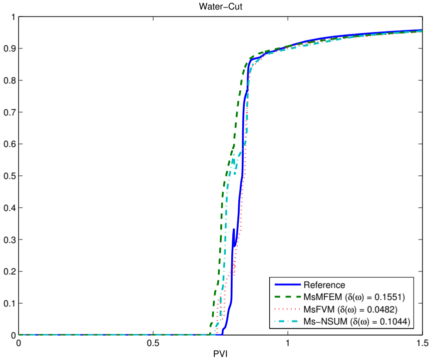
- Possible to solve very large problems
- Easy parallelization



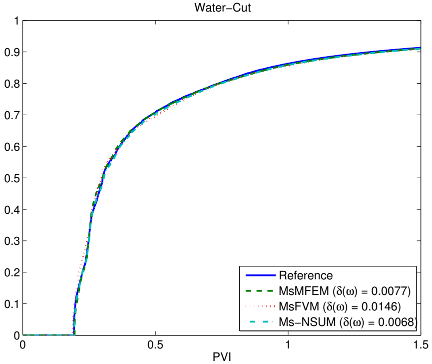
Multiphase Flow

Example: quarter five-spot, Layer 85 from SPE 10, coarse grid: 10×22

Water cuts obtained by never updating basis functions:



favorable ($M = 0.1$)

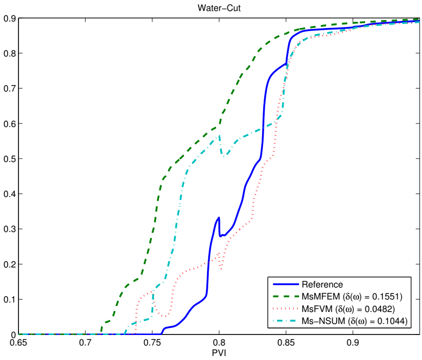


unfavorable ($M = 10.0$)

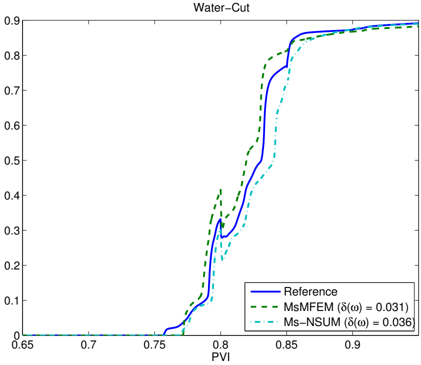
Multiphase Flow

Example: quarter five-spot, Layer 85 from SPE 10, coarse grid: 10×22

Improved accuracy by adaptive updating of basis functions:



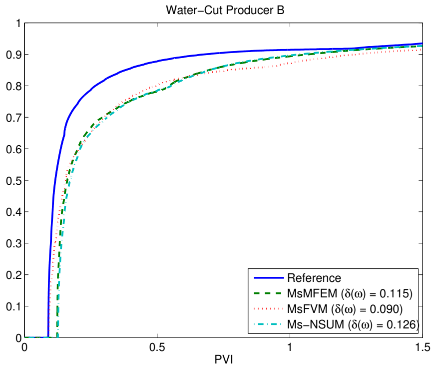
no updating



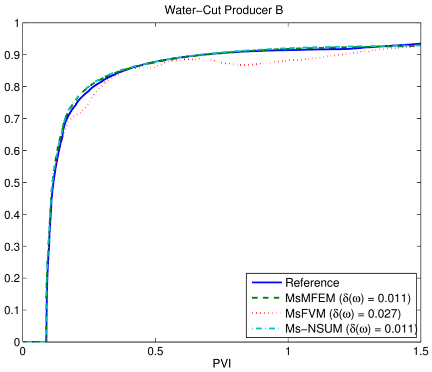
adaptive updating

Example: five-spot, Layer 68 from SPE 10, coarse grid: 15×55

Improved accuracy by using global information (initial fine-scale solution):



local b.c.



global b.c.

Implementation details for MsMFE

There are certain choices....

- Fine-grid discretization
- Generation of coarse grids
- Domain of support and boundary conditions
- Choice of weighting function in definition of basis functions
- Linear algebra

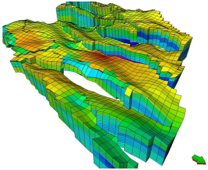
Discretization of the fine-grid problem

Complex reservoir geometries

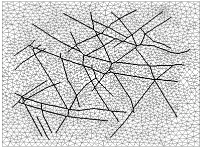
Challenges:

- Industry-standard grids are often nonconforming and contain skewed and degenerate cells
- There is a trend towards unstructured grids
- Standard discretization methods produce wrong results on skewed and rough cells
- The combination of high aspect and anisotropy ratios can give *very large* condition numbers

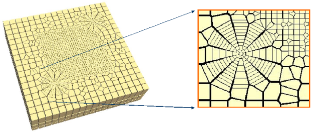
Corner point:



Tetrahedral:

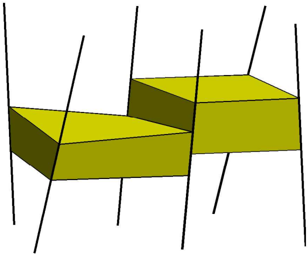


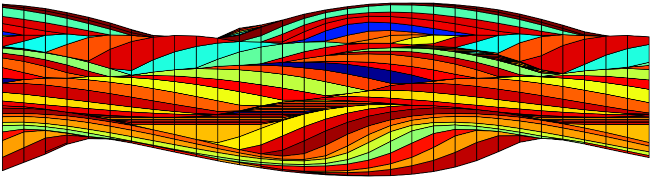
PEBI:



Corner-point grids:

- areal 2D mesh of vertical or inclined pillars
- each volumetric cell is restriced by four pillars
- each cell is defined by eight corner points, two on each pillar

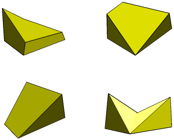




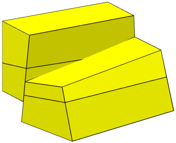
Discretization of the fine-grid problem

Cell geometries are challenging from a discretization point-of-view

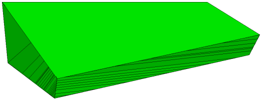
Skewed and deformed blocks:



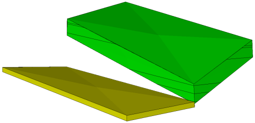
Non-matching cells:



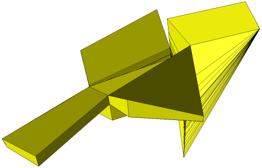
Many faces:



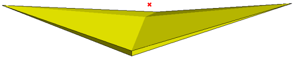
Small interfaces:



Difficult geometries:



(Very) high aspect ratios:



$800 \times 800 \times 0.25$ m

Discretization of the fine-grid problem

The mimetic finite difference method

Mimetic finite-difference methods may be interpreted as a finite-volume counterpart of mixed finite-element methods.

Key features:

- Applicable for models with general polyhedral grid-cells.
- Allow easy treatment of non-conforming grids with complex grid-cell geometries (including curved faces).
- Generic implementation: same code applies to all grids (e.g., corner-point/PEBI, matching/non-matching, ...).

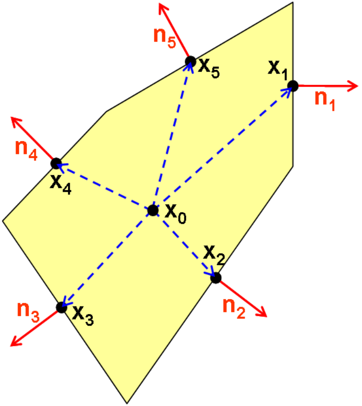
Discretization of the fine-grid problem

The mimetic finite difference method, Brezzi *et al.*, 2005

Express fluxes $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)^\mathsf{T}$ as:

$$\boldsymbol{v} = -\boldsymbol{T}(\boldsymbol{p} - p_0),$$

where $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)^\mathsf{T}$.



Discretization of the fine-grid problem

The mimetic finite difference method, Brezzi *et al.*, 2005

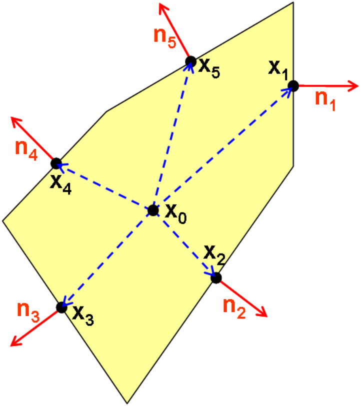
Express fluxes $\boldsymbol{v} = (v_1, v_2, \dots, v_n)^\mathsf{T}$ as:

$$\boldsymbol{v} = -\boldsymbol{T}(\boldsymbol{p} - p_0),$$

where $\boldsymbol{p} = (p_1, p_2, \dots, p_n)^\mathsf{T}$.
Impose exactness for any *linear* pressure field $p = \boldsymbol{x}^\mathsf{T}\boldsymbol{a} + c$ (which gives velocity equal to $-\mathbf{K}\boldsymbol{a}$):

$$v_i = -A_i \boldsymbol{n}_i^\mathsf{T} \mathbf{K} \boldsymbol{a}$$

$$p_i - p_0 = (\boldsymbol{x}_i - \boldsymbol{x}_0)^\mathsf{T} \boldsymbol{a}.$$



Discretization of the fine-grid problem

The mimetic finite difference method, Brezzi *et al.*, 2005

Express fluxes $\boldsymbol{v} = (v_1, v_2, \dots, v_n)^\mathsf{T}$ as:

$$\boldsymbol{v} = -\boldsymbol{T}(\boldsymbol{p} - p_0),$$

where $\boldsymbol{p} = (p_1, p_2, \dots, p_n)^\mathsf{T}$.
Impose exactness for any *linear* pressure field $p = \boldsymbol{x}^\mathsf{T}\boldsymbol{a} + c$ (which gives velocity equal to $-\mathbf{K}\boldsymbol{a}$):

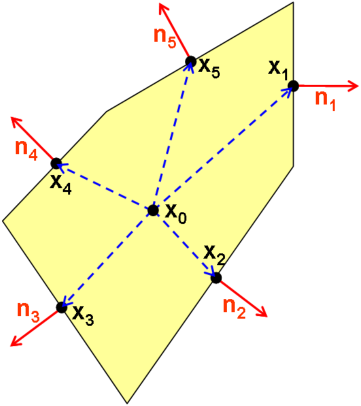
$$v_i = -A_i \boldsymbol{n}_i^\mathsf{T} \mathbf{K}\boldsymbol{a}$$

$$p_i - p_0 = (\boldsymbol{x}_i - \boldsymbol{x}_0)^\mathsf{T}\boldsymbol{a}.$$

As a result, $\mathbf{T}$ must satisfy

$$\mathbf{T} \times \mathbf{C} = \mathbf{N} \times \mathbf{K}$$

where $\boldsymbol{C}(i,:) = (\boldsymbol{x}_i - \boldsymbol{x}_0)^\mathsf{T}$ and $\boldsymbol{N}(i,:) = A_i \boldsymbol{n}_i^\mathsf{T}$



Discretization of the fine-grid problem

The mimetic finite difference method, Brezzi *et al.*, 2005

Express fluxes $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)^\mathsf{T}$ as:

$$\boldsymbol{v} = -\boldsymbol{T}(\boldsymbol{p} - p_0),$$

where $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)^\mathsf{T}$.
Impose exactness for any *linear* pressure field $p = \boldsymbol{x}^\mathsf{T}\boldsymbol{a} + c$ (which gives velocity equal to $-\mathbf{K}\boldsymbol{a}$):

$$v_i = -A_i \boldsymbol{n}_i^\mathsf{T} \mathbf{K} \boldsymbol{a}$$

$$p_i - p_0 = (\boldsymbol{x}_i - \boldsymbol{x}_0)^\mathsf{T} \boldsymbol{a}.$$

As a result, $\mathbf{T}$ must satisfy

$$\mathbf{T} \times \mathbf{C} = \mathbf{N} \times \mathbf{K}$$

where $\boldsymbol{C}(i,:) = (\boldsymbol{x}_i - \boldsymbol{x}_0)^\mathsf{T}$ and $\boldsymbol{N}(i,:) = A_i \boldsymbol{n}_i^\mathsf{T}$

Family of valid solutions:

$$\boldsymbol{T} = \frac{1}{|E|} \boldsymbol{N} \mathbf{K} \boldsymbol{N}^\mathsf{T} + \boldsymbol{T}_2,$$

where $\boldsymbol{T}_2$ is such that $\boldsymbol{T}$ is s.p.d. and $\boldsymbol{T}_2 \boldsymbol{C} = \boldsymbol{O}$.

Discretization of the fine-grid problem

The mimetic finite difference method, Brezzi *et al.*, 2005

Express fluxes $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)^\mathsf{T}$ as:

$$\boldsymbol{v} = -\boldsymbol{T}(\boldsymbol{p} - p_0),$$

where $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)^\mathsf{T}$.
Impose exactness for any *linear* pressure field $p = \boldsymbol{x}^\mathsf{T}\boldsymbol{a} + c$ (which gives velocity equal to $-\mathbf{K}\boldsymbol{a}$):

$$v_i = -A_i \boldsymbol{n}_i^\mathsf{T} \mathbf{K} \boldsymbol{a}$$

$$p_i - p_0 = (\boldsymbol{x}_i - \boldsymbol{x}_0)^\mathsf{T} \boldsymbol{a}.$$

As a result, $\mathbf{T}$ must satisfy

$$\mathbf{T} \times \mathbf{C} = \mathbf{N} \times \mathbf{K}$$

where $\boldsymbol{C}(i,:) = (\boldsymbol{x}_i - \boldsymbol{x}_0)^\mathsf{T}$ and $\boldsymbol{N}(i,:) = A_i \boldsymbol{n}_i^\mathsf{T}$

Family of valid solutions:

$$\boldsymbol{T} = \frac{1}{|E|} \boldsymbol{N} \mathbf{K} \boldsymbol{N}^\mathsf{T} + \boldsymbol{T}_2,$$

where $\boldsymbol{T}_2$ is such that $\boldsymbol{T}$ is s.p.d. and $\boldsymbol{T}_2 \boldsymbol{C} = \boldsymbol{O}$.

Imposing continuity across edges/faces and conservation yields a *hybrid* system:

$$\begin{pmatrix} \boldsymbol{B} & \boldsymbol{C} & \boldsymbol{D} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{D}^\mathsf{T} & \boldsymbol{O} & \boldsymbol{O} \end{pmatrix} \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{p} \\ \boldsymbol{\pi} \end{pmatrix} = \mathrm{RHS}$$

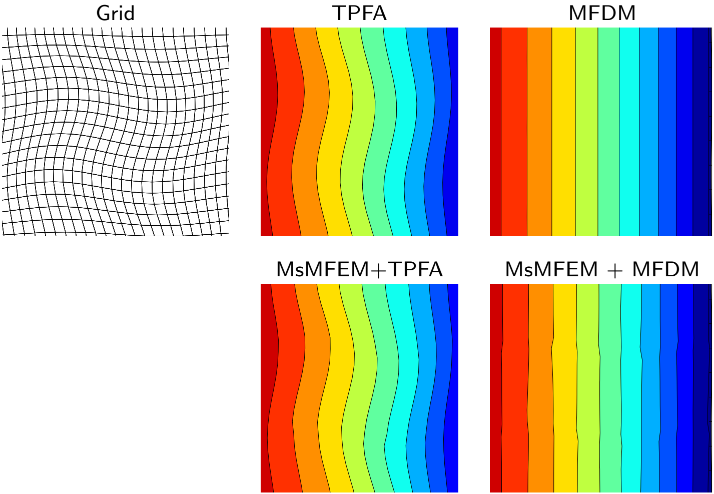
$$\Downarrow$$

Reduces to s.p.d. system for face pressures $\boldsymbol{\pi}$.

Discretization of the fine-grid problem

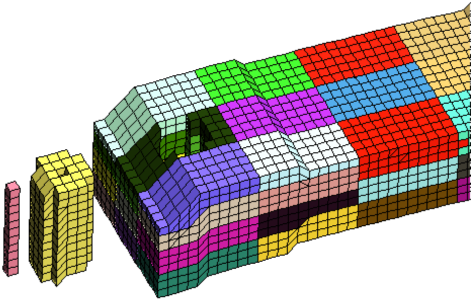
Mimetic: method applicable to general polyhedral cells

Single phase, homogeneous **K**, linear pressure drop



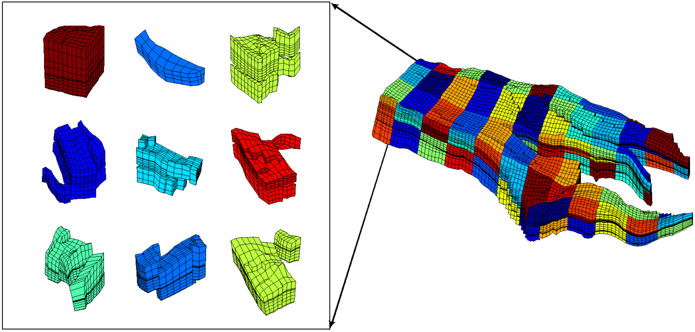
(Unique) grid flexibility:

Given a method that can solve local flow problems on the subgrid, the MsMFE method can be formulated on any coarse grid in which the coarse blocks consist of a connected collection of fine-grid cells



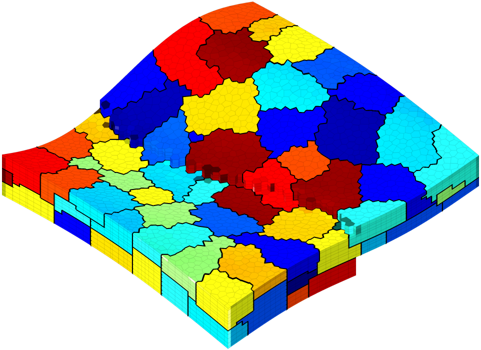
(Unique) grid flexibility:

Given a method that can solve local flow problems on the subgrid, the MsMFE method can be formulated on any coarse grid in which the coarse blocks consist of a connected collection of fine-grid cells



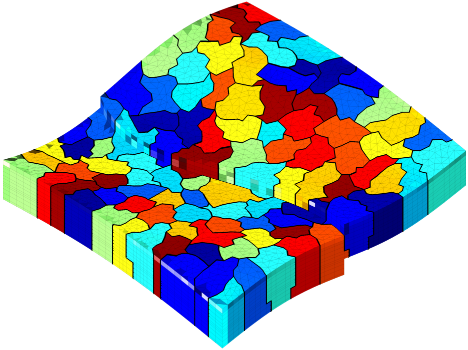
(Unique) grid flexibility:

Given a method that can solve local flow problems on the subgrid, the MsMFE method can be formulated on any coarse grid in which the coarse blocks consist of a connected collection of fine-grid cells



(Unique) grid flexibility:

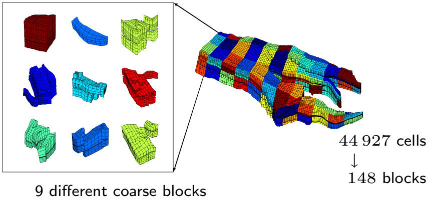
Given a method that can solve local flow problems on the subgrid, the MsMFE method can be formulated on any coarse grid in which the coarse blocks consist of a connected collection of fine-grid cells



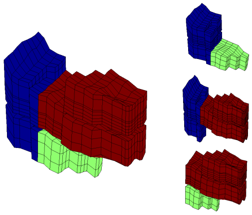
Generation of coarse grids

Workflow with automated upgridding in 3D

1) Coarsen grid by uniform partitioning in index space for corner-point grids

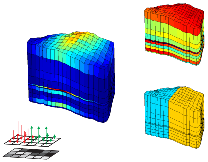


2) Detect all adjacent blocks



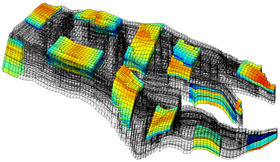
3) Compute basis functions

$$\nabla\cdot\psi_{ij} = \begin{cases} w_i(x), \\ -w_j(x), \end{cases}$$



for all pairs of blocks

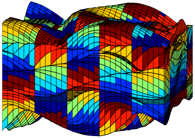
4) Block in coarse grid: component for building global solution



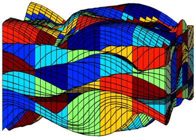
A depositional bed

Eroded layers gives a large number of degenerate and inactive cells.
Relative error in saturation at 0.5PVI:

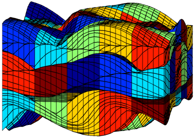
Coarse grid	Isotropic	Anisotropic	Heterogeneous
Physical	0.1339	0.2743	0.2000
Logical	0.0604	0.1381	0.1415
Constrained	0.0573	0.1479	0.0993



physical



logical

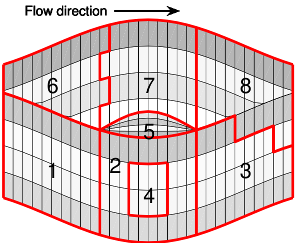


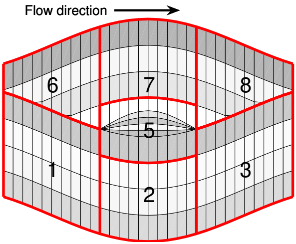
constrained

Generation of coarse grids

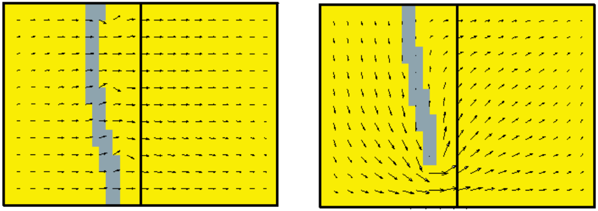
Simple guidelines for choosing good coarse grids

1. Minimize bidirectional flow over interfaces:
   - Avoid unnecessary irregularity ($\Gamma_{6,7}$ and $\Gamma_{3,8}$)
   - Avoid single neighbors (T_4)
   - Ensure that there are faces transverse to flow direction (T_5)
2. Blocks and faces should follow geological layers (T_3 and T_8)
3. Blocks should adapt to flow obstacles whenever possible
4. For efficiency: minimize the number of connections
5. Avoid having too many small blocks





Problems occur when a basis function tries to force flow through a flow barrier



problem | no problem

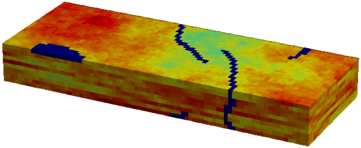
Can be detected automatically through the indicator

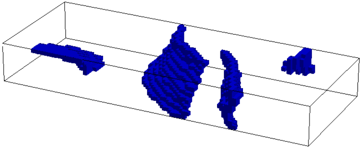
$$v_{ij} = \psi_{ij} \cdot (\lambda K)^{-1} \psi_{ij}$$

If $v_{ij}(x) > C$ for some $x \in T_i$, then split T_i and generate basis functions for the new faces

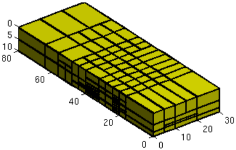
Generation of coarse grids

Example: adaption to flow obstacles

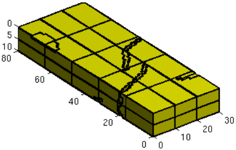




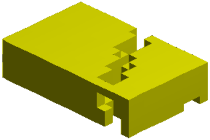
Non-uniform grid, hexahedral cells



Non-uniform grid, general cells

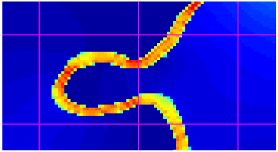


General grid-cell

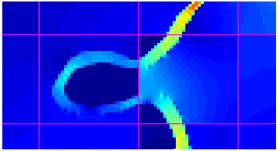


Generation of coarse grids
Problems with crossflow

Problems if there is a strong bi-directional flow over a coarse-grid interface



fine grid



multiscale

Can be detected automatically through the indicator

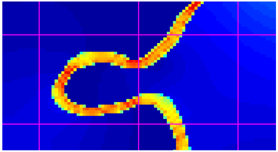
$$|\int_{\Gamma_{ij}} v\cdot n\,ds| \ll \int_{\Gamma_{ij}} |v\cdot n|\,ds, \qquad c \le \int_{\Gamma_{ij}} |v\cdot n|\,ds$$

If so, split T_i and generate basis functions for the new faces.

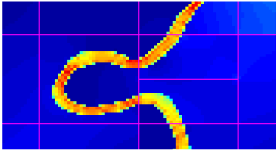
Generation of coarse grids

Problems with crossflow

Problems if there is a strong bi-directional flow over a coarse-grid interface



fine grid



multiscale

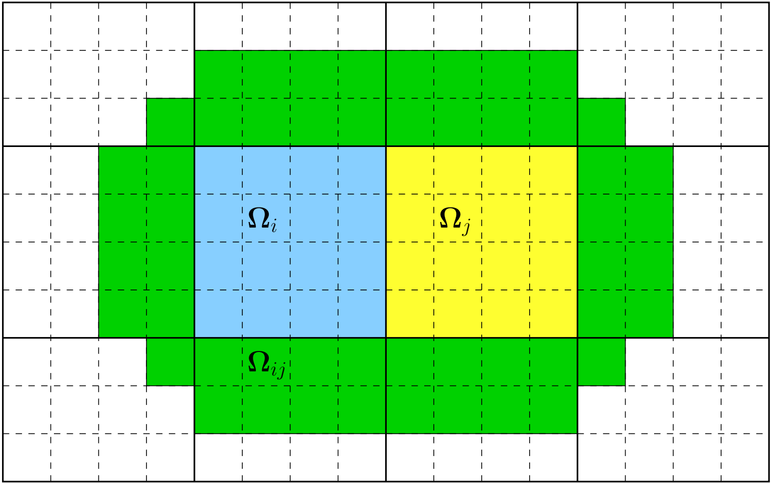
Can be detected automatically through the indicator

$$\left|\int_{\Gamma_{ij}} v \cdot n \, ds\right| \ll \int_{\Gamma_{ij}} |v \cdot n| \, ds, \qquad c \leq \int_{\Gamma_{ij}} |v \cdot n| \, ds$$

If so, split T_i and generate basis functions for the new faces.

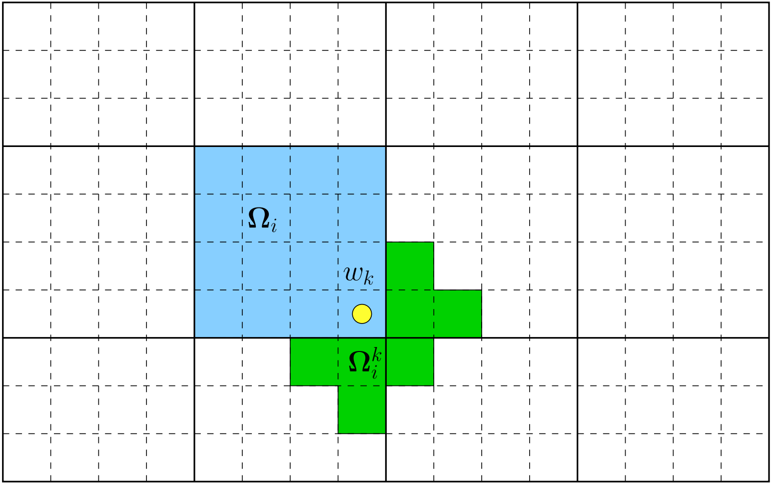
Domain of support and boundary conditions

Overlap may often increase accuracy



Domain of support and boundary conditions

Overlap may often increase accuracy



Key observation:

If $v \in V^{\text{ms}}$, then the MsMFE solution $v_{h,H}$ replicates v regardless of heterogeneity (barriers, channels, etc) and grid.

The pressure p_H is an exact w-weighted average in each grid block

$$p_H|_{B_i} = \int_{B_i} p w_i \, dx$$

Question:

Is it possible to define basis functions so that $v \in V^{\text{ms}}$?

Yes, $v \in V^{\text{ms}}$ if

$$\psi_{ij} \cdot n_{ij} = \frac{v \cdot n_{ij}}{\int_{\Gamma_{ij}} v \cdot n_{ij} \, ds}$$

Domain of support and boundary conditions

Invoking global information

Assume that we have computed v, e.g., on a fine grid using either true or generic boundary conditions (and wells)

Global basis functions:

$$\nabla \cdot \vec{\psi}_{ij} = \begin{cases} w_i(\vec{x}), & \text{if } \vec{x} \in B_i, \\ -w_j(\vec{x}), & \text{if } \vec{x} \in B_j \end{cases}$$

$$\vec{\psi}_{ij} \cdot \vec{n} = 0, \text{on } \partial(B_i \cap B_j) \qquad \vec{\psi}_{ij} \cdot \vec{n}_{ij} = \frac{v \cdot n_{ij}}{\int_{\Gamma_{ij}} v \cdot n_{ij}\, ds} \text{on } \Gamma_{ij}$$

Rationale

Pressure needs to be solved repeatedly in multiphase flow. Hence, can afford fine-scale solution.

Also: bootstrapping local-global MsMFE, use of more than one basis function per interface, etc

The weight function distributes $\nabla \cdot v$ on the coarse blocks:

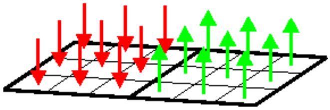
$$(\nabla \cdot v)|_{\Omega_i} = \sum_j v_{ij} (\nabla \cdot \psi_{ij})|_{\Omega_i} = w_i \sum_j v_{ij}$$
$$= w_i \int_{\partial \Omega_i} v \cdot n \, ds = w_i \int_{\Omega_i} \nabla \cdot v \, dx$$

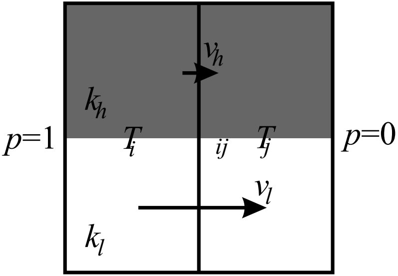
Different roles:

Incompressible flow: $\nabla \cdot v = q$
Compressible flow: $\nabla \cdot v = q - c_t \partial_t p - \sum_j c_j v_j \cdot \nabla p$

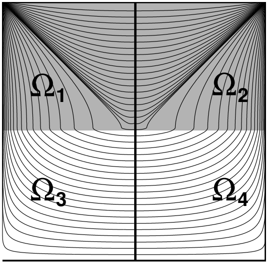
Uniform source:

$$w_i(x) = \frac{1}{|T_i|}$$





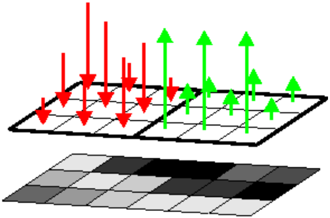
low (k_l) and high (k_h) permeability

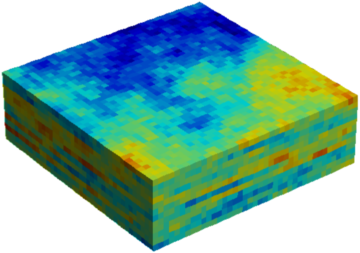


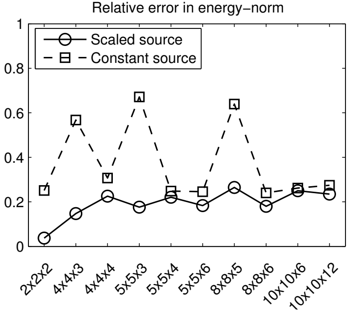
streamlines from basis function

Scaled source:

$$w_i(x) = \frac{\text{trace}(K(x))}{\int_{T_i} \text{trace}(K(\xi))\, d\xi}$$







The role of the weight function

Choice of weight function, $w_i = \theta(x) / \int_{\Omega_i} \theta(x)\,dx$

Incompressible flow:

$$\int_{\Omega_i} q dx = 0, \qquad \theta(x) = \mathrm{trace}(\mathbf{K}(x))$$

$$\int_{\Omega_i} q dx \neq 0, \qquad \theta(x) = q(x)$$

The role of the weight function

Choice of weight function, $w_i = \theta(x)/\int_{\Omega_i}\theta(x)\,dx$

Incompressible flow:

$$\int_{\Omega_i} q dx = 0, \qquad \theta(x) = \mathrm{trace}(\mathbf{K}(x))$$
$$\int_{\Omega_i} q dx \neq 0, \qquad \theta(x) = q(x)$$

Compressible flow:

- $\theta \propto q$: compressibility effects concentrated where $q \neq 0$
- $\theta \propto \mathbf{K}$: $\nabla \cdot v$ over/underestimated for high/low $\mathbf{K}$

Another choice motivated by physics:

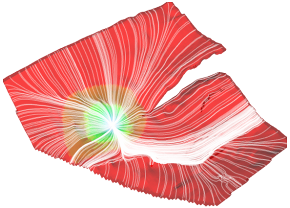
$$\theta(x) = \phi(x), \qquad \text{Motivation: } c_t \frac{\partial p}{\partial t} \propto \phi$$

Usage and outlook

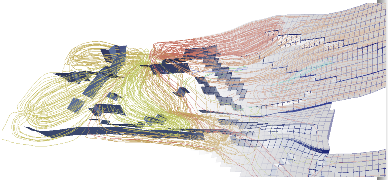
Multiscale methods need efficient transport solvers

- Streamline methods
  - intuitive visualization + new data
  - subscale resolution
  - good scaling, known to be efficient

Flow pattern (CO2 injection):



Connections across faults:

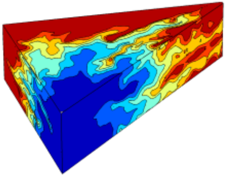


Usage and outlook

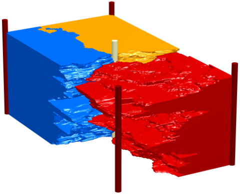
Multiscale methods need efficient transport solvers

- Streamline methods
  - intuitive visualization + new data
  - subscale resolution
  - good scaling, known to be efficient

Time-of-flight (timelines):



Flooded volumes (stationary tracer):

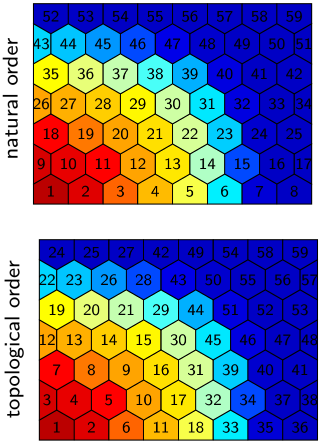


Usage and outlook

Multiscale methods need efficient transport solvers

- Streamline methods
  - intuitive visualization + new data
  - subscale resolution
  - good scaling, known to be efficient
- Optimal ordering
  - same assumptions as for streamlines
  - utilize causality $\longrightarrow \mathcal{O}(n)$ algorithm, cell-by-cell solution
  - local control over (non)linear iterations

Topological sorting

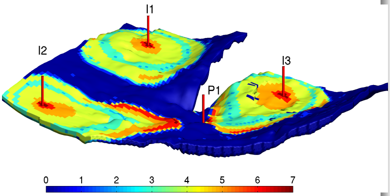


Usage and outlook

Multiscale methods need efficient transport solvers

- Streamline methods
  - intuitive visualization + new data
  - subscale resolution
  - good scaling, known to be efficient
- Optimal ordering
  - same assumptions as for streamlines
  - utilize causality $\longrightarrow \mathcal{O}(n)$ algorithm, cell-by-cell solution
  - local control over (non)linear iterations

Local iterations:



Johansen formation: 27 437 active cells

Global vs local Newton–Raphson solver

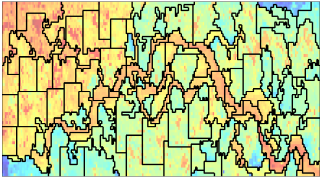
Δt	global		local	
days	time	iter	time (sec)	iter
125	2.26	12.69	0.044	0.93
250	2.35	12.62	0.047	1.10
500	2.38	13.25	0.042	1.41
1000	2.50	13.50	0.042	1.99

Usage and outlook

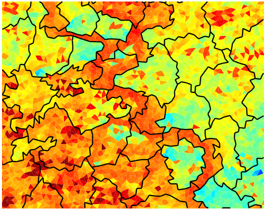
Multiscale methods need efficient transport solvers

- Streamline methods
  - intuitive visualization + new data
  - subscale resolution
  - good scaling, known to be efficient
- Optimal ordering
  - same assumptions as for streamlines
  - utilize causality $\longrightarrow \mathcal{O}(n)$ algorithm, cell-by-cell solution
  - local control over (non)linear iterations
- Flow-based coarsening
  - agglomeration of cells $\longrightarrow$ simple and flexible coarsening
  - hybrid griding schemes
  - heterogeneous multiscale method?
  - efficient model reduction

Cartesian grid:



Triangular grids:

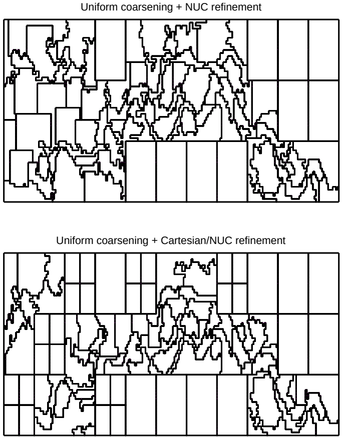


Usage and outlook

Multiscale methods need efficient transport solvers

- Streamline methods
  - intuitive visualization + new data
  - subscale resolution
  - good scaling, known to be efficient
- Optimal ordering
  - same assumptions as for streamlines
  - utilize causality $\longrightarrow \mathcal{O}(n)$ algorithm, cell-by-cell solution
  - local control over (non)linear iterations
- Flow-based coarsening
  - agglomeration of cells $\longrightarrow$ simple and flexible coarsening
  - hybrid griding schemes
  - heterogeneous multiscale method?
  - efficient model reduction

Different partitioning:

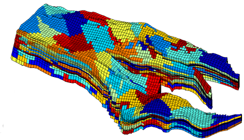


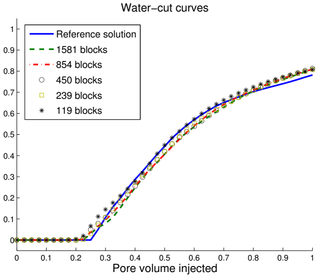
Usage and outlook

Multiscale methods need efficient transport solvers

- Streamline methods
  - intuitive visualization + new data
  - subscale resolution
  - good scaling, known to be efficient
- Optimal ordering
  - same assumptions as for streamlines
  - utilize causality $\longrightarrow \mathcal{O}(n)$ algorithm, cell-by-cell solution
  - local control over (non)linear iterations
- Flow-based coarsening
  - agglomeration of cells $\longrightarrow$ simple and flexible coarsening
  - hybrid griding schemes
  - heterogeneous multiscale method?
  - efficient model reduction

Model reduction by coarsening:





Usage and outlook

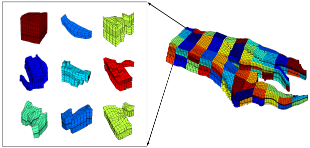
For what purpose are multiscale methods useful?

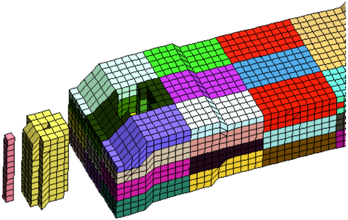
- As robust upscaling methods?
- As alternative to upscaling and fine-scale solution?
- To provide flow simulation earlier in the modelling loop?
- To get 90% of the answer in 10% of the time?
- Fit-for-purpose solvers in workflows for ranking, history matching, planning, optimization, ...

Usage and outlook

Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening



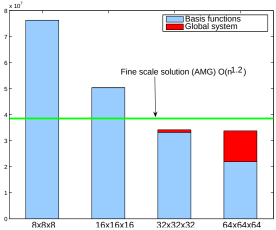


Usage and outlook

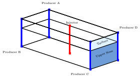
Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening
- Reuse of computations, key to computational efficiency

Operations vs. upscaling factor:



SPE10: 1.1 mill cells



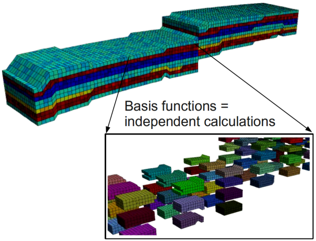
Inhouse code from 2005:

Multiscale: 2 min and 20 sec
Multigrid: 8 min and 36 sec

Usage and outlook

Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening
- Reuse of computations, key to computational efficiency
- Natural (elliptic) parallelism:
  - giga-cell simulations
  - multicore and heterogeneous computing

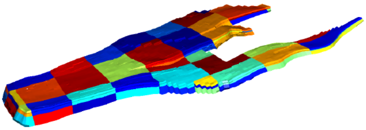


Usage and outlook

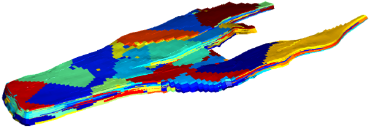
Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening
- Reuse of computations, key to computational efficiency
- Natural (elliptic) parallelism:
  - giga-cell simulations
  - multicore and heterogeneous computing
- Fine-scale velocity ⟶ different grid for flow and transport ⟶ dynamical adaptivity

Pressure grid:



Transport grid:

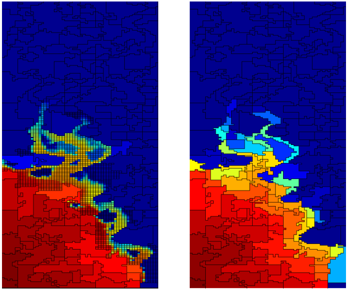


Usage and outlook

Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening
- Reuse of computations, key to computational efficiency
- Natural (elliptic) parallelism:
  - giga-cell simulations
  - multicore and heterogeneous computing
- Fine-scale velocity ⟶ different grid for flow and transport ⟶ dynamical adaptivity

Flow-based gridding:



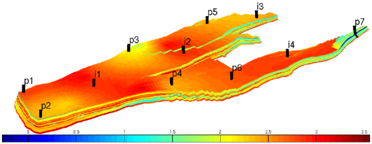
with and without dynamic Cartesian refinement

Usage and outlook

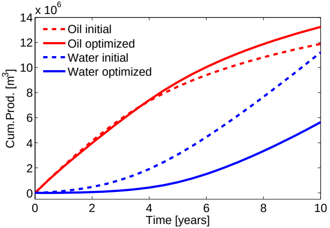
Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening
- Reuse of computations, key to computational efficiency
- Natural (elliptic) parallelism:
  - giga-cell simulations
  - multicore and heterogeneous computing
- Fine-scale velocity ⟶ different grid for flow and transport ⟶ dynamical adaptivity
- Method for model reduction:
  - adjoint simulations ⟶ approximate gradients
  - ensemble simulations with representative basis functions

Water-flood optimization:



Reservoir geometry from a Norwegian Sea field



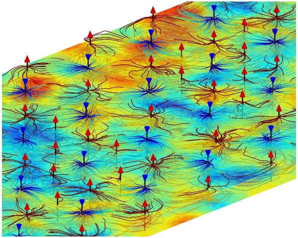
Forward simulations:
44 927 cells, 20 time steps, $<$ 5 sec in Matlab

Usage and outlook

Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening
- Reuse of computations, key to computational efficiency
- Natural (elliptic) parallelism:
  - giga-cell simulations
  - multicore and heterogeneous computing
- Fine-scale velocity $\longrightarrow$ different grid for flow and transport $\longrightarrow$ dynamical adaptivity
- Method for model reduction:
  - adjoint simulations $\longrightarrow$ approximate gradients
  - ensemble simulations with representative basis functions

History matching 1 million cells:



7 years: 32 injectors, 69 producers

Generalized travel-time inversion + multiscale:
7 forward simulations, 6 inversions

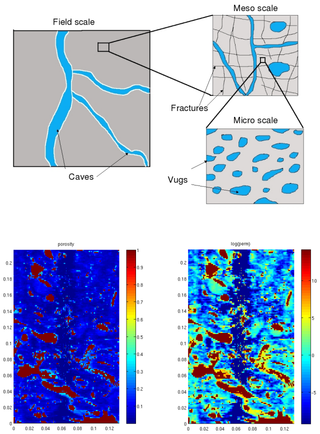
Solver	CPU-time (wall clock)		
	Total	Pres.	Transp.
Multigrid	39 min	30 min	5 min
Multiscale	17 min	7 min	6 min

Usage and outlook

Success stories and unreaped potential

- More flexible wrt grids than standard upscaling methods: automatic coarsening
- Reuse of computations, key to computational efficiency
- Natural (elliptic) parallelism:
  - giga-cell simulations
  - multicore and heterogeneous computing
- Fine-scale velocity $\longrightarrow$ different grid for flow and transport $\longrightarrow$ dynamical adaptivity
- Method for model reduction:
  - adjoint simulations $\longrightarrow$ approximate gradients
  - ensemble simulations with representative basis functions
- Multiphysics applications

Stokes–Brinkmann:



Usage and outlook

What are the limitations?

Capabilities:

- ✓ Two-phase flow
- ✓ Cartesian / unstructured grids
- ✓ Realistic flow physics $\Rightarrow$ iterations
  - Correction functions + smoothing
  - Residual formulation + domain decomposition
- ✓ Pointwise accuracy $\Rightarrow$ iterations

Usage and outlook

What are the limitations?

Capabilities:

- ✓ Two-phase flow
- ✓ Cartesian / unstructured grids
- ✓ Realistic flow physics ⇒ iterations
  - Correction functions + smoothing
  - Residual formulation + domain decomposition
- ✓ Pointwise accuracy ⇒ iterations

Not yet there:

- Compressible three-phase black-oil + non-Cartesian grids
- Fully implicit formulation
- Parallelization
- Compositional, thermal, ...

Usage and outlook

What are the limitations?

Other issues:

- How to choose good coarse grids for unstructured grids?
- Need for global information or iterative procedures?
- A posteriori error analysis (resolution or fine-scale junk)?
- More than two levels in hierarchical grid?
- How to include models from finer scales?