Multiscale Methods for Reservoir Simulation

Knut–Andreas Lie

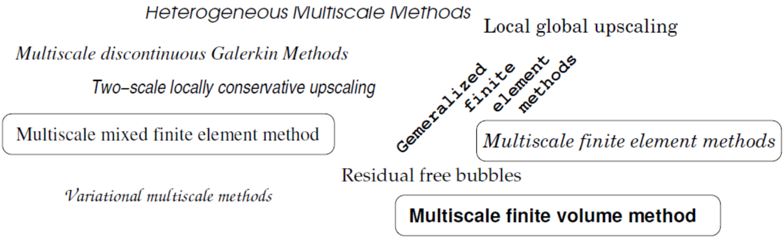
Department of Mathematics and Cybernetics, SINTEF Digital/
Department of Mathematical Sciences, NTNU, Norway

Multiscale Methods Summer School
June 26–29, 2017, Hasselt, Belgium

Outline

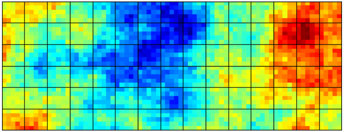
Multiscale methods

Numerical methods that attempt to model physical phenomena on coarse grids while honoring small-scale features in an appropriate way consistent with the local property of the differential operator

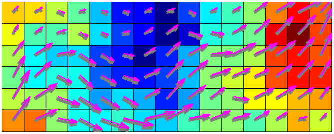


Multiscale methods versus upscaling

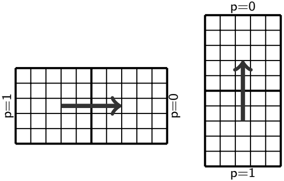
Coarse partitioning:



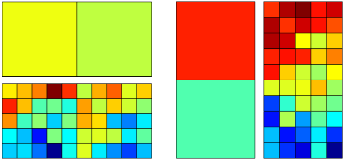
Coarse-scale solution:



Localized flow problems:

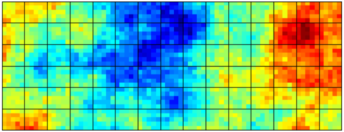


Compute effective parameters:

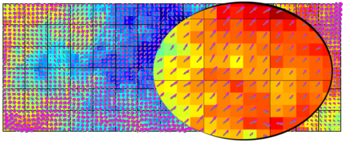


Multiscale methods versus upscaling

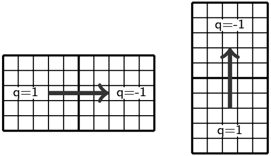
Coarse partitioning:



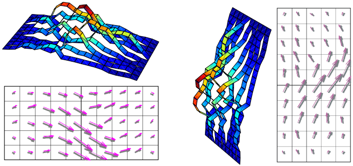
Flow field with subresolution:



Localized flow problems:

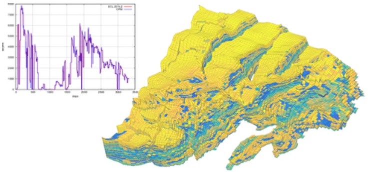


Flow solution → basis functions:



From Poisson's equation to reservoir simulation

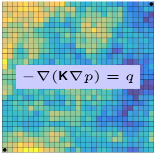
Flow physics



$$
\begin{aligned}
0 =& \partial_t(\phi b_o S_o) + \nabla \cdot (b_o \vec{v_o}) - b_o q_o \\
0 =& \partial_t(\phi b_w S_w) + \nabla \cdot (b_w \vec{v_w}) - b_w q_w \\
0 =& \partial_t\big[\phi(b_g S_b + b_o r_{so} S_o)\big] + \nabla \cdot (b_g \vec{v_g}) \\
& + \nabla \cdot (b_o r_{so} \vec{v_o}) - b_g q_g - b_o r_{so} q_o
\end{aligned}
$$

???

MsFV, MsMFE, 2003



$$-\nabla(\mathbf{K}\nabla p) = q$$

Geology

From concept to commercial deployment

Two main tracks for commercial simulation: multiscale finite-volume (MsFV) and multiscale mixed finite-element (MsMFE) methods

Property	MsFE	MsMFE	MsFV	MsRSB
Conservative velocity field	×	✓	✓	✓
Applicable to unstructured grids	×	✓	×	✓
Robustness: aspect ratio / high contrast	✓	✓	?	✓
Compressible flow	×	?	✓	✓
Systematic error control	✓	?	✓	✓
Locally smooth	✓	×	✓	✓
Partition of unity	✓	×	✓	✓
Efficient	✓	✓	✓	✓

Disclaimer: many methods and a lot of academic research will not be covered in the following

Outline

The multiscale finite-element (MsFE) method

Model problem

Variable-coefficient Poisson problem in 1D

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

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

Variational formulation

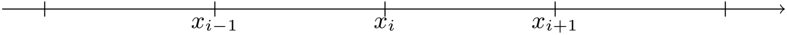
Find $p \in H_0^1(\Omega)$ such that

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

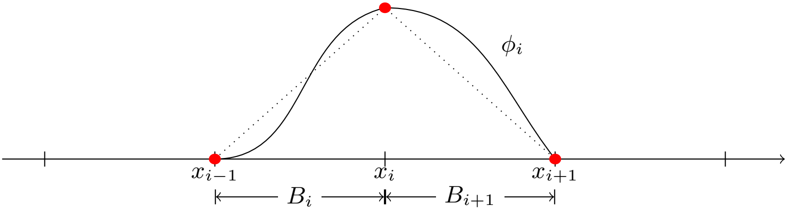
where $(\cdot,\cdot)$ is the L^2 inner-product and

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

The MsFE method



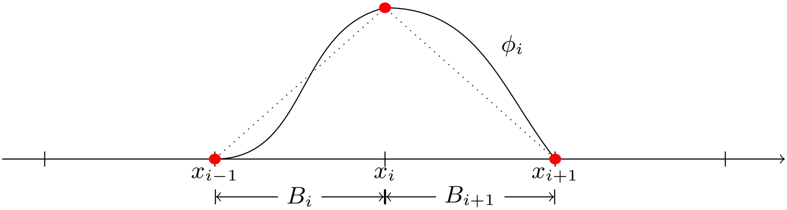
The MsFE method



For $i = 1, \ldots, n-1$, we define a basis function $\phi_i \in H_0^1(\Omega)$ by

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

The MsFE method



For $i = 1, \ldots, n-1$, we define a basis function $\phi_i \in H_0^1(\Omega)$ by

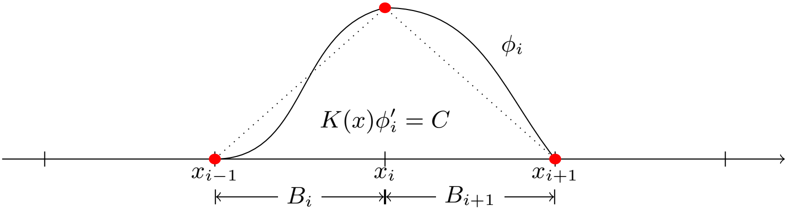
$$a(\phi_i, \varphi) = 0 \qquad \text{for all } \varphi \in H_0^1(B_i \cup B_{i+1}), \qquad \phi_i(x_j) = \delta_{ij},$$

Multiscale basis function associated with node x_i is given as

$$-\big(K(x)\partial_x \phi_i'(x)\big)' = 0, \qquad x \in [x_{i-1}, x_{i+1}] = B_i \cup B_{i+1}$$

Obviously, $K(x)\phi_i' = C$, for some constant C

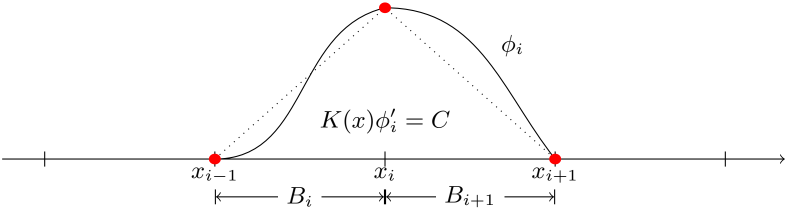
The MsFE method



Integrating over B_i and using the prescribed values $\phi_i(x_{i-1}) = 0$ and $\phi_i(x_i) = 1$ gives

$$\int_{x_{i-1}}^{x_i} \phi_i'(x)\, dx = \phi_i(x_i) - \phi_i(x_{i-1}) = 1 = \int_{x_{i-1}}^{x_i} \frac{C}{K(x)}\, dx$$

The MsFE method



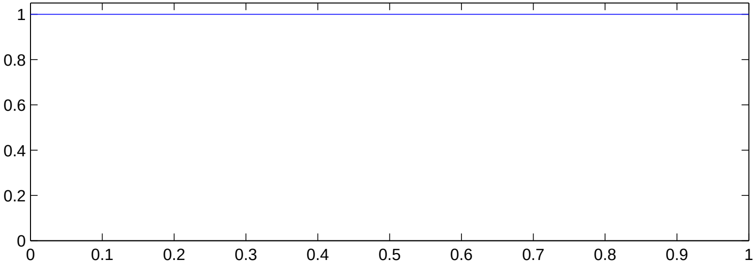
Integrating over B_i and using the prescribed values $\phi_i(x_{i-1}) = 0$ and $\phi_i(x_i) = 1$ gives

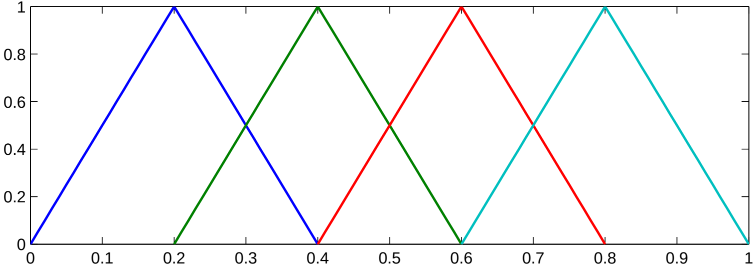
$$\int_{x_{i-1}}^{x_i} \phi_i'(x)\, dx = \phi_i(x_i) - \phi_i(x_{i-1}) = 1 = \int_{x_{i-1}}^{x_i} \frac{C}{K(x)}\, dx$$

From which it follows that for $x \in B_i = [x_{i-1}, x_i]$

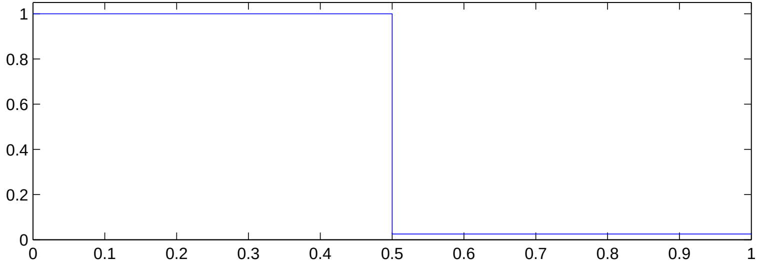
$$\phi_i'(x) = \frac{1/K(x)}{\int_{x_{i-1}}^{x_i} \frac{1}{K(x)}\, dx} \quad \Longrightarrow \quad \phi_i(x) = \frac{\int_{x_{i-1}}^{x} \frac{1}{K(x)}\, dx}{\int_{x_{i-1}}^{x_i} \frac{1}{K(x)}\, dx}$$

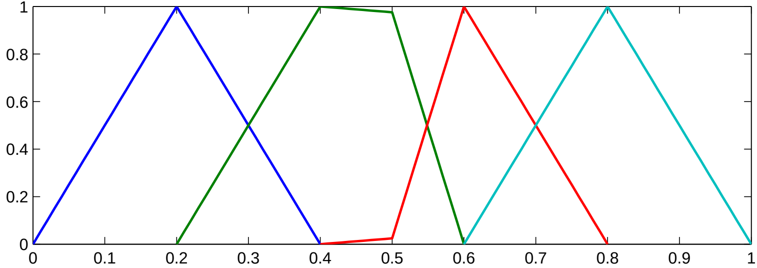
The MsFE method: basis functions



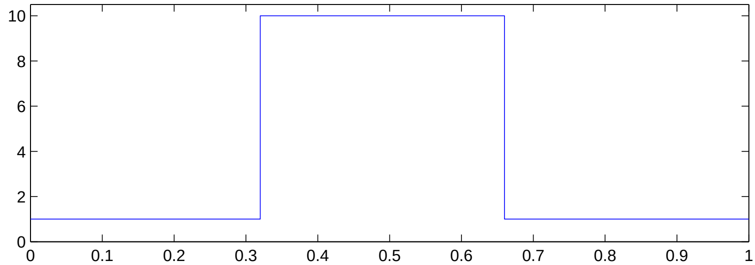


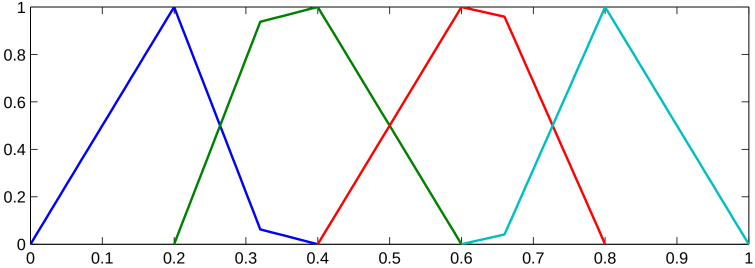
The MsFE method: basis functions



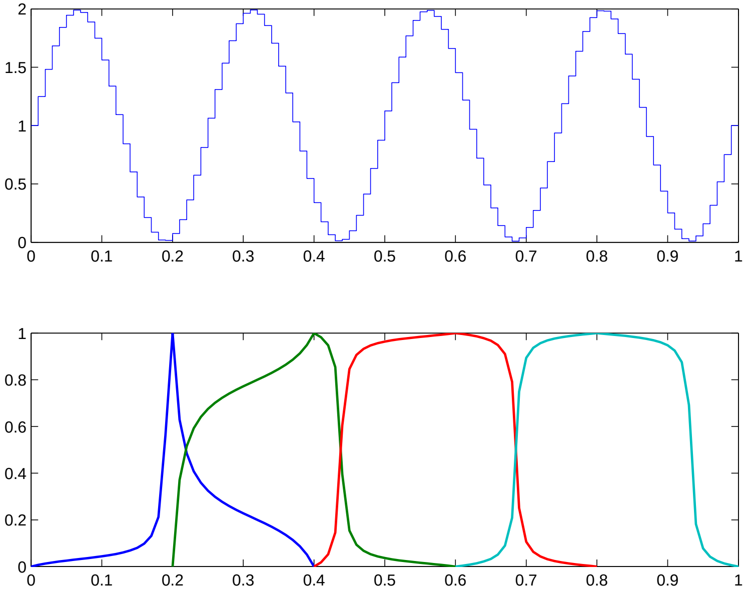


The MsFE method: basis functions





The MsFE method: basis functions



The MsFE method: patch refinement property

The MsFE method

Find the unique function p_0 in

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

satisfying

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

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

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

The MsFE method: patch refinement property

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

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

The MsFE method: patch refinement property

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

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

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

$$a(p - p_0, \varphi) = a(p, \varphi) - a(p_0, \varphi) = (f, \varphi) - (f, \varphi) = 0$$

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

The MsFE method: patch refinement property

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

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

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

$$a(p - p_0, \varphi) = a(p, \varphi) - a(p_0, \varphi) = (f, \varphi) - (f, \varphi) = 0$$

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}

The MsFE method: patch refinement property

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

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

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

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

The MsFE method: patch refinement property

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

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

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

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

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

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

The MsFE method: patch refinement property

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

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

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

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

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

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

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

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

which implies that $p_0 = p_I$

The MsFE method: patch refinement property

Theorem

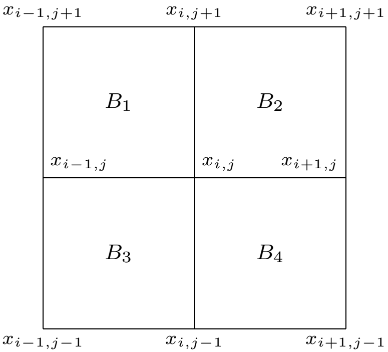
Assume that p solves the variational formulation. Then $p = p_0 + \sum_{i=1}^{n} p_i$, where $p_i \in H_0^1(B_i)$ is defined by

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

In other words: the solution of the variational problem is decomposed into the MsFE solution and solutions of independent local subgrid problems.

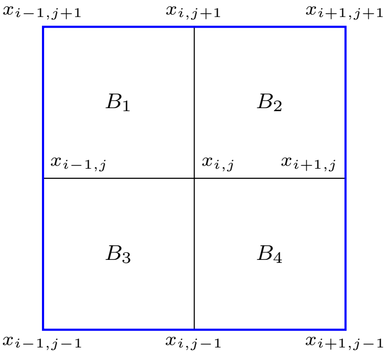
This result does not extend to higher dimensions, but the basic construction applies and helps us understand how subgrid features of the solution can be embodied into a coarse grid approximation space

The MsFE method in 2D



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

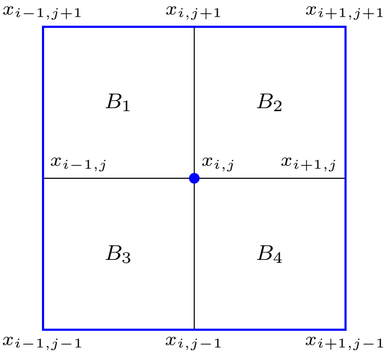
The MsFE method in 2D



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

$\phi^{ij} = 0$ on block interface not emanating from $x_{i,j}$

The MsFE method in 2D

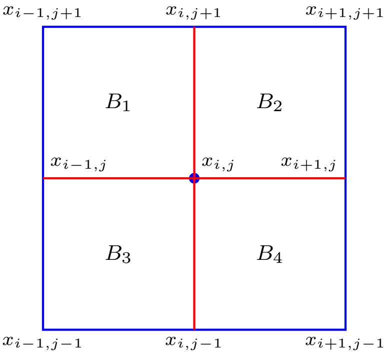


$p \in V^{\text{ms}}$ implies that $\nabla \cdot \mathbf{K} \nabla \phi^{ij} = 0$ in all coarse blocks B_m

$\phi^{ij} = 0$ on block interface not emanating from $x_{i,j}$

$\phi^{ij}(x_{m,n}) = \delta_{i,m}\delta_{j,n}$

The MsFE method in 2D



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

$\phi^{ij} = 0$ on block interface 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

Outline

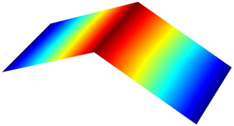
The multiscale mixed finite-element method

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

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

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 (e.g., RT0)



$$H_0^{1,\text{div}} = \{\vec{v} \in L^2(\Omega)^d \ : \ \nabla \cdot \vec{v} \in L^2(\Omega) \text{ and } \vec{v} \cdot \vec{n} = 0 \text{ on } \partial\Omega\}$$

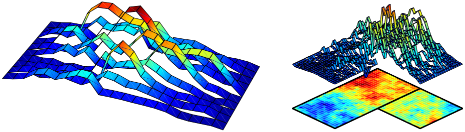
The multiscale mixed finite-element method

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

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

Multiscale MFE method

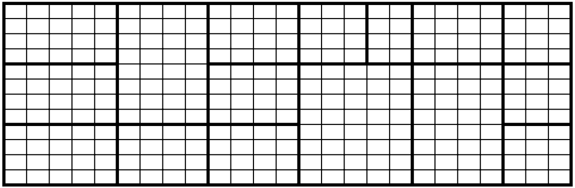
- Seek solution in $\mathbf{V}_{H,h} \times W_{H,h} \subset H_0^{1,\mathsf{div}} \times L^2$
- Approximation spaces: local numerical solutions



$$H_0^{1,\mathsf{div}} = \{\vec{v} \in L^2(\Omega)^d \ : \ \nabla \cdot \vec{v} \in L^2(\Omega) \text{ and } \vec{v} \cdot \vec{n} = 0 \text{ on } \partial\Omega\}$$

Hierarchical grids and basis functions

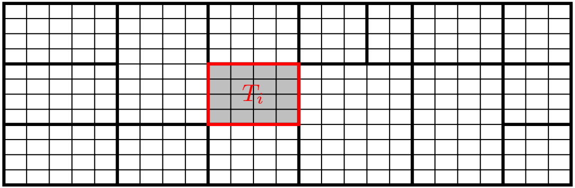
Fine grid with petrophysical parameters cell



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

Hierarchical grids and basis functions

Fine grid with petrophysical parameters cell

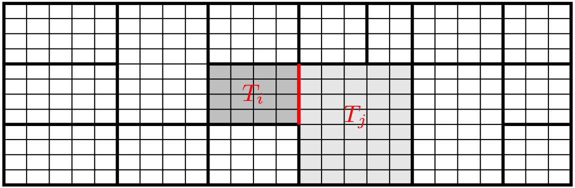


Construct a *coarse* grid, and choose the discretisation spaces V and U^{ms} such that:

- For each coarse block T_i, there is at least one basis function $\phi_i \in V$

Hierarchical grids and basis functions

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 at least one basis function $\phi_i \in V$
- For each coarse edge Γ_{ij}, there is at least one basis function $\psi_{ij} \in U^{ms}$

Basis functions $\phi_i(x,y)$ and $\vec{\psi}_{ij}(x,y)$ are computed numerically by solving a local flow problem, using an artificial source term to drive a unit flow over the interface between two pairs of blocks

Coarse-scale mixed system

The coarse-scale system can be derived algebraically from a fine-scale discretization. Here, we will use a mixed formulation.

Fine-scale system:

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ -\boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{q} \end{bmatrix},$$

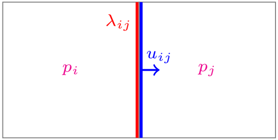
$$b_{ij} = \int_\Omega \psi_i \left(\lambda K\right)^{-1} \psi_j \, dx,$$

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

Alternatively – mixed hybrid form:

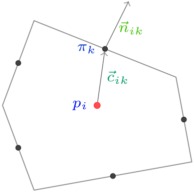
$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} & \boldsymbol{D} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{D}^\mathsf{T} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ -\boldsymbol{p} \\ \boldsymbol{\pi} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{q} \\ \boldsymbol{0} \end{bmatrix}$$

$$d_{ik} = \int_{\partial\Omega} |\psi_i \cdot n_k| \, dx$$



Multipoint method:

- Darcy: $\boldsymbol{u}_i = \boldsymbol{T}_i(\boldsymbol{e}_i \boldsymbol{p}_i - \boldsymbol{\pi}_i)$
- Mass conservation for all cells
- Continuity of fluxes across faces

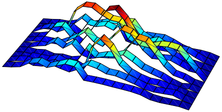


Coarse-scale mixed system

Make the following assumption

$$\boldsymbol{u} = \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}}$$

$$\boldsymbol{p} = \boldsymbol{\mathcal{I}}\boldsymbol{p}_c + \tilde{\boldsymbol{p}}$$



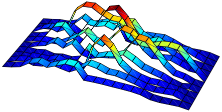
$\boldsymbol{\Psi}$ – matrix with basis functions
$\boldsymbol{\mathcal{I}}$ – prolongation from blocks to cells

Coarse-scale mixed system

Make the following assumption

$$\boldsymbol{u} = \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}}$$

$$\boldsymbol{p} = \boldsymbol{\mathcal{I}}\boldsymbol{p}_c + \tilde{\boldsymbol{p}}$$



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

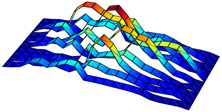
Reduction to coarse-scale system:

$$\begin{bmatrix} \boldsymbol{\Psi}^{\mathsf{T}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mathcal{I}}^{\mathsf{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^{\mathsf{T}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}} \\ -\boldsymbol{\mathcal{I}}\boldsymbol{p}_c - \tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\mathcal{I}}^{\mathsf{T}}\boldsymbol{q} \end{bmatrix}$$

Coarse-scale mixed system

Make the following assumption

$$\boldsymbol{u} = \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}}$$
$$\boldsymbol{p} = \boldsymbol{\mathcal{I}}\boldsymbol{p}_c + \tilde{\boldsymbol{p}}$$



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

Reduction to coarse-scale system:

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mathcal{I}}^\mathsf{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}} \\ -\boldsymbol{\mathcal{I}}\boldsymbol{p}_c - \tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{q} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{B}\boldsymbol{\Psi} & \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{C}\boldsymbol{\mathcal{I}} \\ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{C}^\mathsf{T}\boldsymbol{\Psi} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_c \\ -\boldsymbol{p}_c \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{B}\tilde{\boldsymbol{u}} \; + \; \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{C}\tilde{\boldsymbol{p}} \\ \boldsymbol{q}_c - \; \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{C}^\mathsf{T}\tilde{\boldsymbol{u}} \end{bmatrix}$$

Coarse-scale mixed system

Make the following assumption

$$\boldsymbol{u} = \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}}$$

$$\boldsymbol{p} = \boldsymbol{\mathcal{I}}\boldsymbol{p}_c + \tilde{\boldsymbol{p}}$$

Multiscale basis function:

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi} \\ -\boldsymbol{\Phi} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{w} \end{bmatrix}$$

Set of equations located to coarse blocks. Flow driven by weight w

Reduction to coarse-scale system:

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\mathcal{I}}^\mathsf{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}} \\ -\boldsymbol{\mathcal{I}}\boldsymbol{p}_c - \tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{q} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{B}\boldsymbol{\Psi} & \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{C}\boldsymbol{\mathcal{I}} \\ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{C}^\mathsf{T}\boldsymbol{\Psi} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_c \\ -\boldsymbol{p}_c \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{B}\tilde{\boldsymbol{u}} \ + \ \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{C}\tilde{\boldsymbol{p}} \\ \boldsymbol{q}_c - \ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{C}^\mathsf{T}\tilde{\boldsymbol{u}} \end{bmatrix}$$

Coarse-scale mixed system

Make the following assumption

$$\boldsymbol{u} = \boldsymbol{\Psi} \boldsymbol{u}_c + \tilde{\boldsymbol{u}}$$
$$\boldsymbol{p} = \boldsymbol{\mathcal{I}} \boldsymbol{p}_c + \tilde{\boldsymbol{p}}$$

Multiscale basis function:

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi} \\ -\boldsymbol{\Phi} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{w} \end{bmatrix}$$

Set of equations located to coarse blocks. Flow driven by weight w

Reduction to coarse-scale system:

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\mathcal{I}}^\mathsf{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi} \boldsymbol{u}_c + \tilde{\boldsymbol{u}} \\ -\boldsymbol{\mathcal{I}} \boldsymbol{p}_c - \tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\mathcal{I}}^\mathsf{T} \boldsymbol{q} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{B} \boldsymbol{\Psi} & \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{C} \boldsymbol{\mathcal{I}} \\ \boldsymbol{\mathcal{I}}^\mathsf{T} \boldsymbol{C}^\mathsf{T} \boldsymbol{\Psi} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_c \\ -\boldsymbol{p}_c \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\Psi}^\mathsf{T} \boldsymbol{B} \tilde{\boldsymbol{u}} + \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{C} \tilde{\boldsymbol{p}} \\ \boldsymbol{q}_c - \boldsymbol{\mathcal{I}}^\mathsf{T} \boldsymbol{C}^\mathsf{T} \tilde{\boldsymbol{u}} \end{bmatrix}$$

Additional assumptions:

Since p is immaterial, assume $\boldsymbol{w}^\mathsf{T} \tilde{\boldsymbol{p}} = 0$.
Hence, $p_c^i = \int_{\Omega_i} wp \, dx$

Coarse-scale mixed system

Make the following assumption

$$\boldsymbol{u} = \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}}$$
$$\boldsymbol{p} = \boldsymbol{\mathcal{I}}\boldsymbol{p}_c + \tilde{\boldsymbol{p}}$$

Multiscale basis function:

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi} \\ -\boldsymbol{\Phi} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{w} \end{bmatrix}$$

Set of equations located to coarse blocks. Flow driven by weight w

Reduction to coarse-scale system:

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\mathcal{I}}^\mathsf{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi}\boldsymbol{u}_c + \tilde{\boldsymbol{u}} \\ -\boldsymbol{\mathcal{I}}\boldsymbol{p}_c - \tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{q} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{B}\boldsymbol{\Psi} & \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{C}\boldsymbol{\mathcal{I}} \\ \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{C}^\mathsf{T}\boldsymbol{\Psi} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_c \\ -\boldsymbol{p}_c \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{B}\tilde{\boldsymbol{u}} + \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{C}\tilde{\boldsymbol{p}} \\ \boldsymbol{q}_c - \boldsymbol{\mathcal{I}}^\mathsf{T}\boldsymbol{C}^\mathsf{T}\tilde{\boldsymbol{u}} \end{bmatrix}$$

Additional assumptions:

Since p is immaterial, assume $\boldsymbol{w}^\mathsf{T}\tilde{\boldsymbol{p}} = 0$. Hence, $p_c^i = \int_{\Omega_i} wp\,dx$

Assume that $\boldsymbol{\Psi}$ spans velocity space, i.e., $\tilde{\boldsymbol{u}} \equiv \boldsymbol{0}$.

Subresolution in pressure

Why not also use the basis functions for pressure?

Pressure is immaterial, but still we need to scale the pressure basis functions. From the definition of the basis functions we have that

$$\boldsymbol{B\Psi} - \boldsymbol{C\Phi} = \mathbf{0} \quad \Longrightarrow \quad \boldsymbol{B\Psi u}_c - \boldsymbol{C\Phi u}_c = \mathbf{0}$$

which implies that $\mathbf{\Phi}$ and $\mathbf{\Psi}$ should scale similarly.

Subresolution in pressure

Why not also use the basis functions for pressure?

Pressure is immaterial, but still we need to scale the pressure basis functions. From the definition of the basis functions we have that

$$\boldsymbol{B\Psi} - \boldsymbol{C\Phi} = \boldsymbol{0} \quad \Longrightarrow \quad \boldsymbol{B\Psi u}_c - \boldsymbol{C\Phi u}_c = \boldsymbol{0}$$

which implies that $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ should scale similarly.

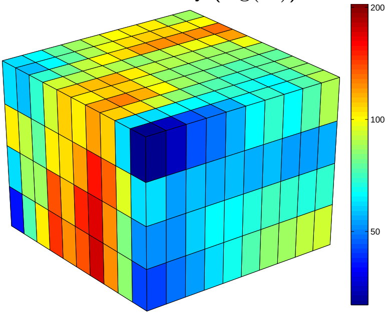
Hence, the starting-point for the algebraic reduction should be

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi u}_c \\ -\boldsymbol{\mathcal{I} p}_c - \boldsymbol{D}_\lambda \boldsymbol{\Phi u}_c \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{q} \end{bmatrix}$$

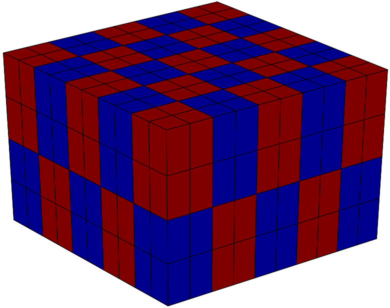
where $\boldsymbol{D}_\lambda = \mathrm{diag}(\lambda_i^0/\lambda_i)$ accounts for saturation variations

Example: linear systems

Permeability $(\log(K))$



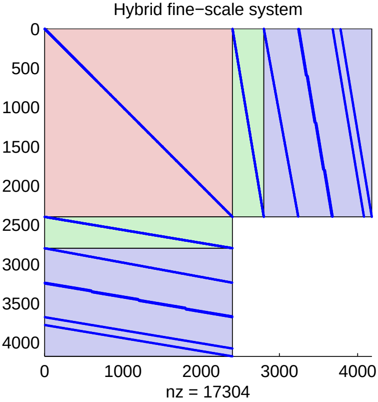
Coarse partition



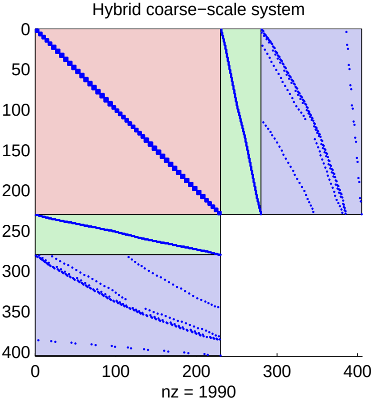
Simple flow problem:

Flux given on left boundary, $p = 0$ on right, no-flow elsewhere
Fine grid: $10 \times 10 \times 4$. Coarse grid: $5 \times 5 \times 2$

Example: linear systems

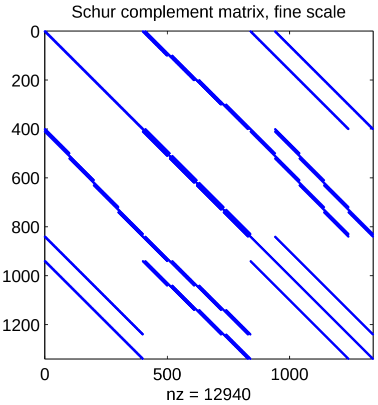


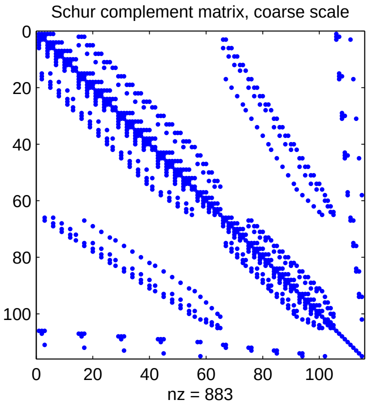
$$\begin{bmatrix} B & C & D \\ C^{\mathsf{T}} & 0 & 0 \\ D^{\mathsf{T}} & 0 & 0 \end{bmatrix}$$



$$\begin{bmatrix} \boldsymbol{\Psi}^{\mathsf{T}} \boldsymbol{B} \boldsymbol{\Psi} & \boldsymbol{\Psi}^{\mathsf{T}} \boldsymbol{C} \boldsymbol{\mathcal{I}} & \boldsymbol{\Psi}^{\mathsf{T}} \boldsymbol{D} \boldsymbol{\mathcal{J}} \\ \boldsymbol{\mathcal{I}}^{\mathsf{T}} \boldsymbol{C}^{\mathsf{T}} \boldsymbol{\Psi} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\mathcal{J}}^{\mathsf{T}} \boldsymbol{D}^{\mathsf{T}} \boldsymbol{\Psi} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Example: linear systems





Schur complement (block-wise Gauss elimination):

$$(\boldsymbol{D}^{\mathsf{T}}\boldsymbol{B}^{-1}\boldsymbol{D} - \boldsymbol{F}^{\mathsf{T}}\boldsymbol{L}^{-1}\boldsymbol{F})\boldsymbol{\pi} = \boldsymbol{F}^{\mathsf{T}}\boldsymbol{L}^{-1}\boldsymbol{g},$$

$$\boldsymbol{F} = \boldsymbol{C}^{\mathsf{T}}\boldsymbol{B}^{-1}\boldsymbol{D}, \quad \boldsymbol{L} = \boldsymbol{C}^{\mathsf{T}}\boldsymbol{B}^{-1}\boldsymbol{C}.$$

Basis functions

In Ω_i:

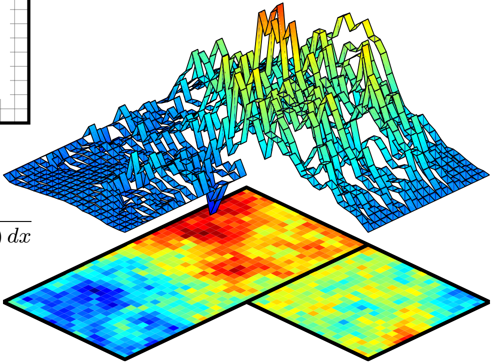
$$\vec{\psi}_{ij} = -K\nabla p$$
$$\nabla \cdot \vec{\psi}_{ij} = \omega_i$$

In Ω_j:

$$\vec{\psi}_{ij} = -K\nabla p$$
$$\nabla \cdot \vec{\psi}_{ij} = -\omega_j$$

Source ω_i:

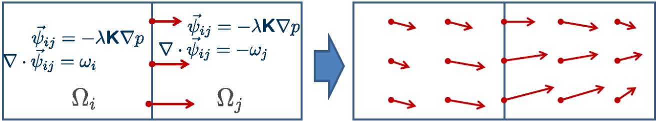
$$\omega_i(x) = \frac{K(x)}{\int_{\Omega_i} K(x)\,dx}$$



x-component of $\vec{\psi}_{ij}$

Basis functions

One-block approach:

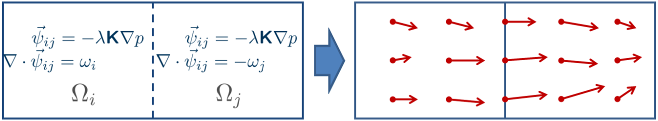


Boundary condition

$$\vec{\psi}_{i_j} \cdot \vec{n}_i = \nu_{ij} \quad \text{on } \Gamma_{ij}, \quad \vec{\psi}_{i_j} \cdot \vec{n}_i = 0 \quad \text{on } \partial B_i \setminus \Gamma_{ij}.$$

ν_{ij} determined by petrophysical properties (local) or flow solution (global)

Two-block approach:



No boundary condition on inner boundary. Not consistent, but accurate in practice.
Can also use overlap if desired

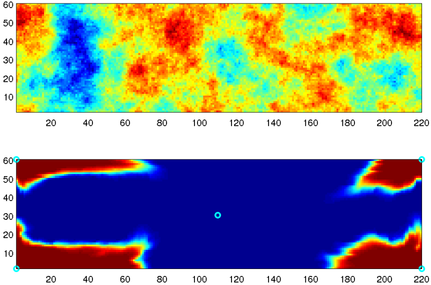
Comparison with upscaling methods

Model equations:

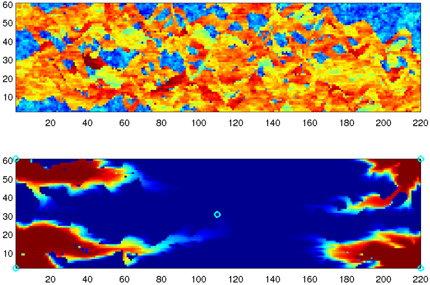
$$\nabla \cdot \vec{u} = q, \qquad \vec{u} = -\mathbf{K}\nabla p$$

$$S_t + \nabla \cdot \big(S\vec{u}\big) = \max(q,0) + S\min(q,0)$$

Simulation setup: classical five-spot pattern on layers of SPE10



Layer 1, 400 days



Layer 85, 400 days

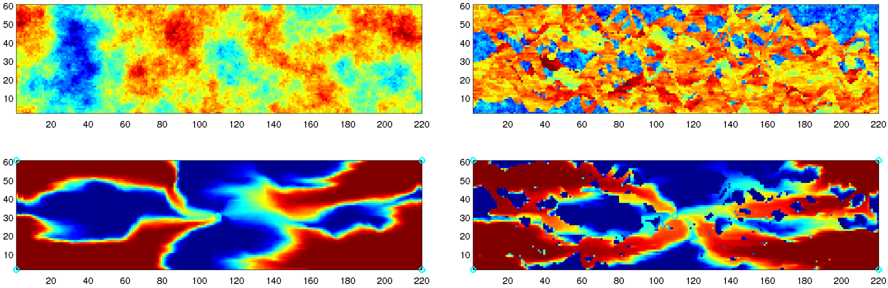
Comparison with upscaling methods

Model equations:

$$\nabla \cdot \vec{u} = q, \qquad \vec{u} = -\mathbf{K}\nabla p$$

$$S_t + \nabla \cdot \big(S\vec{u}\big) = \max(q,0) + S\min(q,0)$$

Simulation setup: classical five-spot pattern on layers of SPE10

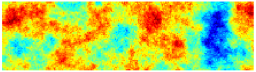


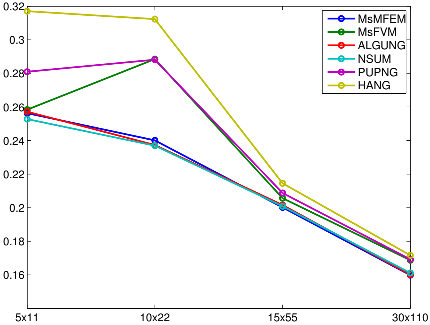
Layer 1, 1200 days

Layer 85, 1200 days

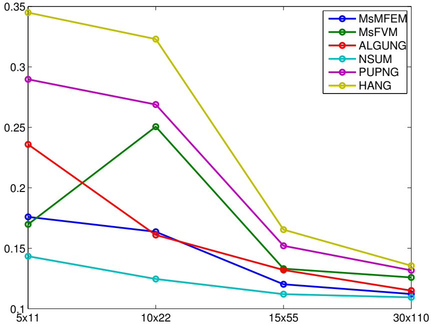
Example: layers of SPE10

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





Coarse-grid simulation

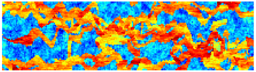


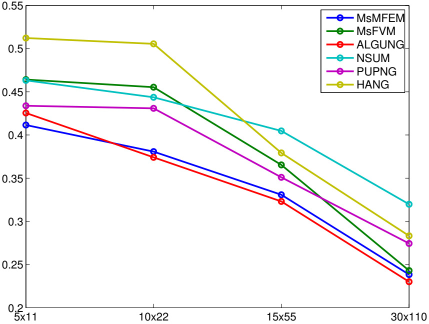
Fine-grid simulation

Saturation error: $e(S) = \frac{\|S - S^{\mathrm{ref}}\|_2}{\|S^{\mathrm{ref}}\|_2}$

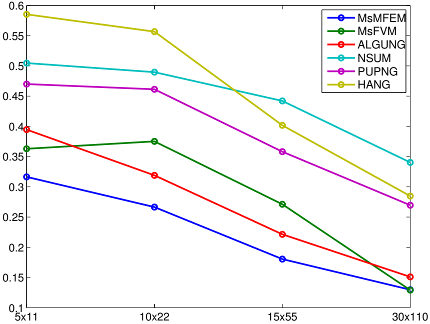
Example: layers of SPE10

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





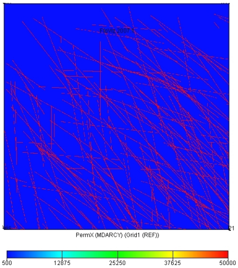
Coarse-grid simulation

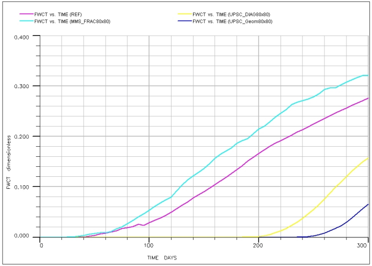


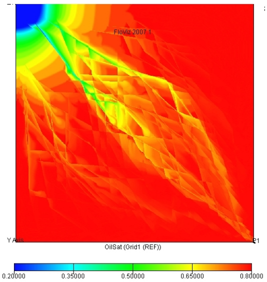
Fine-grid simulation

Saturation error: $e(S) = \frac{\|S - S^{\text{ref}}\|_2}{\|S^{\text{ref}}\|_2}$

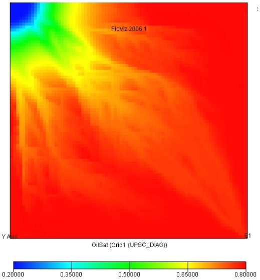
Example: a dense system of fracture corridors



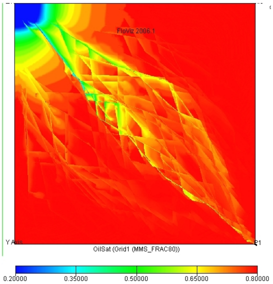




800 × 800



80 × 80 upscaled



80 × 80 multiscale

Computational complexity

Assume a uniform grid on a subset of $\mathbb{R}^d$:

- Grid model with $N = n_f * N_c$ cells:
  - N_c number of coarse blocks
  - n_f 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 for some methods)

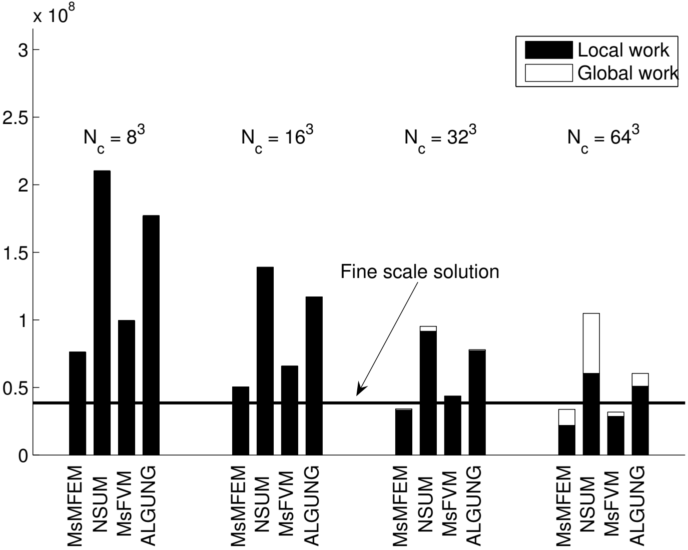
Direct solution

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

MsMFE

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

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



Comparison with algebraic multigrid, $\alpha = 1.2$

Multiphase flow: time-dependent problems

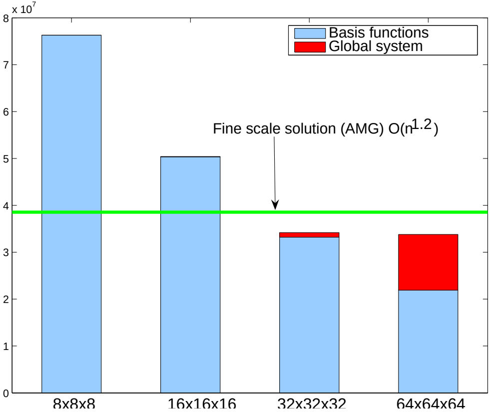
Direct solution may be more efficient, so why bother with multiscale?

In a typical simulation of multiphase flow:

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

Also:

- Possible to solve very large problems
- Easy parallelization



Example: 10^{th} SPE Comparative Solution Project

SPE 10, Model 2:

Fine grid: $60 \times 220 \times 85$

Coarse grid: $5 \times 11 \times 17$

2000 days production

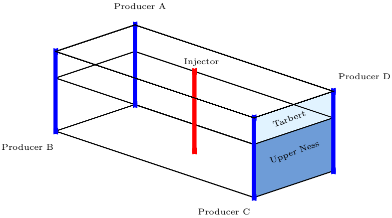
25 time steps

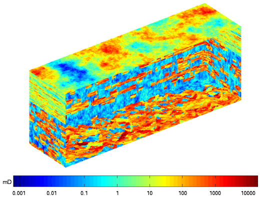
Streamline solver from 2005:

multiscale: 2 min and 20 sec
multigrid: 8 min and 36 sec

Fully unstructured Matlab/C code from 2010:

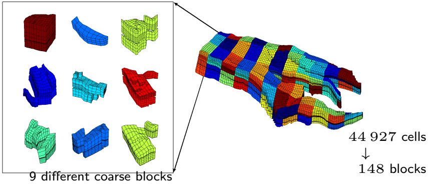
mimetic : 5–6 min



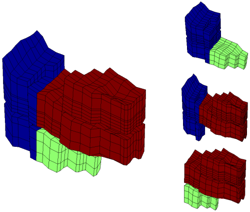


Workflow with automated upgridding in 3D

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

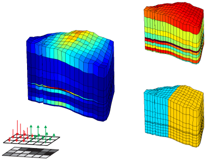


2) Detect all adjacent blocks



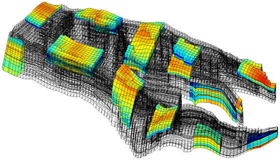
3) Compute basis functions

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



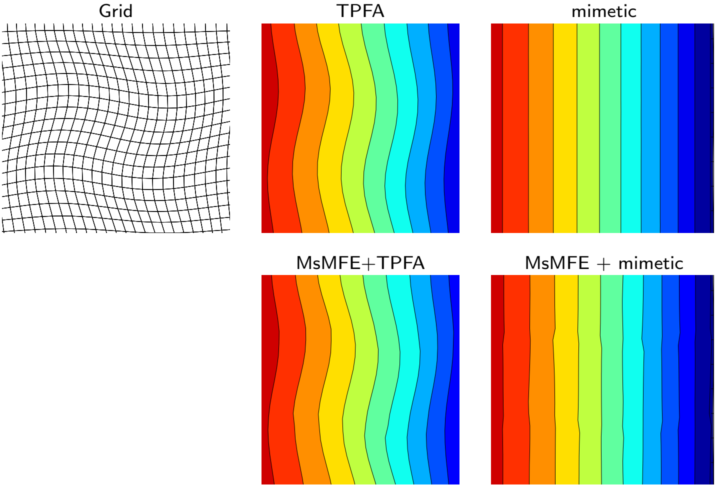
for all pairs of blocks

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



Multiscale method inherits properties of fine-scale solver

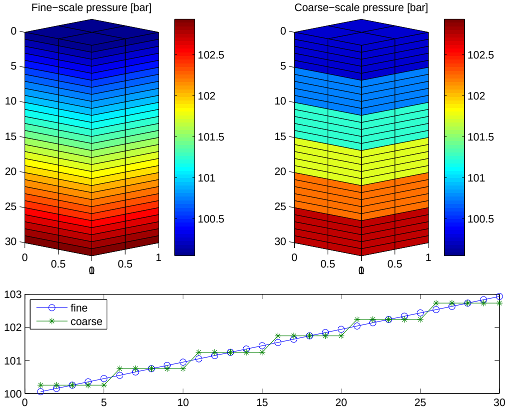
Single-phase flow, homogeneous **K**, linear pressure drop



More physics

The method so far:

- resolves viscous forces on fine-scale using elliptic basis functions
- resolves other physical forces like gravity, capillary pressure, compressibility, etc on the coarse scale



More physics

The method so far:

- resolves viscous forces on fine-scale using elliptic basis functions
- resolves other physical forces like gravity, capillary pressure, compressibility, etc on the coarse scale

Why is this so?

Think of the MsMFE method as a means for computing a homogeneous solution of an equation of the form

$$-\nabla \cdot \big(\lambda \mathbf{K} \nabla p\big) = q - h(x,p)$$

In a multiphase setting:

$$-\nabla \cdot \big(\lambda \mathbf{K} \nabla p\big) = q - \nabla \cdot \big(g\mathbf{K} \sum_{\alpha} \rho_\alpha \lambda_\alpha \nabla z\big)$$

Since λ and λ_α depend upon S, the balance of viscous and gravity forces will depend upon $S \longrightarrow$ basis functions would depend strongly upon S

Residual correction

To get a convergent method, we need to also account for variations that are not captured by the basis functions $\longrightarrow$ solve a residual equation

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^{\mathsf{T}} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi} \boldsymbol{u}_c + \tilde{\boldsymbol{u}} \\ -\boldsymbol{\mathcal{I}} \boldsymbol{p}_c - \boldsymbol{D}_\lambda \boldsymbol{\Phi} \boldsymbol{u}_c - \tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{q} \end{bmatrix}$$

Residual correction

To get a convergent method, we need to also account for variations that are not captured by the basis functions $\longrightarrow$ solve a residual equation

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi} \boldsymbol{u}_c + \tilde{\boldsymbol{u}} \\ -\boldsymbol{\mathcal{I}} \boldsymbol{p}_c - \boldsymbol{D}_\lambda \boldsymbol{\Phi} \boldsymbol{u}_c - \tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{q} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{u}} \\ -\tilde{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} (\boldsymbol{C} \boldsymbol{D}_\lambda \boldsymbol{\Phi} - \boldsymbol{B} \boldsymbol{\Psi}) \boldsymbol{u}_c + \boldsymbol{C} \boldsymbol{\mathcal{I}} \boldsymbol{p}_c \\ \boldsymbol{q} - \boldsymbol{C}^\mathsf{T} \boldsymbol{\Psi} \boldsymbol{u}_c \end{bmatrix}$$

To solve this equation, we will typically use a (non)overlapping domain-decomposition method.

Compressible flow

Parabolic pressure equation

$$\vec{v} = -\lambda \mathbf{K}\big(\nabla p - \sum_j \rho_j f_j \vec{g}\big)$$

$$\nabla \cdot \vec{v} = q - c_t \frac{\partial p}{\partial t} + \Big(\sum_j c_j f_j \vec{v} + \alpha(p)\mathbf{K}\vec{g}\Big) \cdot \nabla p$$

Iterative mixed formulation:

$$\begin{bmatrix} \boldsymbol{B}(\boldsymbol{s}^n) & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{P}(\boldsymbol{s}^n, \boldsymbol{p}^{n+1}_{\nu+1}) \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^{n+1}_{\nu+1} \\ -\boldsymbol{p}^{n+1}_{\nu+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}(\boldsymbol{s}^n, \boldsymbol{p}^{n+1}_{\nu}) \\ \boldsymbol{g}(\boldsymbol{s}^n, \boldsymbol{p}^n, \boldsymbol{p}^{n+1}_{\nu}) \end{bmatrix}$$

n denotes time step and ν denotes iteration step

Iterative MsMFE for compressible flow

Compute elliptic basis functions, constructed with $w(x) \propto \phi(x)$

For t=0:Δt:T

1. Solve coarse-scale system iteratively until convergence

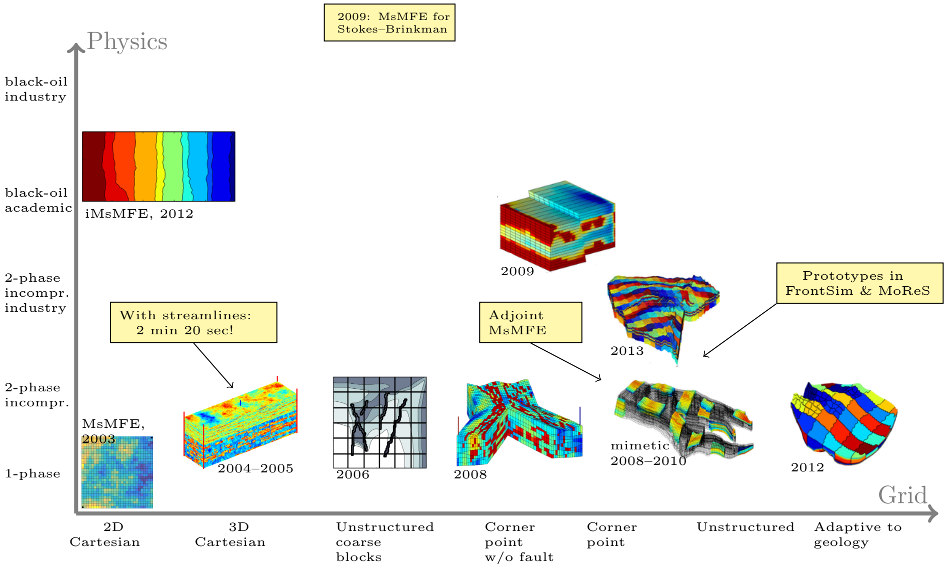
$$\begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{B} \boldsymbol{\Psi} & \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{C} \boldsymbol{\mathcal{I}} \\ \boldsymbol{\mathcal{I}}^\mathsf{T} (\boldsymbol{C}^\mathsf{T} \boldsymbol{\Psi} - \boldsymbol{P}_\nu \boldsymbol{D}_\lambda \boldsymbol{\Phi}) & \boldsymbol{\mathcal{I}}^\mathsf{T} \boldsymbol{P}_\nu \boldsymbol{\mathcal{I}} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_c^{\nu+1} \\ -\boldsymbol{p}_c^{\nu+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{f}_\nu \\ \boldsymbol{\mathcal{I}}^\mathsf{T} \boldsymbol{g}_\nu \end{bmatrix}$$

2. Compute residual equation by domain decomposition

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{C}^\mathsf{T} & \boldsymbol{P} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{u}}^{\nu+1} \\ -\hat{\boldsymbol{p}}^{\nu+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_c - \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{B} \boldsymbol{\Psi} \boldsymbol{u}_c + \boldsymbol{\Psi}^\mathsf{T} \boldsymbol{C} \boldsymbol{\mathcal{I}} \boldsymbol{p}_c \\ \boldsymbol{g}_c - \boldsymbol{\mathcal{I}}^\mathsf{T} (\boldsymbol{C}^\mathsf{T} \boldsymbol{\Psi} - \boldsymbol{P}_\nu \boldsymbol{D}_\lambda \boldsymbol{\Phi}) \boldsymbol{u}_c + \boldsymbol{\mathcal{I}}^\mathsf{T} \boldsymbol{P}_\nu \boldsymbol{\mathcal{I}} \boldsymbol{p}_c \end{bmatrix}$$

3. If fine-scale residual is not below tolerance, go to Step 1

Development towards industry deployment



Outline

Multiscale finite-volume methods

Extensive research over the past 15 years – more than 60 papers by Jenny, Lee, Tchelepi, Lunati, Hajibeygi, and others:

- correction functions to handle non-elliptic features
- extension to compressible flow
- adaptivity in updating of basis functions
- iterative formulation with smoothers (Jacobi, GMRES, . . .)
- algebraic formulation
- fracture models (embedded/hierarchical, etc)

$\vdots$

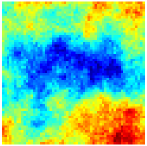
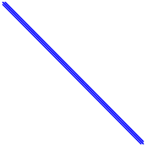
Strong focus on the ability to converge to a fine-scale solution has gradually made MsFV similar to multigrid methods

Multiscale finite-volume methods: the key concept

Illustration: cell-centered TPFA

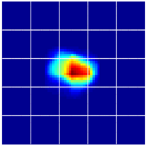
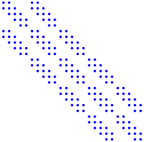
$$-\nabla \cdot \mathbf{K}\nabla p = q$$
$$A\boldsymbol{x} = \boldsymbol{q}$$

Initial fine-scale system, incorporating all details of geological model

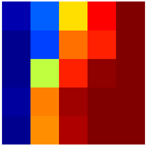
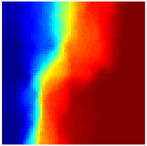
$$\boldsymbol{x} = P\boldsymbol{x}_c$$
$$P = \text{basis}(A)$$
$$A_{ms} = RAP$$
$$\boldsymbol{q}_c = R\boldsymbol{q}$$

Multiscale expansion:
generate basis functions,
restrict fine-scale system
and right-hand side

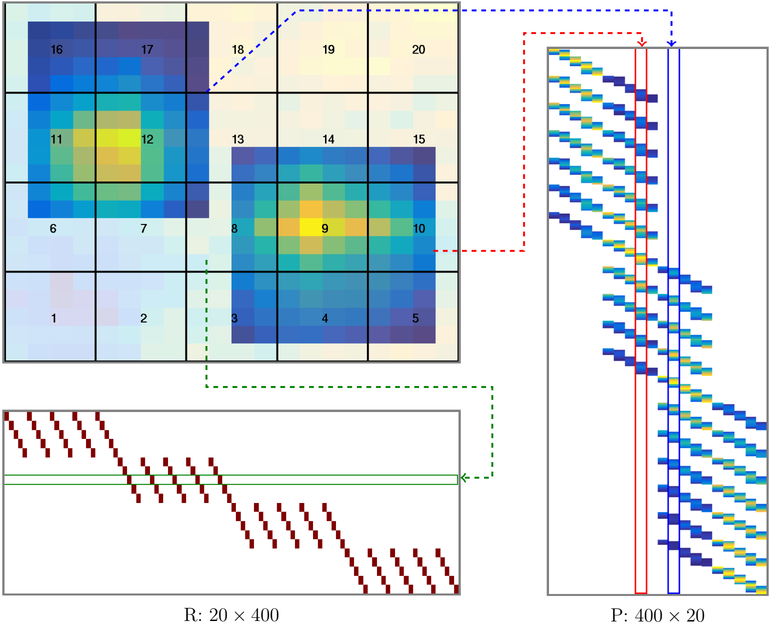



$$\boldsymbol{x}_c = A_{ms}^{-1}\boldsymbol{q}_c$$
$$\boldsymbol{x} \approx P\boldsymbol{x}_c$$

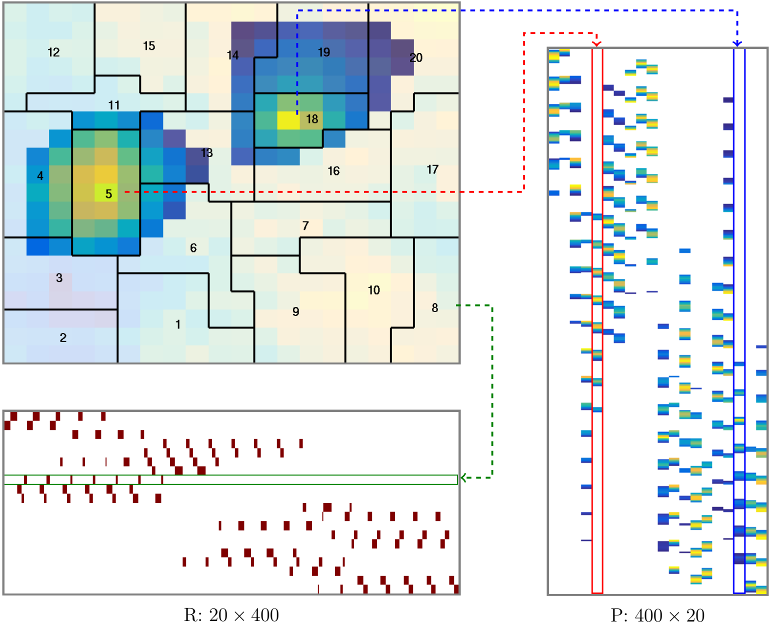
Solve **reduced** system,
prolongate to obtain
approximate pressure

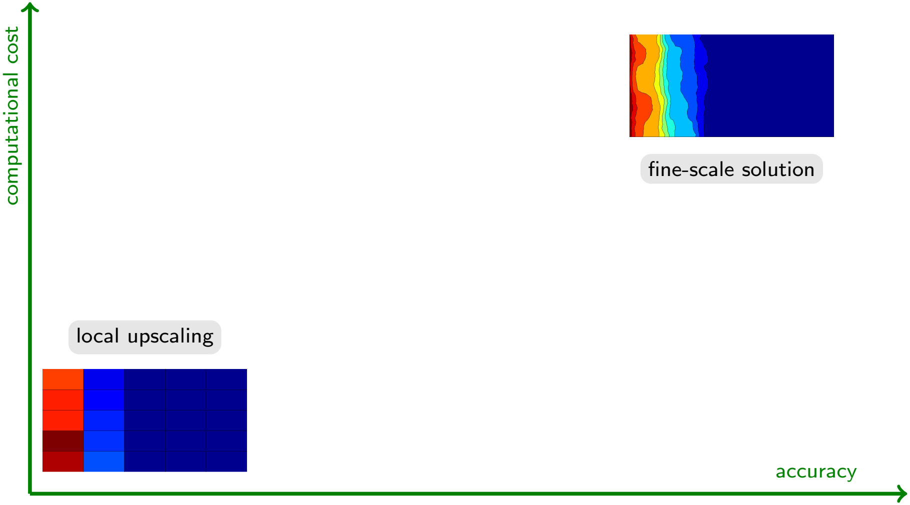
Prolongation and restriction operators



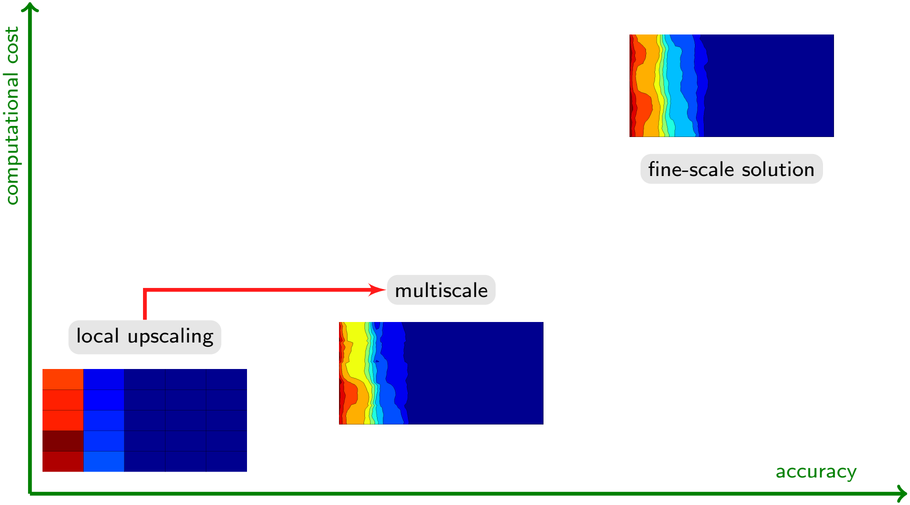
Prolongation and restriction operators



Qualitatively correct → small residual



Qualitatively correct → small residual



Qualitatively correct → small residual

Residual iteration:

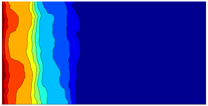
$$p^* = p^\nu + S(q - Ap^\nu)$$

$$p^{\nu+1} = p^* + A_{ms}^{-1}(q - Ap^*)$$

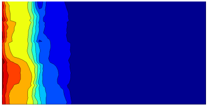
S is some inexpensive smoother, e.g., ILU(0)

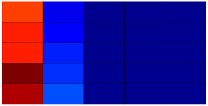
Hajibeygi, Jenny, Tchelepi, Wang, . . . (2008–2015)

computational cost



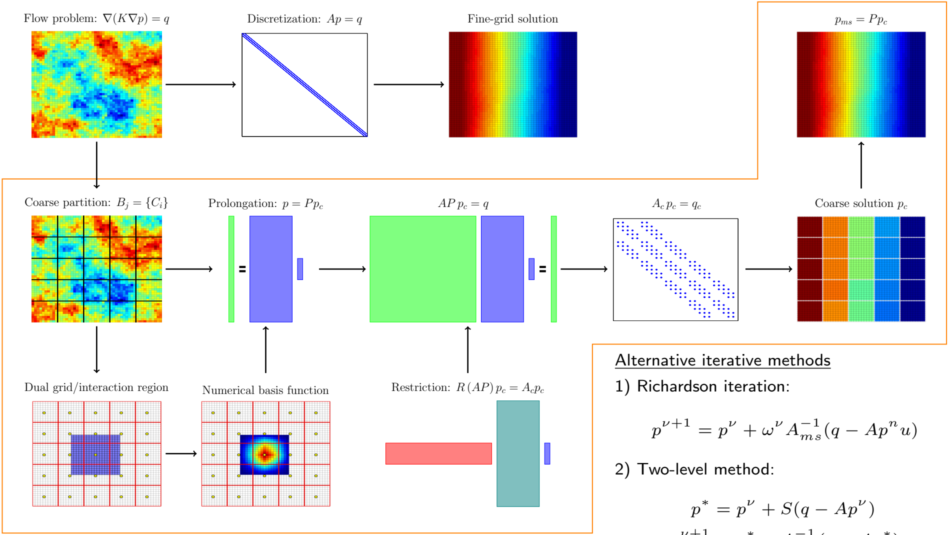
extra iterations





accuracy

Iterative multiscale framework



Alternative iterative methods

1) Richardson iteration:

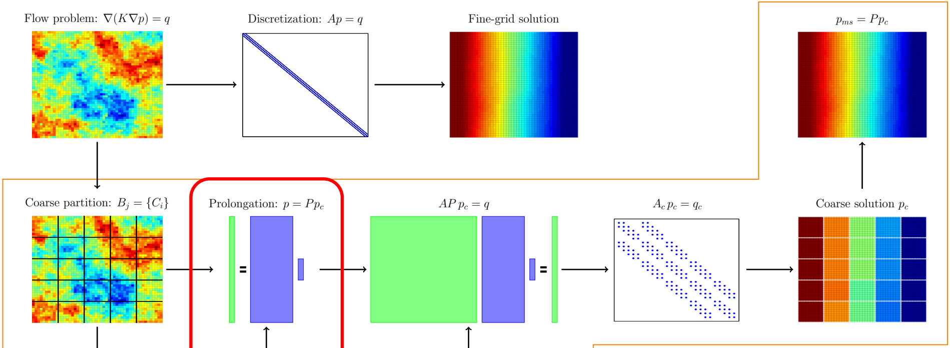
$$p^{\nu+1} = p^{\nu} + \omega^{\nu} A_{ms}^{-1}(q - Ap^{n}u)$$

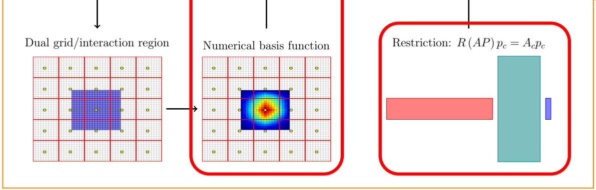
2) Two-level method:

$$p^{*} = p^{\nu} + S(q - Ap^{\nu})$$
$$p^{\nu+1} = p^{*} + A_{ms}^{-1}(q - Ap^{*})$$

3) A_{ms}^{-1}: preconditioner for GMRES

Iterative multiscale framework





These can be modified

Alternative iterative methods

1) Richardson iteration:

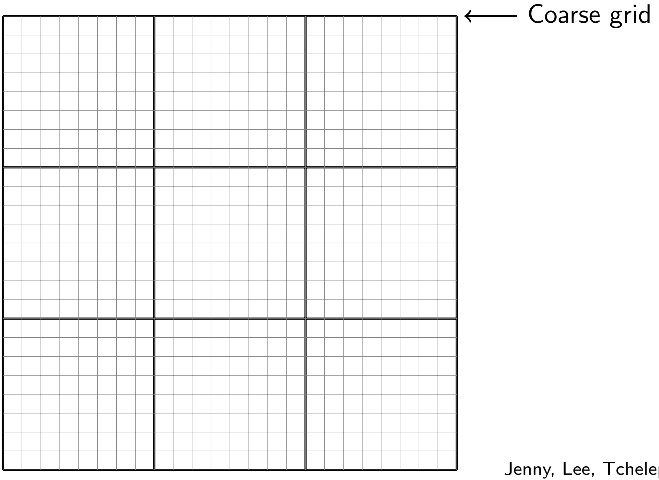
$$p^{\nu+1} = p^{\nu} + \omega^{\nu} A_{ms}^{-1}(q - Ap^{n}u)$$

2) Two-level method:

$$p^{*} = p^{\nu} + S(q - Ap^{\nu})$$
$$p^{\nu+1} = p^{*} + A_{ms}^{-1}(q - Ap^{*})$$

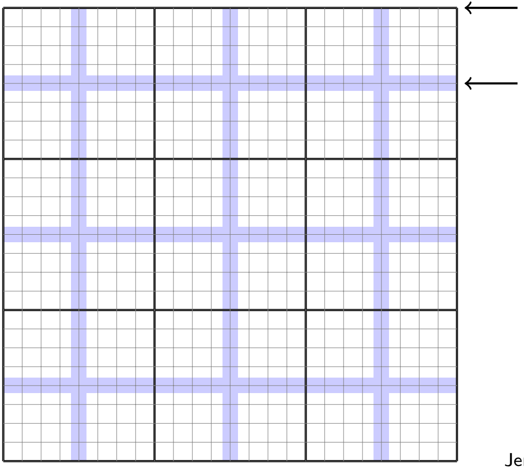
3) A_{ms}^{-1}: preconditioner for GMRES

The MsFV prolongation operator



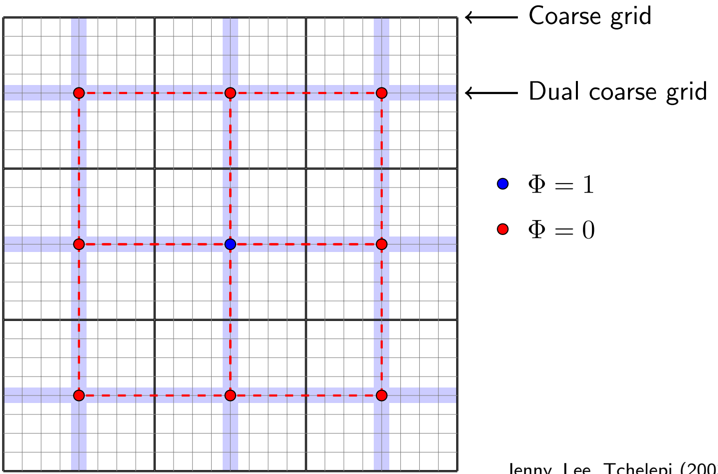
Jenny, Lee, Tchelepi (2003)

The MsFV prolongation operator



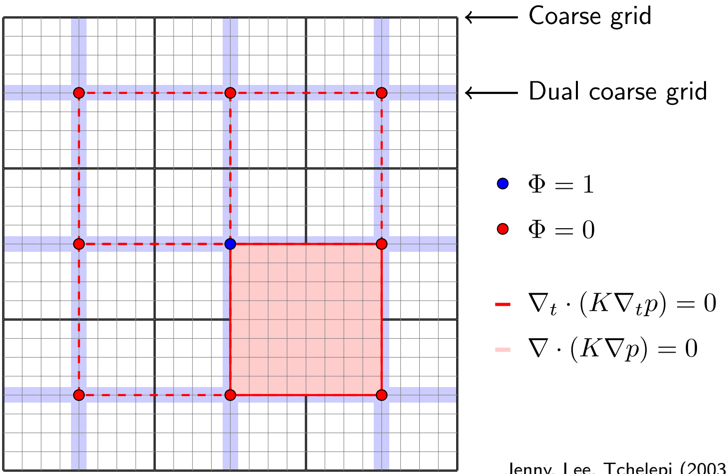
Jenny, Lee, Tchelepi (2003)

The MsFV prolongation operator



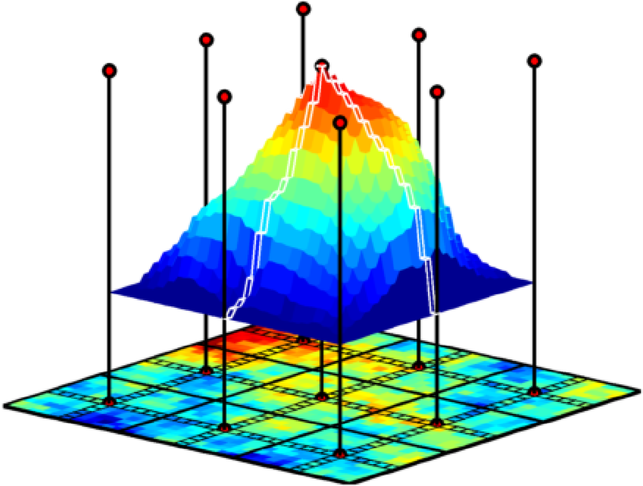
Jenny, Lee, Tchelepi (2003)

The MsFV prolongation operator

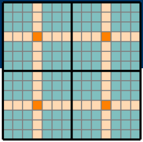


Jenny, Lee, Tchelepi (2003)

The MsFV prolongation operator



The MsFV method: operator formulation

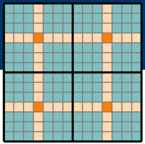


Permute system based on dual-grid ordering

$$Q\mathbf{p}_h = \mathbf{p} = \begin{bmatrix} \mathbf{p}_i \\ \mathbf{p}_f \\ \mathbf{p}_e \\ \mathbf{p}_n \end{bmatrix}, \qquad QA_hQ^T = A = \begin{bmatrix} A_{ii} & A_{if} & 0 & 0 \\ A_{fi} & A_{ff} & A_{fe} & 0 \\ 0 & A_{ef} & A_{ee} & A_{en} \\ 0 & 0 & A_{ne} & A_{nn} \end{bmatrix}$$

Matrix block A_{kl}: influence from cells l to mass balance of cells k

The MsFV method: operator formulation



Permute system based on dual-grid ordering

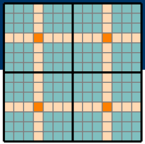
$$Q\mathbf{p}_h = \mathbf{p} = \begin{bmatrix} \mathbf{p}_i \\ \mathbf{p}_f \\ \mathbf{p}_e \\ \mathbf{p}_n \end{bmatrix}, \qquad QA_hQ^T = A = \begin{bmatrix} A_{ii} & A_{if} & 0 & 0 \\ A_{fi} & A_{ff} & A_{fe} & 0 \\ 0 & A_{ef} & A_{ee} & A_{en} \\ 0 & 0 & A_{ne} & A_{nn} \end{bmatrix}$$

Matrix block A_{kl}: influence from cells l to mass balance of cells k

Remove lower-diagonal blocks and ensure mass balance is still enforced,

$$(M_{kk})_{rr} = (A_{kk})_{rr} + \sum_s (A_{kl})_{rs} \quad \longrightarrow \quad \begin{bmatrix} A_{ii} & A_{if} & 0 & 0 \\ 0 & M_{ff} & A_{fe} & 0 \\ 0 & 0 & M_{ee} & A_{en} \\ 0 & 0 & 0 & M_{nn} \end{bmatrix}$$

The MsFV method: operator formulation



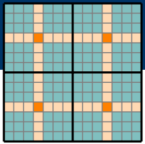
Assume nodal pressure $\mathbf{p}_n$ to be known. This gives a solution

$$\mathbf{p} = P\mathbf{p}_n$$

where B are the basis functions

$$P = \begin{bmatrix} A_{ii}^{-1} A_{if} M_{ff}^{-1} A_{fe} M_{ee}^{-1} A_{en} \\ M_{ff}^{-1} A_{fe} M_{ee}^{-1} A_{en} \\ M_{ee}^{-1} A_{en} \\ I \end{bmatrix}$$

The MsFV method: operator formulation



Assume nodal pressure $\mathbf{p}_n$ to be known. This gives a solution

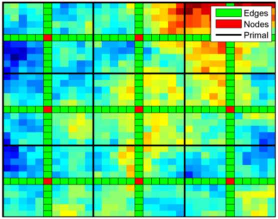
$$\mathbf{p} = P\mathbf{p}_n$$

where B are the basis functions

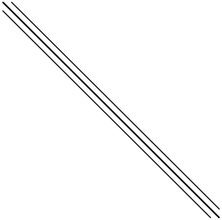
$$P = \begin{bmatrix} A_{ii}^{-1} A_{if} M_{ff}^{-1} A_{fe} M_{ee}^{-1} A_{en} \\ M_{ff}^{-1} A_{fe} M_{ee}^{-1} A_{en} \\ M_{ee}^{-1} A_{en} \\ I \end{bmatrix}$$

Pressure in nodes $\mathbf{p}_n$ found by enforcing mass balance on the coarse grid

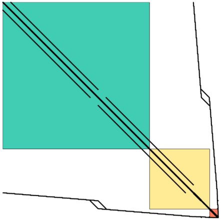
The MsFV method: operator formulation



Categorization of cells



System matrix A



$$\begin{bmatrix} A_{ii} & A_{ei} & 0 \\ A_{ie} & A_{ee} & A_{ne} \\ 0 & A_{en} & A_{nn} \end{bmatrix}$$

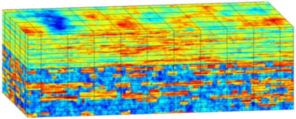
$$\begin{bmatrix} A_{ii} & A_{ei} & 0 \\ 0 & M_{ee} & A_{ne} \\ 0 & 0 & M_{nn} \end{bmatrix} \quad \rightarrow \quad \widetilde{P} = \begin{bmatrix} A_{ii}^{-1} A_{ei} M_{ee}^{-1} A_{ne} \\ M_{ee}^{-1} A_{ne} \\ I \end{bmatrix}$$

$$(M_{ll})_{rr} = (A_{ll})_{rr} + \textstyle\sum_s (A_{kl})_{rs}$$

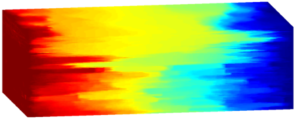
The MsFV method: prominent shortcomings

Not working as well as you may get the impression of:

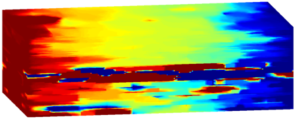
- Only applicable to relatively simple grids: Cartesian, simplexes, 'conceptual' fault models
- Localization procedure not robust $\rightarrow$ unstable multipoint coarse-scale stencil gives oscillatory solutions
- Test cases reported in literature use seemingly complex flow physics
- Use of iterations over-emphasized!



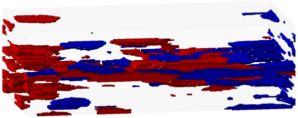
SPE 10: $\log(K)$



Reference solution



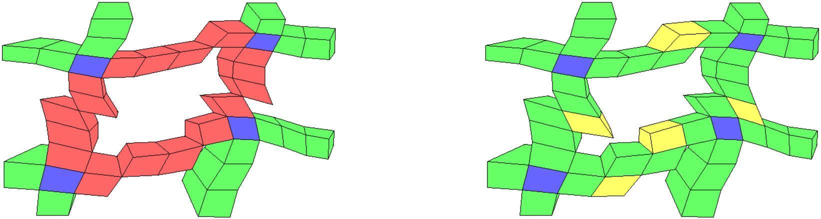
MsFV solution



MsFV $p \notin [0, 1]$

The MsFV method: wirebasket ordering

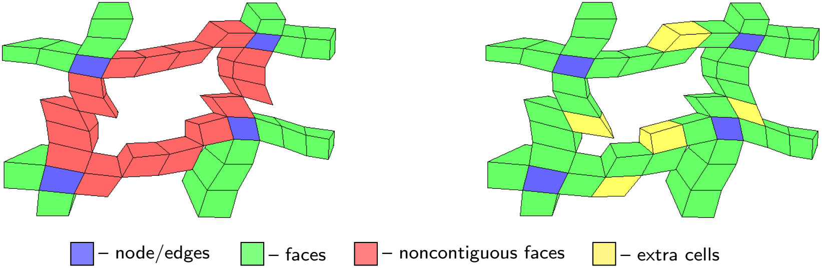
Requirement of consistent dual-primal partition makes coarsening difficult



– node/edges – faces – noncontiguous faces – extra cells

The MsFV method: wirebasket ordering

Requirement of consistent dual-primal partition makes coarsening difficult

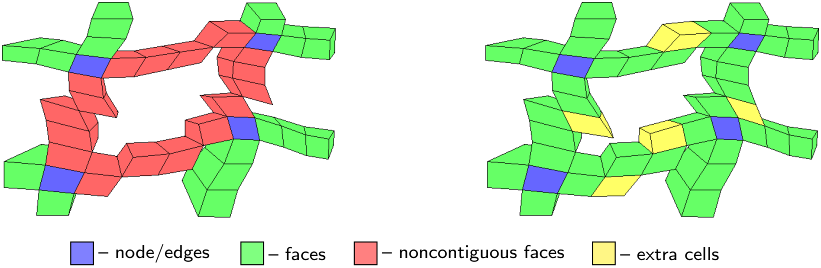


Algorithms for generating partitions on general grids:

- automated on rectilinear, curvilinear, triangular, and Voronoi grids
- semi-automated on (simple) stratigraphic grids non-matching faces
- no known algorithm for full industry-standard complexity

The MsFV method: wirebasket ordering

Requirement of consistent dual-primal partition makes coarsening difficult



Automated algorithms struggle with:

- dual block centers in low-permeable regions
- dual edges crossing strong permeability contrasts (twice)
- large number of cells categorized as edges

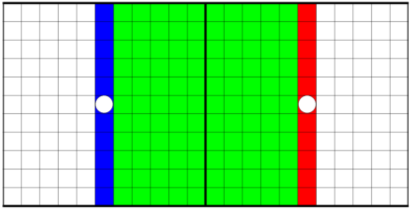
$\longrightarrow$ nonmonotonicity, poor decoupling, failure to reproduce linear flow

MsTPFA: improve monotonicity properties

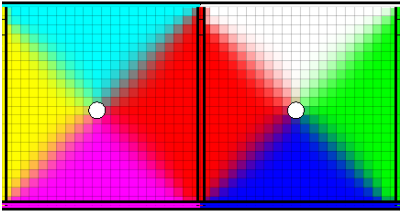
Idea: make coarse-scale stencil be of two-point type

Approach:

- Move degrees-of-freedom to block faces (as in MsMFE)
- Compute flow solutions as in transmissibility upscaling
- Use additional partition-of-unity to define basis functions

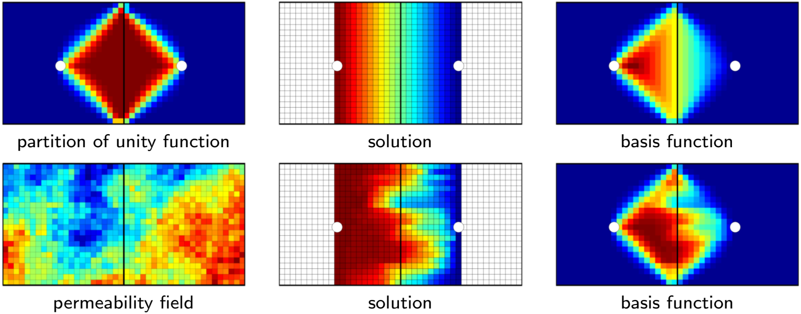


Local flow problems



Partition of unity

MsTPFA: improve monotonicity properties



- Much more stable than MsFV, although not 100% perfect
- Applicable to stratigraphic and fully unstructured grids
- Can be used both as preconditioner and approximate solver
- Slightly less accurate than MsFV on simple rectangular grids
- Can likely be generalized to other MPFA-type methods

Rethinking prolongation operator

What are our requirments on the prolongation operator?

- Partition of unity to represent constant fields

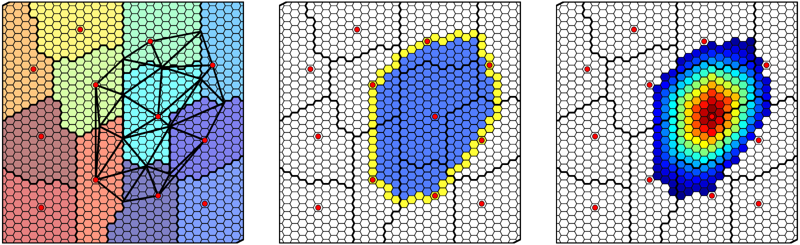
$$\sum_j P_{ij} = 1 \longrightarrow \text{Exact interpolation of constant modes}$$

- Algebraically smooth: minimize $\|AP\|_1$ implies that $APp_c \approx Ap$ locally
- Localication: coarse system $A_c = RAP$ becomes denser as the support of basis functions grows

Prolongation operator: MsRSB

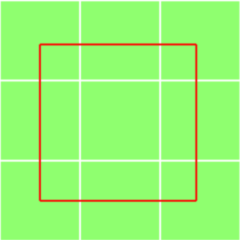
Basis functions require **a coarse grid** and **a support region**

- Region constructed using triangulation of nodal coarse neighbors, resulting in a multipoint stencil on the coarse scale
- Avoid solving reduced flow problem along perimeter
- Main point: simple to implement in 3D for general polyhedral grids



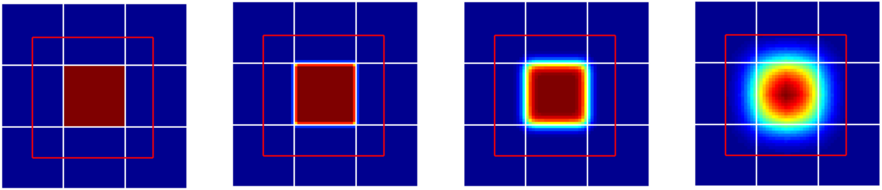
Coarse grid + triangulation   support region   basis function

MsRSB: restricted smoothing



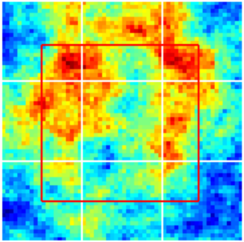
Ideally, operators are both *smooth* and *local*

1. Start with constant functions on primal grid
2. Apply Jacobi-like iterations as in algebraic multigrid methods, $P^{n+1} = P^n - \omega D^{-1}(AP^n)$
3. Restrict each function to its support region
4. Repeat Steps 2 and 3 until convergence



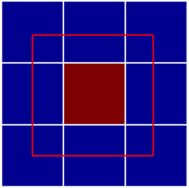
Initial constant basis | After one pass | After 10 passes | Converged ($n \approx 100$)

MsRSB: restricted smoothing

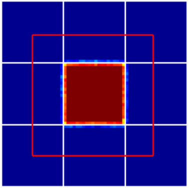


Ideally, operators are both *smooth* and *local*

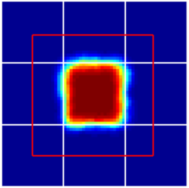
1. Start with constant functions on primal grid
2. Apply Jacobi-like iterations as in algebraic multigrid methods, $P^{n+1} = P^n - \omega D^{-1}(AP^n)$
3. Restrict each function to its support region
4. Repeat Steps 2 and 3 until convergence



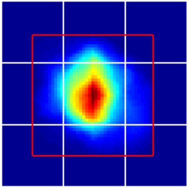
Initial constant basis



After one pass



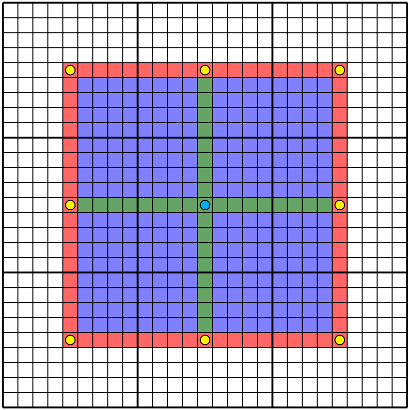
After 10 passes



Converged ($n \approx 100$)

MsRSB: restricted smoothing

Coarse grid: 3×3 partition



Set P_j to one inside block j

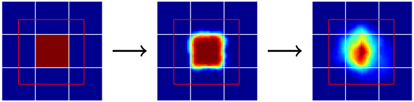
Jacobi increment: $d_j = -\omega D^{-1} A P_j^n$

Localize update:

$$\hat{d}_{ij} = \begin{cases} \frac{d_{ij} - P_{ij}^n \sum_k d_{ik}}{1 + \sum_k d_{ik}} \\ d_{ij} \\ 0 \end{cases}$$

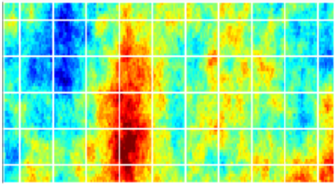
Apply increment: $P_{ij}^{n+1} = P_{ij}^n + \hat{d}_{ij}$

Indices: i=cell, j= ● , k= ●



Outline

Example: validation on SPE10 layers



Permeability

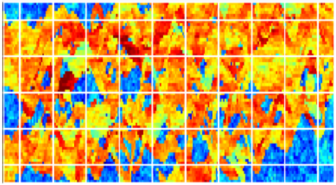
Reference solution

MsRSB

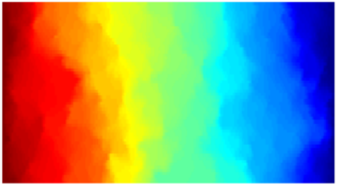
MsFV

Error	Grid	p (L^2)	p (L^∞)	v (L^2)	v (L^∞)
MsFV	6×11	0.0313	0.0910	0.1138	0.4151
MsRSB	6×11	0.0204	0.0766	0.0880	0.4071

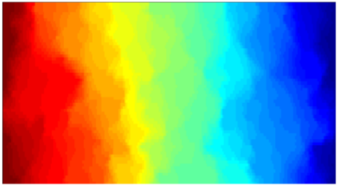
Example: validation on SPE10 layers



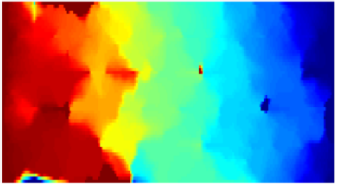
Permeability



Reference solution



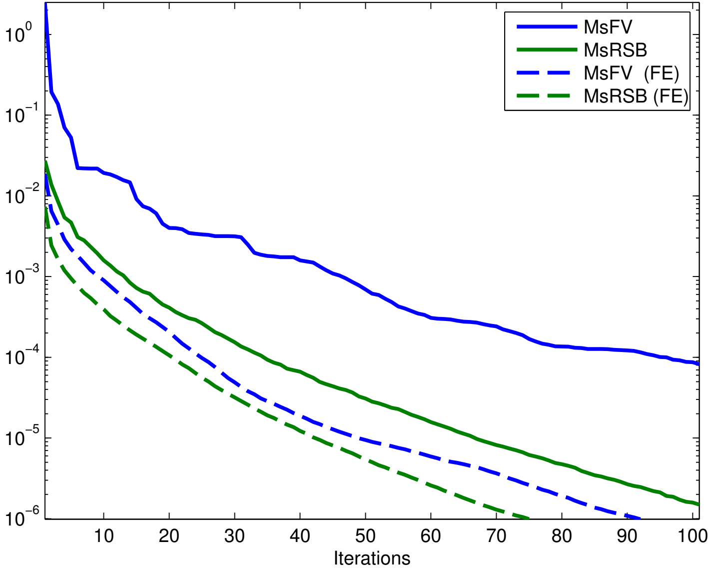
MsRSB



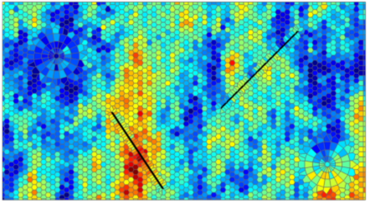
MsFV

Error	Grid	p (L^2)	p (L^∞)	v (L^2)	v (L^∞)
MsFV	6×11	0.2299	2.0725	0.4913	0.7124
MsRSB	6×11	0.0232	0.0801	0.1658	0.3240

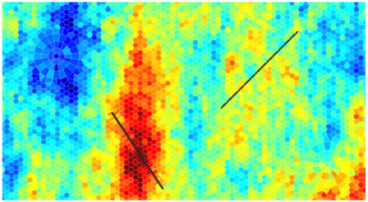
Example: GMRES-MS-ILU(0) for full SPE10



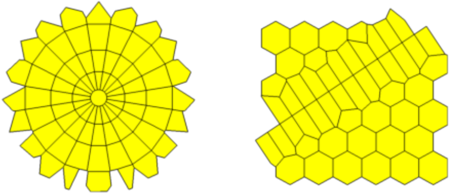
Example: unstructured PEBI grid



Porosity and grid



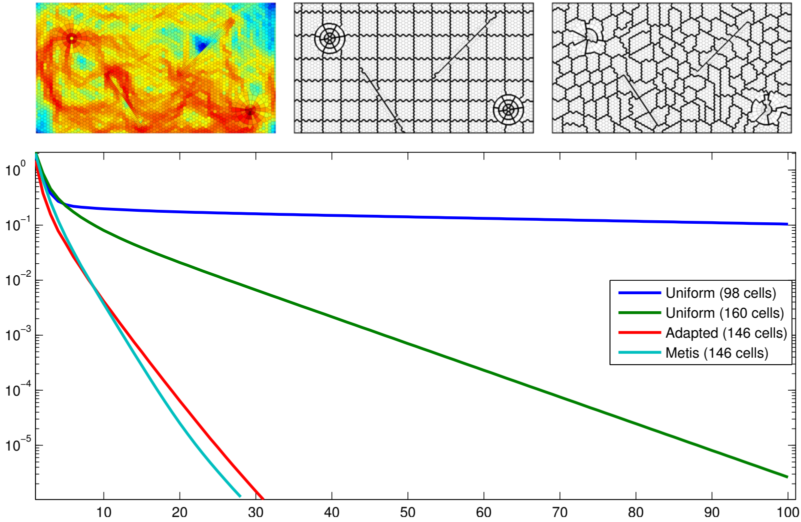
Permability from SPE 10, Layer 35



Detailed view of refinement

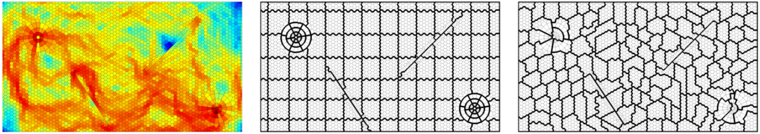
- Unstructured grid designed to minimize grid orientation effects
- Two embedded radial grids near wells
- Fine grid adapts to faults
- The faults are sealed, i.e. allow no fluid flow through

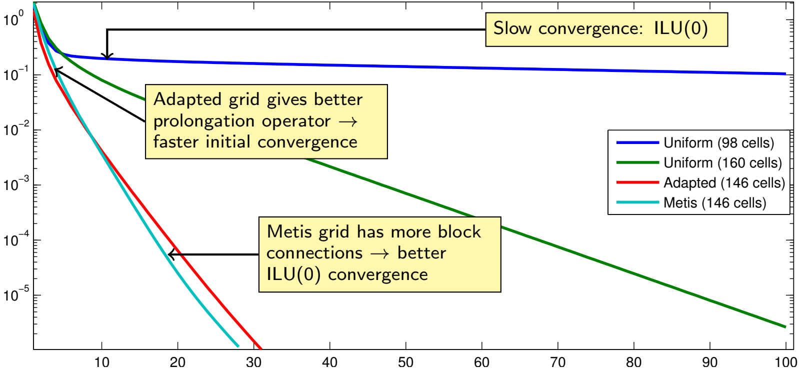
Example: unstructured PEBI grid



Two-step preconditioner, ILU(0) as 2nd stage, Richardson iterations

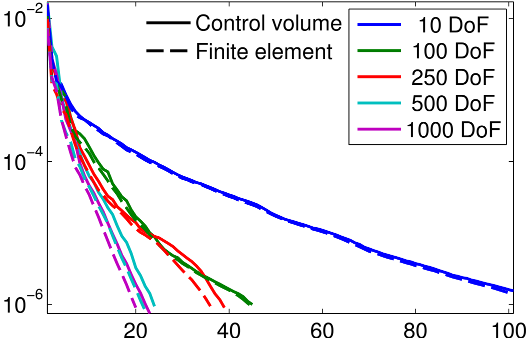
Example: unstructured PEBI grid

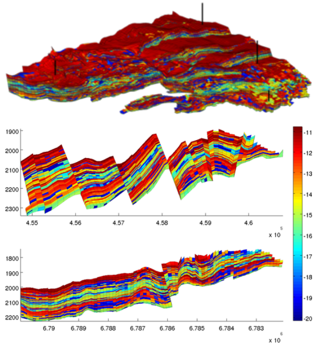




Two-step preconditioner, ILU(0) as 2nd stage, Richardson iterations

Example: Gullfaks field

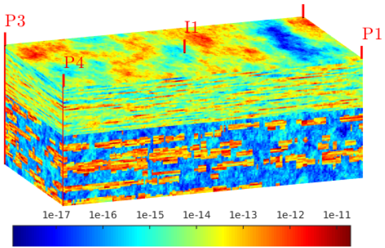


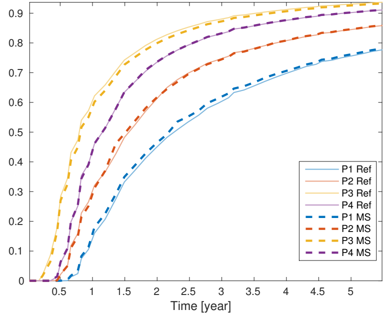


- Early field model of a giant reservoir from the Norwegian North Sea
- 216 000 cells with a large number of faults and eroded layers
- Very challenging anisotropic permeability and grid
- Model includes cells with nearly 40 faces
- Contrived well pattern: four vertical wells force flow through the whole model

Example: trade accuracy for compuational efficiency

SPE 10 with two strongly compressible fluids





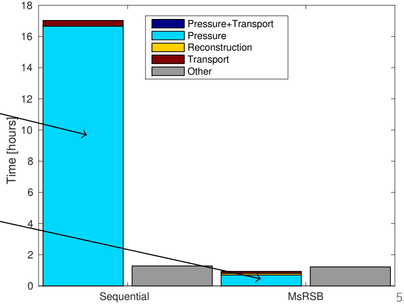
Iterated fine-scale solver:

- 0.001 pressure increment tolerance
- 10^{-6} tolerance for algebraic multigrid

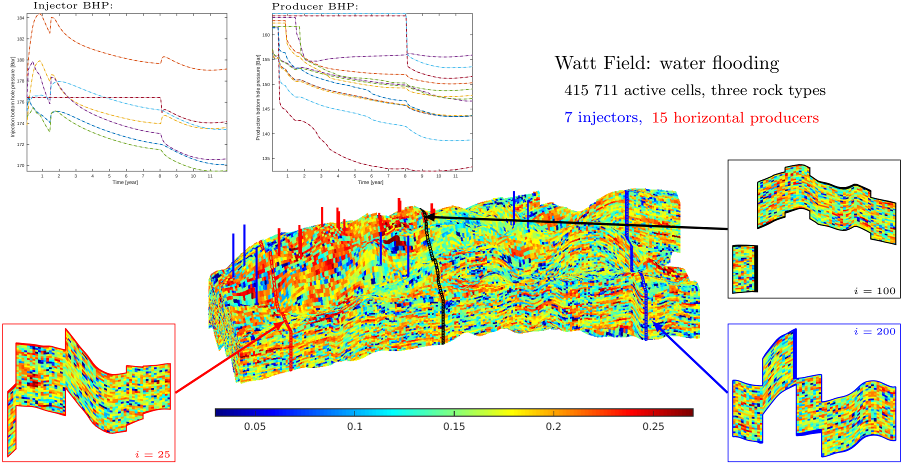
Iterated multiscale solver:

- 0.005 pressure increment tolerance
- 10^{-2} tolerance for MsRSB solver

Approximate MsRSB solver is ten times faster than baseline sequential



Example: realistic waterflooding

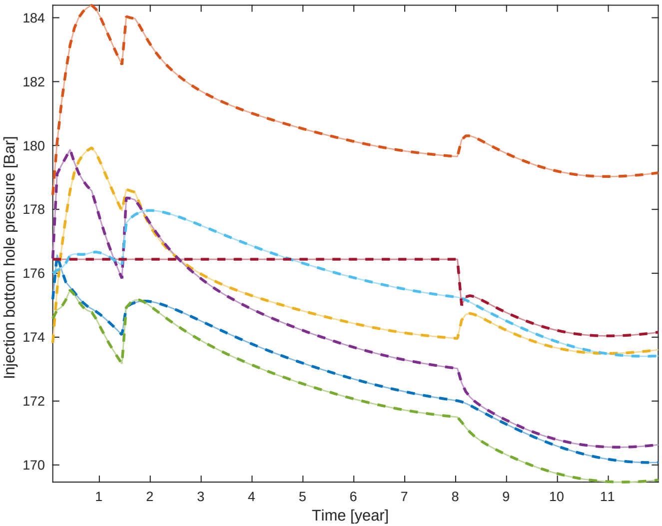


Watt Field: water flooding

415 711 active cells, three rock types

7 injectors, 15 horizontal producers

Example: realistic waterflooding



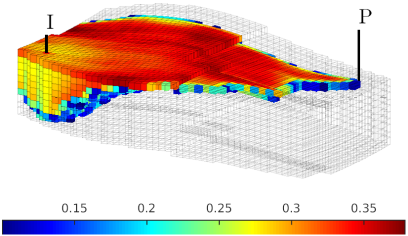
Thin solid: fine-scale solution
Thick dashed: multiscale solution

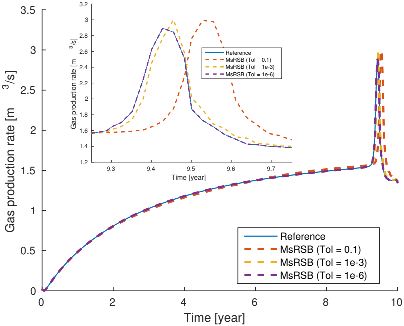
Multiscale: 800 blocks, tolerance 0.05
Solver speedup: 9×

Example: 3-phase flow

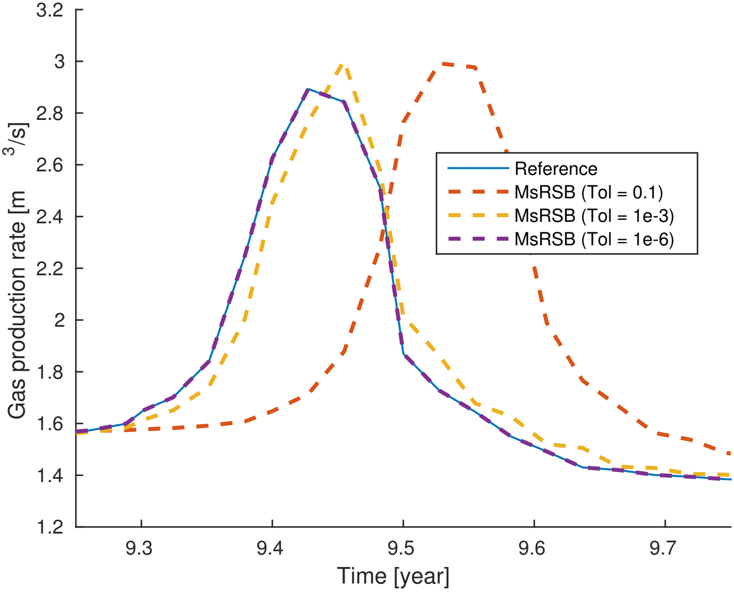
- Synthetic model with fluid behavior based on SPE1 benchmark
- Gas is injected at constant rate into an undersaturated reservoir
- Producer at fixed bottom hole pressure
- Highly sensitive to pressure approximation

Gas saturation at breakthrough



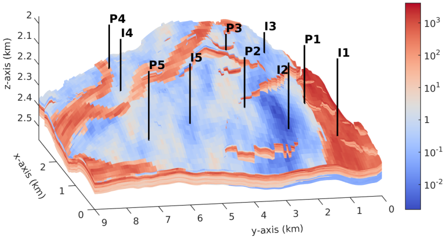


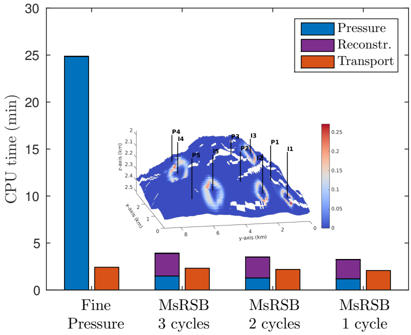
Example: 3-phase flow



Example: water-based EOR

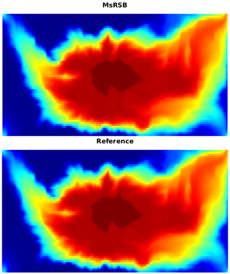
- Full Eclipse 100 polymer model with adsorption, Todd–Longstaff mixing, inaccessible pore volume, and permeability reduction
- Polymer concentration changes water viscosity to achieve better sweep
- Viscosity of water-polymer mixture depends on velocity (shear thinning)
- Non-Newtonian fluid rheology makes the pressure equation highly nonlinear

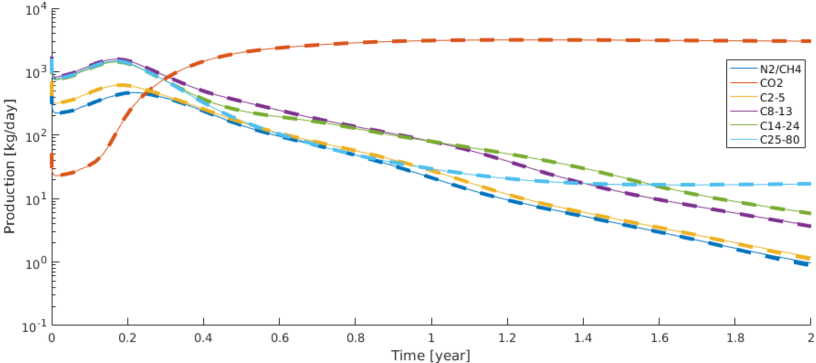




Example: compositional flow

- Carefully designed, sequentially-implicit method
- Challenging six-component fluid model from Mallison et al. (SPE 79691)
- Peng–Robison equation of state
- Heterogeneity sampled from the SPE 10 model





Room for improvements

There are still issues that can be improved:

- Slow convergence in certain cases with strong contrasts and long correlation lengths
- Desire to adapt coarse grid to geological features
- Improved resolution of wells
- More efficient reconstruction of conservative fluxes

Previous work:

- generalized multiscale element methods (Efendiev et al)
- hybrid finite-volume/Galerkin method (Cortinovis and Jenny)

New idea: multiple multiscale operators

Assume N prolongation operators $P^1, \ldots, P^N$ that may come from different coarse grids and support regions, or different multiscale methods (MsRSB, MsFV,...)

Likewise, there are N restriction operators $R^1, \ldots, R^N$

New idea: multiple multiscale operators

Assume N prolongation operators $P^1, \dots, P^N$ that may come from different coarse grids and support regions, or different multiscale methods (MsRSB, MsFV,...)

Likewise, there are N restriction operators $R^1, \dots, R^N$

Multiplicative multistep method:

$$p^* = p^{k+(\ell-1)/N} + S(q - Ap^{k+(\ell-1)/N})$$

$$p^{k+\ell/N} = p^* + P^\ell(\underbrace{R^\ell A P^\ell}_{A^\ell_{ms}})^{-1} R^\ell (q - Ap^*),$$

Example setup: P^1 is *general* and covers domain evenly, whereas $P^2, \dots, P^N$ are *feature specific*

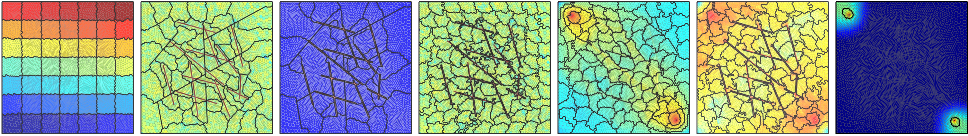
Minimal assumptions on operators

Three requirements on pairs of prolongation/restriction operators:

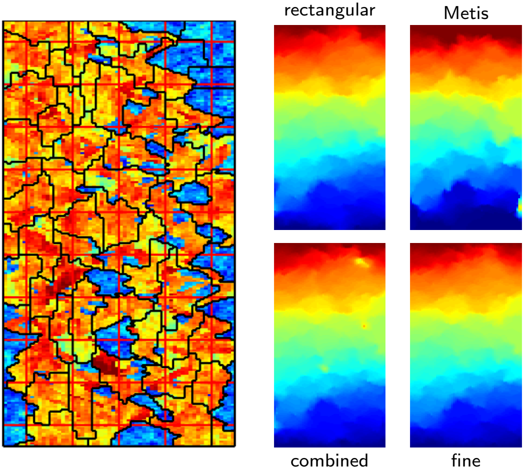
1. P^ℓ and R^ℓ are constructed from a non-overlapping partition of the fine grid. Each column j in $\boldsymbol{P}^\ell$ is called a *basis function* and is associated with a coarse grid block B_j^ℓ

2. The support S_j^ℓ of each basis function is compact and contains B_j^ℓ

3. The columns of P^ℓ form a partition of unity, i.e., each row in P^ℓ has unit row sum



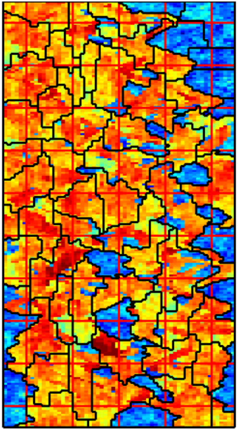
Numerical example: SPE10

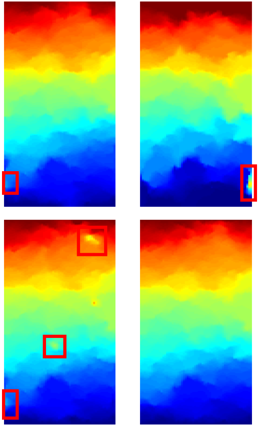


Layer 85: pressure drop from north to south end, linear relperms, unit viscosity

Partition	L^2	L^∞
Rectangular	0.0307	0.1782
Metis	0.0791	0.5506
Combined	0.0293	0.2929

Numerical example: SPE10



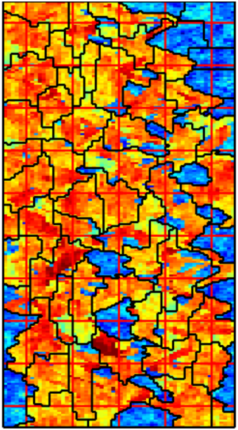


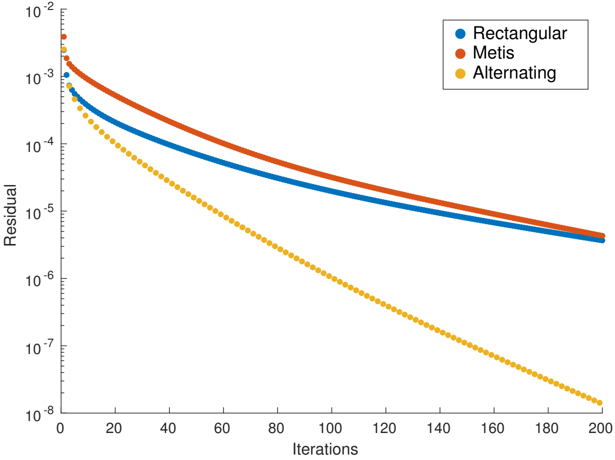
Layer 85: pressure drop from north to south end, linear relperms, unit viscosity

Partition	L^2	L^∞
Rectangular	0.0307	0.1782
Metis	0.0791	0.5506
Combined	0.0293	0.2929

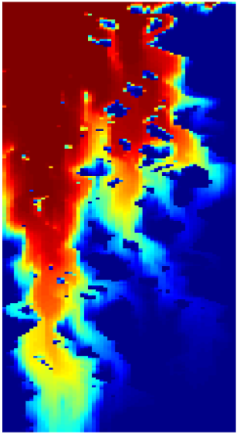
Local nonmonotonicity

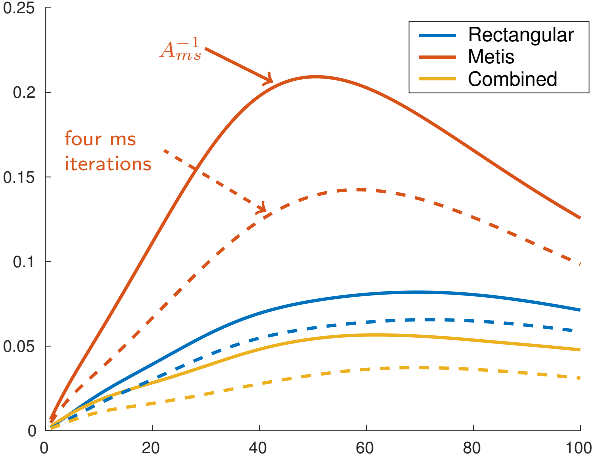
Numerical example: SPE10



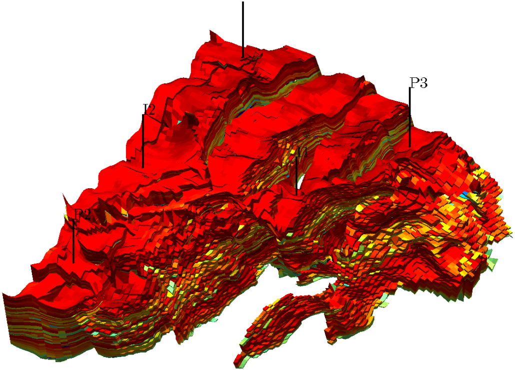


Numerical example: SPE10

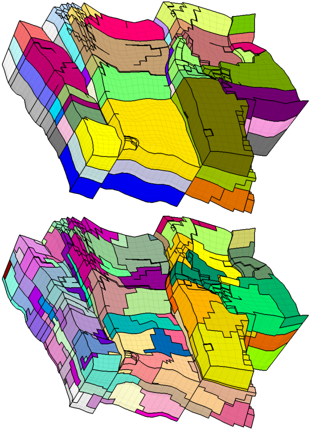




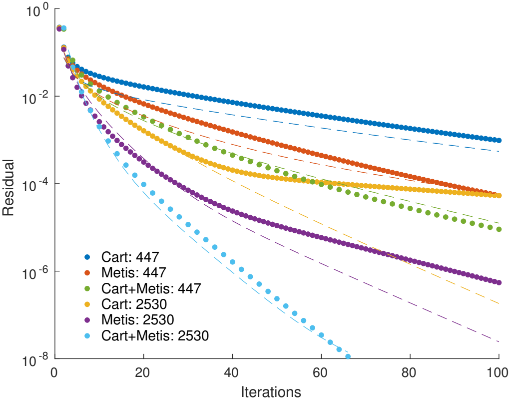
Numerical example: Gullfaks

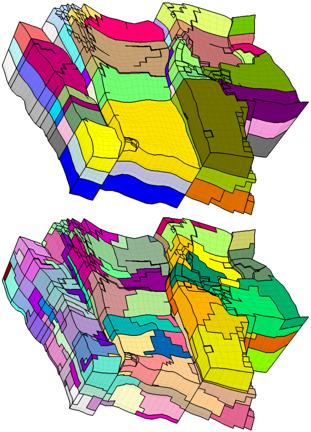


Higher resolution: $80 \times 100 \times 52$ cells, 416 000 active
Partition: rectangular (upper) and by Metis (lower)



Numerical example: Gullfaks





Summary

Presented a number of different multiscale methods:

- 15+ years of research with many detours/focus on unimportant issues
- MsRSB is probably the most simplistic found in the literature ...
- Large number of tests — very encouraging results!
- Finally, we seem to have a method that is working as required
- Key to efficiency: reduce accuracy, but retain mass conservation
- MsRSB is implemented in the INTERSECT R&P simulator
- MsMFE, MsFV, and MsRSB all available in MRST