Noname manuscript No.
(will be inserted by the editor)

Accelerating Multiscale Simulation of Complex Geomodels by Use of Dynamically Adapted Basis Functions

Øystein S. Klemetsdal · Olav Møyner · Knut-Andreas Lie

Received: date / Accepted: date

Abstract A number of different multiscale methods have been developed as a robust alternative to upscaling and as a means for accelerated reservoir simulation of high-resolution geomodels. In their basic setup, multiscale methods use a restriction operator to construct a reduced system of flow equations on a coarser grid, and a prolongation operator to map pressure unknowns from the coarse grid back to the original simulation grid. The prolongation operator consists of basis functions computed numerically by solving localized flow problems. One can use the resulting multiscale solver both as a CPR-preconditioner in fully implicit simulators or as an efficient approximate iterative linear solver in a sequential setting. The latter approach has been successful implemented in a commercial simulator. Recently, we have shown that you can obtain significantly faster convergence if you instead of using a single pair of prolongation-restriction operators apply a sequence of such operators, where some of the operators adapt to faults, fractures, facies, or other geobodies. Herein, we present how you can accelerate the convergence even further, if you also include additional basis functions that capture local changes in the pressure.

Ø.S. Klemetsdal
Norwegian University of Science and Technology
Tel.: +47 98 43 86 39
E-mail: oystein.klemetsdal@ntnu.no

O. Møyner
Norwegian University of Science and Technology/SINTEF Digital, Norway
E-mail: olav.moyner@sintef.no

K.-A. Lie
Norwegian University of Science and Technology/SINTEF Digital, Norway
E-mail: knut-andreas.lie@sintef.no

1 Introduction

A problem is said to have a multiscale character if it is governed by parameters or mechanisms that act across a wide range of spatial or temporal scales. In many applications there is a clear separation between local processes taking place on a microscale and the macroscale behavior of the whole system. Flow in porous media does not have a clear scale separation, which can be seen from the elliptic Poisson equations, $\nabla\cdot(\mathbf{K}\nabla p) = q$, modelling single-phase, incompressible flow. Here, $\mathbf{K}$ represents rock permeability and has a multiscale structure in the sense that spatial variations are characterized by a wide spectrum of length scales. Over the past decade, we have seen a large number of so-called multiscale methods [14, 8] that try to systematically account for small-scale variations and incorporate them in elliptic Poisson solvers formulated on a coarser scale. This is done by a pair of restriction and prolongation operators that restrict properties defined on a fine grid onto a coarse partition of the grid and map quantities from the coarse partition onto the fine grid, respectively. The prolongation operator is formed from a set of basis functions, computed numerically by solving localized versions of the original flow problem.

Multiscale methods were originally developed as a robust alternative to upscaling, which only represents subscale variations through a homogenized coefficient $\mathbf{K}^*$. Multiscale methods not only produce reduced models for computing flow solutions on a coarser grid, but the basis functions represent a systematic means to propagate this solution to define approximate solutions on fine or intermediate scales. More important, these methods offer a natural way to accelerate multiphase flow simulations by reusing basis functions from one step to the next. For this, it is particularly important

that the approximate pressure is monotone and that fluxes are mass conservative. Many multiscale methods have been proposed for incompressible flow on Cartesian grids or grids consisting of simplices, but only a few methods can provide realistic simulations of real hydrocarbon assets; see [26] for a more detailed discussion. To this end, a multiscale method must handle complex flow physics and the rough, unstructured grids that are usually encountered in simulation models. To simulate real assets, one also has to incorporate models of wells and surface facilities, with accompanying strategies for controlling the injection and production of fluids, and implement robust nonlinear solvers and time-stepping strategies.

Realistic flow physics is described by two main classes of models on the reservoir scale. Black-oil type models lump the chemical hydrocarbon species into two pseudo-components (a light gas component and a heavier oil component) that can mix and appear in a liquid oleic and/or a gaseous phase at reservoir conditions. Compositional simulators represent individual chemical species (or a wider range of pseudo-components) of hydrocarbons (and solvents) and use an equation of state to compute densities and fluid compositions at equilibrium states. In many cases, one can split the model equations into a flow equation for pressure and fluxes and a set of transport equations for saturations and compositions. These two are then solved consecutively in separate steps. First, the flow equation is solved while keeping saturations/concentrations fixed, and then the transport equations are solved with fixed pressures and fluxes [42, 37]. Flow equations typically have a certain elliptic character, whereas transport equations tend to be strongly hyperbolic [4]. A sequential approach makes it easy to use different discretizations and solvers for the two types of equations, and, in particular, offers a natural way to incorporate multiscale methods developed for (elliptic) flow equations. Sequential solutions can reduce computational costs significantly but also lead to inaccurate solutions that fail to resolve the correct coupling between pressure and the other variables. This can to a large degree be mitigated by adding outer iterations over the flow equation and transport equation steps [16]. This approach is used in state-of-the-art multiscale solvers and has produced good results for challenging black-oil [28, 18, 19, 26, 3] and compositional cases [31]. In some cases, the best way to resolve couplings between flow and transport is nevertheless to use a fully implicit discretization and solve for all variables simultaneously. The resulting system of discrete equations is usually ill-conditioned and requires special solution strategies. One can use the sequential splitting procedure as an initial guess for the fully implicit system. Couplings between pressure and other variables are chiefly local [22] and one can reduce the fully implicit system significantly by use of certain indicators for the splitting error [30]. So-called constrained-pressure-residual (CPR) methods [39, 40] constitute another popular approach. In CPR, one computes an inexpensive estimate to a reduced elliptic 'pressure' equation and uses this as a preconditioner for the full linearized system in an iterative Newton–Raphson type solution procedure. In early 2013, our group conducted the first tests with the multiscale finite-volume method [15] as a solver for the CPR step in a fully implicit black-oil simulator. With our prototype MATLAB simulator, we observed approximately 50% reduction in computational costs for the two-phase SPE 10 benchmark [5] and approximately 30% reduction for a refined version of the three-phase SPE 1 benchmark [32]. This was much less than we hoped for, and results were never published. Cusini et al. [7] later reported results from a more in-depth study. Herein, we go back and revisit this idea, this time with a more robust multiscale solver, the multiscale restriction-smoothed basis (MsRSB) method [29], which has been developed to handle the type of complex, unstructured grids encountered in contemporary simulation models.

Reservoir models generally adapt to surfaces that describe both the external and internal geology. This gives highly deformed and skew cell geometries with high aspect ratios and large variations in interface areas. Experience shows that the coarse partition used in multiscale methods should try to preserve the stratigraphic architecture and the stratigraphy as closely as possible and thus adapt to faults, major fractures, and internal stratigraphic layering. Petrophysical properties usually form a cell-wise representation of volumetric elements like depositional environments, flow units, channels, and lobes from an underlying geological model. Preserving these may be important to maximize the accuracy of the reduced, coarse-scale equations. Generating a single coarse partition that takes all these constraints into account is very challenging and seldom possible. However, this is fortunately not necessary, since multiscale methods like the MsRSB method are usually both robust and relatively accurate for a wide variety of coarse partitions. On the other hand, Lie et al. [25] recently demonstrated how one can improve the convergence rate for the pressure problem significantly by using multiple restriction/prolongation operators, where each operator targets specific geological features or provides increased resolution near wells. In this work, we investigate whether we can apply the same ideas to the CPR pressure preconditioner to improve convergence rates for the full system. We also

discuss how to honor strong couplings between pressure and the other unknowns by use of dynamically adapted basis functions that capture local pressure changes. We demonstrate the new ideas on examples ranging from simple conceptual models to a WAG scenario posed on a realistic shallow-marine reservoir model.

2 Model equations

For a fluid with an aqueous phase and two hydrocarbon phases made up of N_c components, we state conservation of mass for component β as follows

$$\begin{aligned}\frac{\partial}{\partial t}\left(\phi\left[\rho_w S_w X_{w\beta} + \rho_o S_o X_{o\beta} + \rho_v S_v X_{v\beta}\right]\right)\\ + \nabla\cdot\left(\rho_w X_{w\beta}\mathbf{v}_w + \rho_o X_{o\beta}\mathbf{v}_o + \rho_v X_{v\beta}\mathbf{v}_v\right) = q_\beta.\end{aligned} \tag{1}$$

Here, ρ_α and S_α are the mass density and saturation of phase α, $X_{\alpha\beta}$ is the mass fraction of component β in phase α, whereas q_β is the source/sink term of component β. The Darcy phase velocities $\mathbf{v}_\alpha$ read

$$\mathbf{v}_\alpha = -\frac{k_{r\alpha}}{\mu_\alpha}\mathbf{K}\left(\nabla p_\alpha - \rho_\alpha g\nabla z\right), \tag{2}$$

where μ_α and p_α denote viscosity and pressure of phase α, ϕ and $\mathbf{K}$ the rock porosity and permeability, $k_{r\alpha}$ models the reduced permeability experienced by one phase in the presence of another, g is the gravity acceleration, and z is the vertical coordinate. In most cases, the water component only appears in the aqueous phase, so that $X_{w,\beta} = 1$ if β is the water component, and thermodynamic equilibrium is determined by the fugacity balance between liquid and vapor for each of the hydrocarbon components:

$$f^o_\beta(p, T, X_{o1}, \ldots, X_{oN_c}) = f^v_\beta(p, T, X_{v1}, \ldots, X_{vN_c}).$$

The system is closed by assuming that the phases occupy the entire pore space, and that the mass fractions sum to unity for each phase:

$$S_w + S_o + S_v = 1, \quad \sum_{\beta=1}^{N_c} X_{\alpha\beta} = 1, \quad \alpha = w, o, v.$$

Finally, the phase pressures are related to each other through capillary pressure equations on the form

$$p_o = p_w + P_{cow}(S_w, S_o), \quad p_v = p_o + P_{cvo}(S_o, S_v).$$

The type of multiscale methods considered herein were originally developed to solve Poisson type equations arising in incompressible flow models. Let us therefore explicitly state the corresponding flow equation for the special case of a two-phase model with an aqueous and a liquid phase. Assuming that each phase only consists of a single component, we can sum the conservation equations (1) and use the fact that the saturations sum to unity to derive the following elliptic equation for the fluid pressure,

$$\begin{aligned}\nabla\cdot\left(\frac{1}{\mu_o}\left[k_{ro} + \frac{\mu_o}{\mu_w}k_{rw}\right]\mathbf{K}\nabla p_o\right) = q\\ -\nabla\cdot\left(\frac{k_{ro}}{\mu_o}\mathbf{K}\nabla P_{cow} + \frac{\rho_o}{\mu_o}\left[k_{ro} + \frac{\rho_w}{\rho_o}\frac{\mu_o}{\mu_w}k_{rw}\right]g\mathbf{K}\nabla z\right).\end{aligned} \tag{3}$$

This equation is coupled to a hyperbolic (or parabolic) saturation equation through the phase mobilities $\lambda_\alpha = k_{r\alpha}/\mu_\alpha$ and the capillary pressure function P_{cow}. In the special case of linear relative permeabilities and equal viscosities, the viscous term does not depend on saturation and reduces to the single-phase case $\mathbf{v} = \frac{1}{\mu_o}\mathbf{K}\nabla p_o$. For fixed relative permeabilities, the coupling depends on the viscosity ratio. A similar relation holds for the gravity term: the coupling vanishes for linear relative permeabilities and equal viscosities and densities, but depends in the general case on the viscosity and density ratios.

3 Discrete equations

We introduce a grid with cells $\{\Omega_i\}_{i=1}^N$, use backward Euler for the temporal discretization, and integrate over each cell in space using the divergence theorem and the midpoint rule to obtain the following discrete form of (1):

$$F^i_\beta = A^i_\beta + \sum_{j\in\mathcal{N}(i)} G^{i,j}_\beta - Q^i_\beta = 0, \quad i = 1, \ldots, N, \tag{4}$$

where

$$A^i_\beta = \sum_{\alpha=w,o,v} A^i_{\alpha,\beta}, \qquad \text{(Accumulation)} \tag{5}$$

$$G^{i,j}_\beta = \sum_{\alpha=w,o,v} G^{i,j}_{\alpha,\beta}, \qquad \text{(Intercell fluxes)} \tag{6}$$

$$Q^i_\beta = \frac{\Delta t}{|\Omega_i|}(q_\beta)^{n+1}_i. \qquad \text{(Sources/sinks)} \tag{7}$$

Accumulation and intercell flux for each phase read

$$A^i_{\alpha,\beta} = \left(\phi\rho_\alpha S_\alpha X_{\alpha\beta}\right)^{n+1}_i - \left(\phi\rho_\alpha S_\alpha X_{\alpha\beta}\right)^n_i,$$

$$G^{i,j}_{\alpha,\beta} = \frac{\Delta t}{|\Omega_i|}|\Gamma_{ij}|\left(\rho_\alpha X_{\alpha\beta}\mathbf{v}_\alpha\cdot\mathbf{n}\right)^{n+1}_{ij}.$$

Subscript i refers to values in cell Ω_i with bulk volume $|\Omega_i|$, subscript ij refers to interface values at interface Γ_{ij} with area $|\Gamma_{ij}|$, and superscript n refers to the time

step. The set $\mathcal{N}(i)$ consists of indices to all cells sharing a common interface with cell i. As is standard in reservoir simulation, we use a two-point approximation for the flux terms and single-point upstream evaluation for the interface mobilities, see e.g., [23] for details, although other choices are equally possible.

To write the discrete system in matrix form, we first use a Schur complement reduction to remove well equations and closure relations local to each cell. This gives a system of $N_c \times N$ equations, with N cell pressures and $(N_c - 1) \times N$ additional non-pressure variables as unknowns:

$$\begin{aligned}\mathbf{F} &= (\mathbf{F}_1, \ldots, \mathbf{F}_{N_c})^T \\ &= (F_1^1, \ldots, F_1^N, \ldots, F_{N_c}^1, \ldots, F_{N_c}^N)^T, && N_c \times N \\ \mathbf{x} &= (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_c}) \\ &= (p_1, \ldots, p_N, && N \\ &\quad x_2^1, \ldots, x_2^N, \ldots, x_{N_c}^1, \ldots, x_{N_c}^N), && (N_c - 1) \times N\end{aligned}$$

which we can write more compactly as $\mathbf{F} = (\mathbf{F}_p, \mathbf{F}_s)^T$ and $\mathbf{x} = (\mathbf{x}_p, \mathbf{x}_s)^T$. We can now write the system (4) as $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ and use Taylor expansion around $\mathbf{x}$,

$$\mathbf{F}(\mathbf{x} + \Delta\mathbf{x}) = \mathbf{F}(\mathbf{x}) + \mathbf{J}\Delta\mathbf{x} + \mathcal{O}(|\Delta\mathbf{x}|^2),$$

where $\mathbf{J} = \partial\mathbf{F}/\partial\mathbf{x}$ is the Jacobian matrix of $\mathbf{F}$. Neglecting higher-order terms, we obtain the Newton–Raphson method

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta\mathbf{x}^k, \quad -\mathbf{J}(\mathbf{x}^k)\Delta\mathbf{x}^k = \mathbf{F}(\mathbf{x}^k).$$

4 Preconditioning: the constrained pressure residual (CPR) method

We write the linearized Newton system in block matrix form as

$$-\begin{bmatrix}\mathbf{J}_{pp} & \mathbf{J}_{ps} \\ \mathbf{J}_{sp} & \mathbf{J}_{ss}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{x}_p \\ \Delta\mathbf{x}_s\end{bmatrix} = \begin{bmatrix}\mathbf{F}_p \\ \mathbf{F}_s\end{bmatrix}. \tag{8}$$

In most cases, (8) contains so many unknowns that we must use an iterative linear solver, whose convergence depends highly on the spectrum of the matrix. There are two main factors that contribute to impede convergence: the mixed elliptic–hyperbolic character of the flow equations, which makes pressure a strong variable, and large aspect ratios and variations in rock properties, which give high condition numbers. An effective preconditioning strategy [22] is thus required to ensure the effectiveness of the iterative solver. Herein, we apply a so-called constrained-pressure-residual (CPR) method [39, 40, 10], which relies on an inexpensive estimate of the pressure update $\Delta\mathbf{x}_p$ as an initial guess for the solution to the entire system.

The first step in a CPR method is to decouple (8) into a system on the form

$$-\begin{bmatrix}\mathbf{J}_{pp}^* & \mathbf{J}_{ps}^* \\ \mathbf{J}_{sp} & \mathbf{J}_{ss}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{x}_p \\ \Delta\mathbf{x}_s\end{bmatrix} = \begin{bmatrix}\mathbf{F}_p^* \\ \mathbf{F}_s\end{bmatrix},$$

where the matrix product $(\mathbf{J}_{pp}^*)^{-1}\mathbf{J}_{ps}^*$ is sufficiently small so that the solution to $\mathbf{J}_{pp}^*\Delta\mathbf{x}_p = \mathbf{F}_p^*$ is a reasonable approximation to the pressure. In addition to reducing the coupling strength, the decoupling should ultimately [35]

- reduce the condition number of the overall system;
- reduce the condition number of the block $\mathbf{J}_{pp}$;
- render $\mathbf{J}_{pp}^*$ and $\mathbf{J}_{ss}^*$ as M-matrices;
- be computationally inexpensive to perform.

Popular decoupling strategies include alternate-block factorization (ABF) [2], IMPES and quasi-IMPES (IMplicit Pressure, Explicit Saturation) [6, 21], and dynamic row sum (DRS) [10]. Herein, we follow the IMPES and quasi-IMPES approaches. Like most of the decoupling methods described in the literature, these two methods utilize the fact that couplings between pressure and the other quantities are chiefly local [22]. Algebraically, this implies that the diagonal terms of the coupling block $\mathbf{J}_{ps}$ dominate the off-diagonal terms. Thus, since we only seek an approximation to the pressure update, it is sufficient to reduce diagonal coupling terms of $\mathbf{J}_{ps}$. The IMPES decoupling amounts to finding weights $\mathbf{w}_\beta = (w_{\beta,1}, \ldots, w_{\beta,N})^T$ so that the derivatives of the accumulation terms with respect to non-pressure variables cancel:

$$\sum_{\beta=1}^{N_c} \mathbf{W}_\beta \frac{\partial \mathbf{A}_\beta}{\partial \mathbf{x}_k} = \mathbf{0}, \quad k = 2, \ldots, N_c.$$

Here, $\mathbf{W}_\beta$ is a diagonal matrix with $(\mathbf{W}_\beta)_{ii} = w_{\beta,i}$, and $\mathbf{A}_\beta = (A_\beta^1, \ldots, A_\beta^N)$ is the vector of accumulation terms for the β-th component, as defined in (5). The quasi-IMPES approach aims to eliminate all couplings between pressure and non-pressure variables in the same cell:

$$\sum_{\beta=1}^{N_c} \mathbf{W}_\beta \operatorname{diag}\left(\frac{\partial \mathbf{F}_\beta}{\partial \mathbf{x}_k}\right) = \mathbf{0}, \quad k = 2, \ldots, N_c,$$

where $\operatorname{diag}(\partial\mathbf{F}_\beta/\partial\mathbf{x}_k)$ refers to the diagonal of the Jacobian block $\partial\mathbf{F}_\beta/\partial\mathbf{x}_k$. In either case, the decoupling

procedure amounts to solving a system on the form

$$\begin{bmatrix} \mathbf{M}_{2,1}^T & \dots & \mathbf{M}_{2,N_c}^T \\ \vdots & \ddots & \vdots \\ \mathbf{M}_{N_c,1}^T & \dots & \mathbf{M}_{N_c,N_c}^T \\ \mathbf{I} & \dots & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \vdots \\ \mathbf{w}_{N_c} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix},$$

$$\text{where} \quad \mathbf{M}_{k,\beta} = \begin{cases} \frac{\partial \mathbf{A}_\beta}{\partial \mathbf{x}_k} & \text{IMPES}, \\ \operatorname{diag}\left(\frac{\partial \mathbf{F}_\beta}{\partial \mathbf{x}_k}\right) & \text{quasi-IMPES}, \end{cases}$$

and then premultiply $\mathbf{J}$ and $\mathbf{F}$ by

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \dots & \mathbf{W}_{N_c} \\ \mathbf{0} & \mathbf{I} \dots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & \mathbf{I} \end{bmatrix}, \quad \text{where} \quad (\mathbf{W}_\beta)_{ii} = w_{\beta,i}.$$

The CPR method presumes that the solution to the elliptic pressure equation dominates the solution to the full system. When this is not the case, CPR may not be beneficial at all. Gries et al. [10] describes a heuristic approach to turn off two-stage preconditioning when the coupling is stronger than a given threshold.

5 Multiscale methods

In this section, we briefly describe the framework of algebraic multiscale methods, before we discuss how to apply such methods as pressure solvers in the CPR method. Consider a discrete and linearized pressure equation on the form

$$\mathbf{A}\mathbf{p} = \mathbf{q}, \tag{9}$$

defined over a fine grid $\{\Omega_i\}_{i=1}^N$ that incorporates all details of the geological model. Multiscale methods start from a coarse partition $\{\Omega_i^c\}_{i=1}^M$ of the fine grid consisting of $M < N$ blocks defined so that each cell Ω_i in the fine grid belongs to a single block Ω_j^c in the coarse grid. We then associate basis functions that map the pressure degrees of freedom on the coarse grid to degrees of freedom on the fine grid. We can collect the basis functions in a prolongation operator $P : \{\Omega_j^c\} \to \{\Omega_i\}$ and express it in matrix form as an $N \times M$ matrix $\mathbf{P}$. Element i, j of $\mathbf{P}$ is the value of the jth basis function in the ith cell. Given a coarse-scale approximation p_c, we can now compute an approximation to the fine-scale pressure by prolongation of p_c back to the fine grid, $\mathbf{p} \approx \mathbf{P}\mathbf{p}_c$. Analogously, we define a restriction operator $R : \{\Omega_i\} \to \{\Omega_j^c\}$ that maps quantities from the fine grid onto the coarse blocks. We write this operator in matrix form as an $M \times N$ matrix $\mathbf{R}$. To form a reduced linear system on the coarse grid, we substitute $\mathbf{p}$ by $\mathbf{P}\mathbf{p}_c$ in (9) and multiply by the restriction operator from the left on both sides of the equation,

$$(\mathbf{RAP})\mathbf{p}_c = \mathbf{R}\mathbf{q}, \quad \text{or} \quad \mathbf{A}_c\mathbf{p}_c = \mathbf{q}_c. \tag{10}$$

To be feasible, any pair of operators (P, R) should fulfill the following three requirements [25]:

1. Both operators are defined over the same non-overlapping coarse partition of the fine grid. Each column of $\mathbf{P}$ constitutes a basis function, and is associated with a coarse grid block.
2. The support of each basis function is compact and must contain the associated coarse block. In other words, the support region S_j of the jth basis function must satisfy the following conditions: $\Omega_j^c \subset S_j \subset \cup_{k=1}^M \Omega_k^c$
3. The columns of $\mathbf{P}$ and the rows of $\mathbf{R}$ must form partitions of unity over the fine grid; that is, each row of $\mathbf{P}$ has unit row sum and each column of $\mathbf{R}$ has unit column sum.

Specific multiscale methods differ in the way they define the prolongation and restriction operators. This type of method was first proposed by Hou and Wu [14] and has later been developed in many different directions, see [26] for a review focused on methods that could be applicable to practical reservoir simulation. The main contenders used to be the multiscale finite-volume (MsFV) method [15] and the multiscale mixed finite-element (MsMFE) method [1], but these have later been superseded by the multiscale restriction-smoothed basis (MsRSB) method [29]. The mixed method is formulated for an extended mixed hybrid system that also contains unknowns for phase fluxes and pressures at cell interfaces.

How one should interpret (10) depends on the choice of operators. For the restriction operator, a natural choice is to set $\mathbf{R} = \mathbf{P}^T$, which corresponds to a Galerkin coarse-scale discretization. This does not give a locally mass-conservative scheme on the coarse scale, and is consequently not a good choice in a standalone solver for the pressure equation. If fine-scale fluxes are needed, it is better to use a finite-volume operator that sums the entries of the corresponding cells for each block. Herein, we use multiscale as a preconditioner [41] and ensure mass conservation by converging the full system. Galerkin restriction is then the preferable choice, since it preserves the (hopefully) symmetric and M-matrix properties of the fine-scale pressure matrix. Figure 1 depicts the multiscale solution procedure, which we can interpret as a two-level algebraic multigrid method, albeit with a very large coarsening ratio.

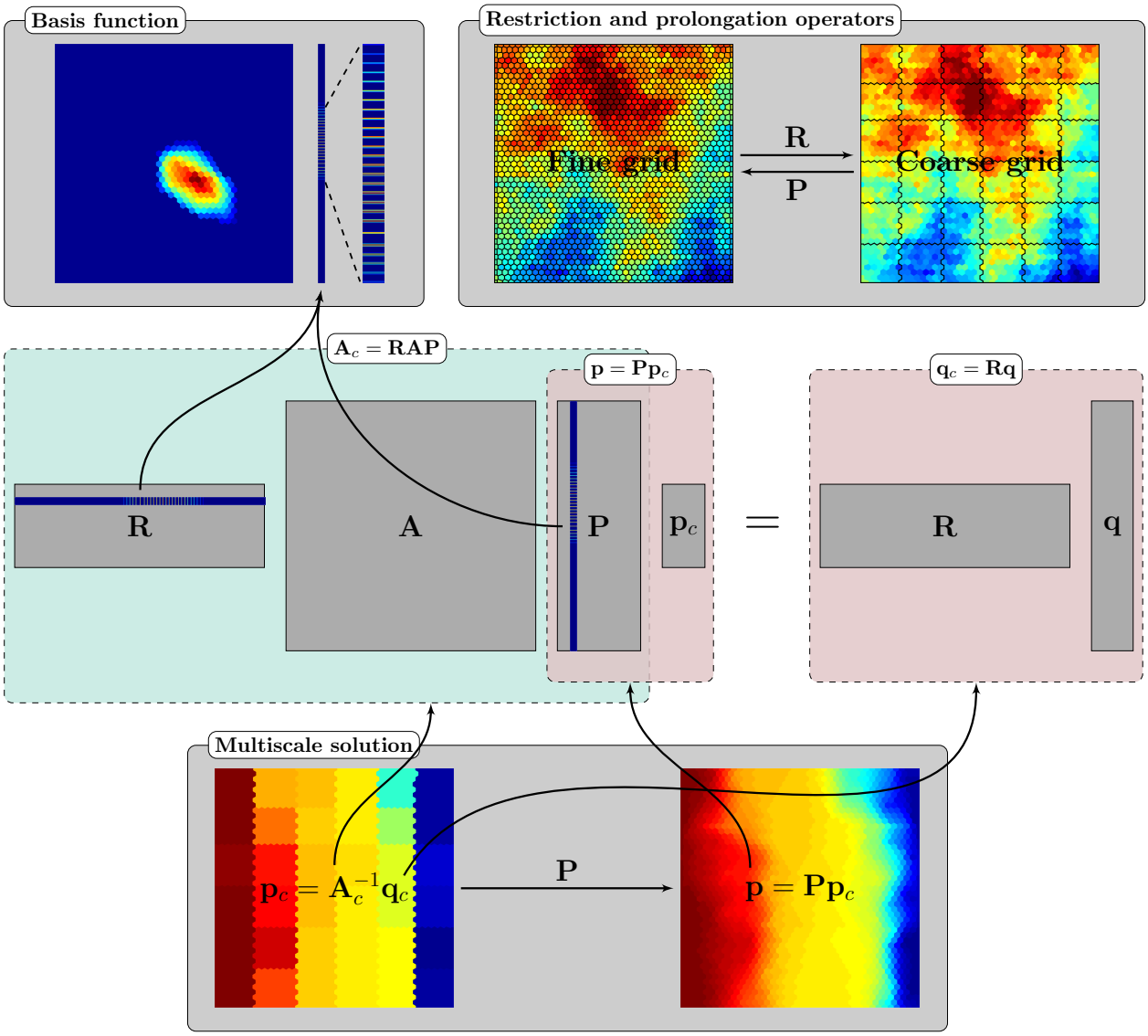


Fig. 1 Illustration of the multiscale solution procedure. The middle section shows how the full system reduces to $\mathbf{A}_c\mathbf{p}_c = \mathbf{q}_c$. In the upper right corner, we see how the restriction and prolongation operators map between the two scales. In the upper left corner, one of the basis functions is depicted in physical space and in vector form as it appears in the restriction and prolongation operators as a row or column. The solution procedure is shown at the bottom, where we first obtain the coarse-scale pressure by solving the reduced system, and then prolongate it onto the fine grid to obtain the fine-scale pressure solution.

5.1 Iterative multiscale multibasis solver

As in standard multigrid theory [38], the multiscale solution typically resolves global low-frequency errors quite effectively, but contains local high-frequency errors due to the localization introduced to define basis functions. It is therefore natural to cast the multiscale method in an iterative framework [11, 41] and combine the multiscale solver with a smoothing step that aims to remove high-frequency errors. One iteration then consists of two steps:

$$\mathbf{x}^{k+1/2} = \mathbf{x}^k + S(\mathbf{A}, \mathbf{q} - \mathbf{A}\mathbf{x}^k),$$
$$\mathbf{x}^{k+1} = \mathbf{x}^{k+1/2} + \mathbf{P}\mathbf{A}_c^{-1}\mathbf{R}(\mathbf{q} - \mathbf{A}\mathbf{x}^{k+1/2}),$$

where $S(\mathbf{A}, \mathbf{b})$ denotes a function that performs one or more smoothing operations, e.g., incomplete LU factorization.

The initial convergence rate of such a two-stage multiscale preconditioner is typically satisfactory, but deteriorates when the error is made up of intermediate-frequency modes that are neither reduced by the smoother nor the multiscale step. When used as pressure solver in a CPR method, convergence to a strict tolerance is not

necessary, and we are instead interested in reducing the error as much as possible using only one or a few iterations. Lie et al. [25] previously showed that the initial reduction can be accelerated significantly if we instead of using basis functions associated with a single coarse partition use a sequence of prolongation/restriction operators associated with different coarse partitions that each targets specific features of the flow field and/or the geological model. In particular, we define a set of prolongation operators $\{P^1, \ldots, P^{N_p}\}$ and restriction operators $\{R^1, \ldots, R^{N_p}\}$ that each corresponds to a (unique) coarse grid $\{\Omega_i^c\}_{i=1}^{M_\ell}$ and satisfies requirements 1 to 3 above. This yields an iterative multiscale multibasis method on the form

$$\begin{aligned}\mathbf{x}^{k+(2\ell-1)/2N_p} &= \mathbf{x}^{k+(\ell-1)/N_p} \\ &\quad + S^\ell(\mathbf{A}, \mathbf{q} - \mathbf{A}\mathbf{x}^{k+(\ell-1)/N_p}), \\ \mathbf{x}^{k+\ell/N_p} &= \mathbf{x}^{k+(2\ell-1)/2N_p} \\ &\quad + \mathbf{P}^\ell(\mathbf{A}_c^\ell)^{-1}\mathbf{R}^\ell(\mathbf{q} - \mathbf{A}\mathbf{x}^{k+(2\ell-1)/2N_p}).\end{aligned} \tag{11}$$

Employing a nested sequence of coarse partitions to capture different error modes is the standard approach in algebraic multigrid methods. In our case, the coarse partitions are all defined relative to the original fine grid and are not necessarily nested.

Careful analysis of many simulation cases have lead us to the following three observations:

(O1) Multiscale operators defined for partitions that cover the entire domain evenly will usually resolve the global pressure (as given by the CPR pressure equation) quite accurately, but may introduce local errors that stem from localization of the basis functions. Such errors typically arise when the support of the basis function contains barriers or regions with distinctively different flow properties, or are the result of significant changes in the drive mechanisms.

(O2) Local errors introduced by geological variations, or in near-well regions, can be more effectively reduced by coarse partitions that adapt to these spatial features.

(O3) Differences between the CPR pressure and the true pressure tend to arise because of inexact decoupling of the pressure equation. This effect increases with the coupling strength and typically gives temporal errors located near propagating displacement fronts. By adapting partitions to dynamic changes in the pressure update, or saturations and/or mass fractions, one can hope to obtain multiscale operators that resolve this coupling better.

With these observations in mind, we herein consider three types of multiscale basis functions:

General: The type of partition one would use to upscale the model and/or form a coarse-scale discretization; that is, a partition with small variations in block sizes. Such partitions can be rectilinear/structured, or generated by a graph partitioning algorithm [17], possibly with transmissibilities as the connection strength to minimize the permeability variation inside each block. In any configuration, our multiscale solvers will use at least one such partition with a prolongation operator generated from MsRSB basis functions.

Static: Partitions constructed to target specific features in the geocellular model. These partitions could be constructed by increasing the coarse-grid resolution near features of interest such as fractures and well paths, and/or by ensuring that block interfaces follow geological layers, fault surfaces, boundaries between different rock types, flow units, and depositional environments, etc. One effective way to generate such partitions is to agglomerate cells into blocks according to user-defined cell/face indicators and partitioning rules [13, 12, 24, 23]. Several partition examples are shown in [25]. We use MsRSB basis functions to define the resulting operators.

Dynamic: Basis functions designed to target dynamic couplings between pressure and saturations/mass fractions, i.e., the non-elliptic character of the pressure update. Such couplings are usually manifested as sharp transitions in the pressure updates across fluid interfaces. These partitions can be constructed using information (e.g., pressure update, velocity, saturations/mass fractions) from earlier nonlinear iterations. Herein, we either construct partitions by tracking displacement fronts, or by assigning cells into bins according to the magnitude of their pressure update from a previous nonlinear iteration. We use constant basis functions to construct the corresponding prolongation operators.

We typically compute general and static partitions and corresponding basis functions during a preprocessing step. Basis functions can also be updated locally during the simulation to account for changing reservoir/fluid properties and driving forces by iterating a few times extra on the localized (elliptic) residual equations that define the MsRSB basis functions. The dynamic partitions must be recomputed during the simulation to account for the movement of fluids. This operation is not very computationally expensive, since it essentially consists of comparing two or more floating point numbers per cell and assigning each cell to a coarse block (i.e., setting an integer number in a partition vector). The optimal frequency for recomputing these partitions is obviously a trade-off between accu-

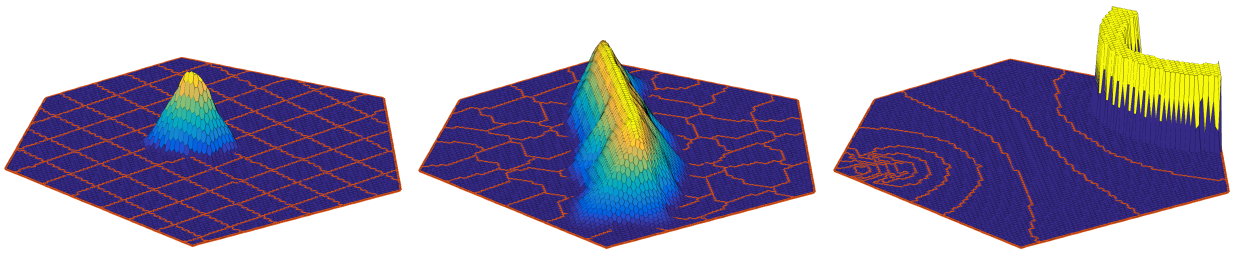


Fig. 2 Examples of different types of basis functions. General basis functions are defined over a structured partition, the static partition traces a major fracture inside the domain, and the dynamic partition is based on Δp. The first two types are constructed using MsRSB, whereas the dynamic partition uses constant basis functions.

rately capturing coupling between pressure and saturations/mass fractions and minimizing computational effort. Herein, we always use information (e.g., pressure update/phase saturations/mass fractions) from the most recent nonlinear iteration to compute dynamic basis functions before the next nonlinear iteration. We suggest the following update approach during a single time step:

1. *Before* the first nonlinear iteration, compute new dynamic basis functions based on information from the last nonlinear iteration of the previous time step.
2. We also update the dynamic basis functions before the second nonlinear iteration of the time step.
3. Before subsequent nonlinear iterations, we may also update the dynamic basis functions at a given update frequency.

This update approach is motivated by the fact that during the solution to a time step, large regions of nonzero Newton updates are usually resolved within the first few iterations, and subsequent Newton updates are typically restricted to a small portion of the grid cells within the same regions [34]. Figure 2 shows one basis function for each of the three types of partitions.

5.2 Description of the solution procedure

One can combine the multiscale method with various iterative methods to form the pressure step of CPR. Herein, we either use a standard Richardson iteration, or use the multiscale solver as a two-step preconditioner in a Generalized Minimum Residual solver (GMRES) [33]. The main steps in the solution procedure involved in one nonlinear iteration are thus: *(i)* linearization to obtain the linear system, *(ii)* decoupling to precondition the system for CPR, *(iii)* the predictor step in which we approximate the pressure using an iterative multiscale multibasis solver, and *(iv)* a correction step in which we smooth the solution to the full system. Figure 3 depicts the entire solution procedure.

6 Numerical examples

To test the effectiveness of the multiscale–multibasis method introduced above, we have implemented it in the open-source MATLAB Reservoir Simulation Toolbox (MRST) [23, 20]. Our conceptual implementation utilizes MATLAB quite efficiently, but is not yet fully optimized and has certain overhead one would not expect to see in a fully optimized code written in a compiled language. We therefore assess the computational efficiency in two steps: First, we present a number of test cases to assess how effective the multiscale, multibasis solver is as a CPR preconditioner. In each test case, we configure the approximate CPR solver to use various combinations of general, static, and dynamic partitions and then investigate which configuration requires fewest linear steps to reach the prescribed linear tolerance, averaged over all the nonlinear steps. Once we have investigated how the use of multiple multiscale bases affects the error reduction in the approximate CPR preconditioner, we analyze the theoretical number of floating-point operations required for the various stages of one linear iteration. Comparing the operational count for a multibasis preconditioner to a basic setup with a single MsRSB basis then gives us an indication of the potential for speedup.

6.1 Example 1: Coupling strength

To study how the effect of using dynamic basis functions depends on coupling strengths, we consider a simple incompressible two-phase model with an aqueous and a liquid phase, each consisting of a single component. Referring back to (3), we know that the coupling between pressure and saturation is determined by the viscosity ratio μ_w/μ_o, the density ratio ρ_w/ρ_o, and the magnitude of the capillary pressure P_{cow}. The geological model consists of a 42×22 subset from Layer 10 of Model 2 from the SPE10 benchmark [5], with the

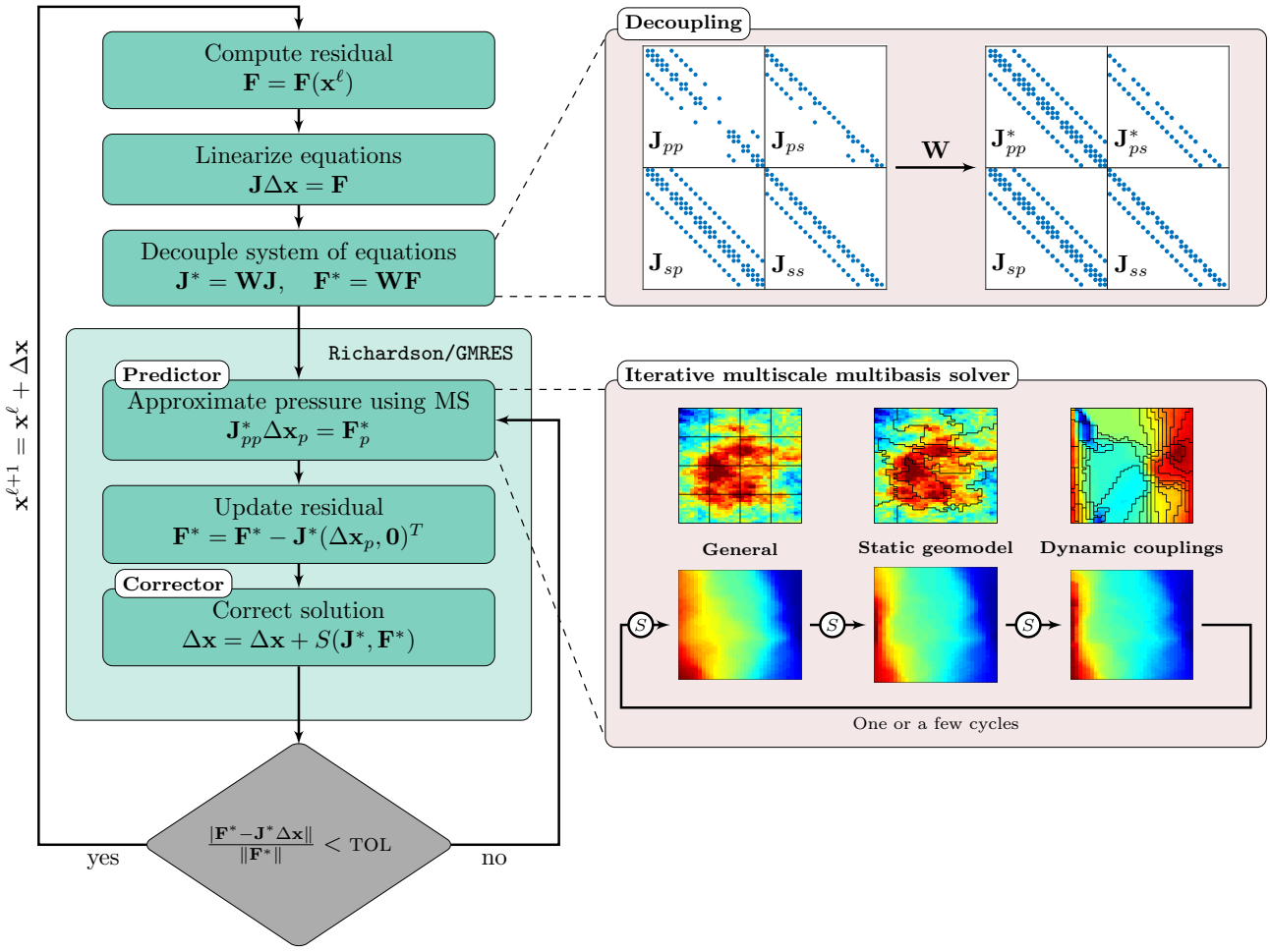


Fig. 3 Flow chart describing the entire solution procedure to advance the solution one time step. After linearization, the linear system is decoupled. This gives a pressure block $\mathbf{J}^*_{pp}$ with a typical elliptic structure, and a coupling block $\mathbf{J}^*_{ps}$ with a typical hyperbolic structure, as seen in the *Decoupling* box. The predictor step consists of one or a few cycles of the iterative multiscale multibasis method, as described in Equation (11). To reduce high-frequency error modes, we apply a smoother between each multiscale solution, indicated by an S in the *Iterative multiscale multibasis solver* box.

origin corresponding to cell $(10, 110)$ in the full layer. The reservoir is initially filled with the liquid phase, and we inject one pore volume of the aqueous phase through an injection well at the left boundary over a period of four years. A producer operated a bottom-hole pressure of 50 bars is placed at the right boundary. A rectilinear 6×2 partition serves as the general basis (see Figure 4), and we use the pressure update Δp to generate dynamic basis functions. The incompressible flow equation (3) is already elliptic and could be solved directly without any CPR reduction, but herein we use the general procedure outlined above. In our experience, multiscale solvers based on Richardson iterations are more sensitive to the quality of the preconditioner than solvers based on GMRES. We thus use a Richardson-type solver to emphasize the effect of adding a dynamic basis.

We neglect gravity and capillary forces, assume linear relative permeabilities for both phases, and let the viscosity ratio μ_w/μ_o vary from 50/1 to 1/50, normalized so that the less viscous phase has a viscosity of 1 cP. Figure 5 shows saturation profiles after 900 days for selected viscosity ratios. We clearly see how the dynamic partitions shown in the right column of the figure adapt to the saturation front for the two cases where the injected fluid is more viscous. Figure 6 reports the average number of linear iterations used for the different solvers, i.e., the number of iterations performed internally in the Richardson/GMRES box in Figure 3 averaged over all the nonlinear iteration steps. We see that adding a dynamic basis is beneficial in all cases except with unit viscosity ratio. This is because pressure and saturation in this case are completely decoupled, and the negligible difference of 0.8 % comes only from the smoothing iteration in the first step of (11). We

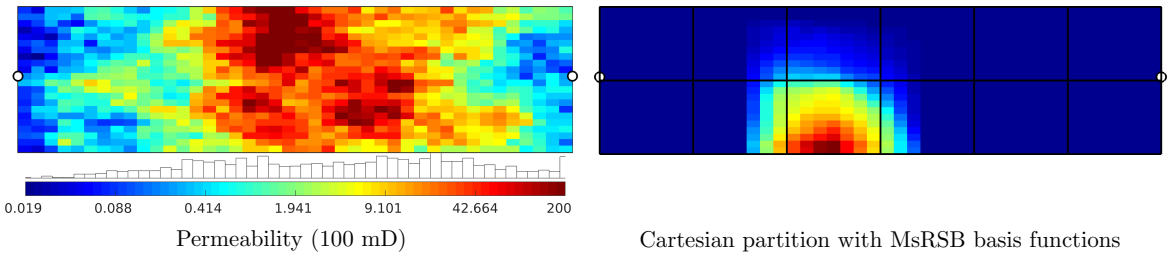


Fig. 4 Permeability and general basis for Example 1.

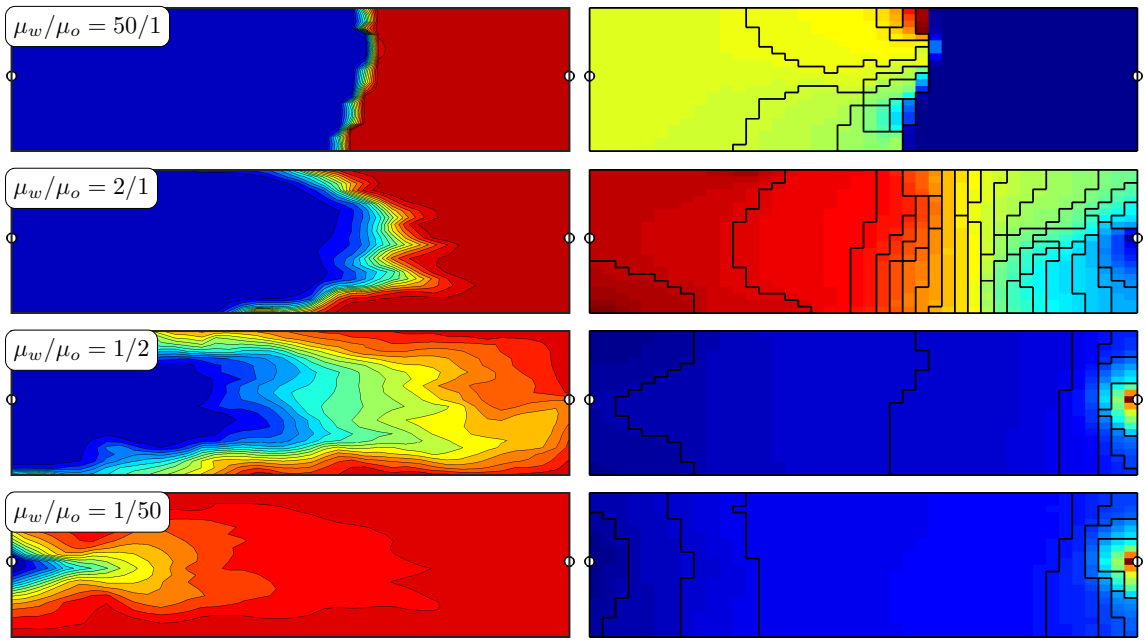


Fig. 5 The left column shows saturation profiles after 900 days with four different viscosity ratios for Example 1. The right column shows the corresponding dynamic partitions (black lines), with colors indicating the magnitude of the pressure update.

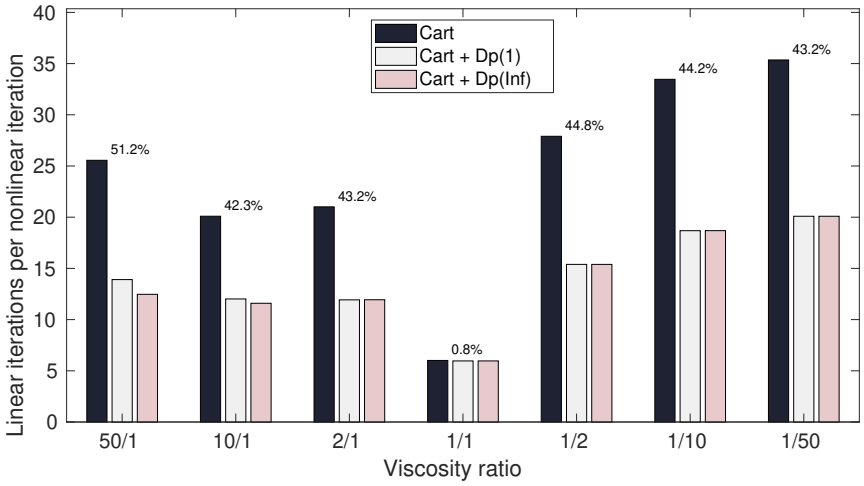


Fig. 6 Average number of linear iterations per nonlinear iteration used by the three solvers for different viscosity ratios in Example 1. Dp(1) indicates recomputation of dynamic basis functions before each nonlinear iteration, whereas Dp(Inf) indicates recomputation before nonlinear iteration one and two. Percentages indicate the relative difference between the best and worst performing solver.

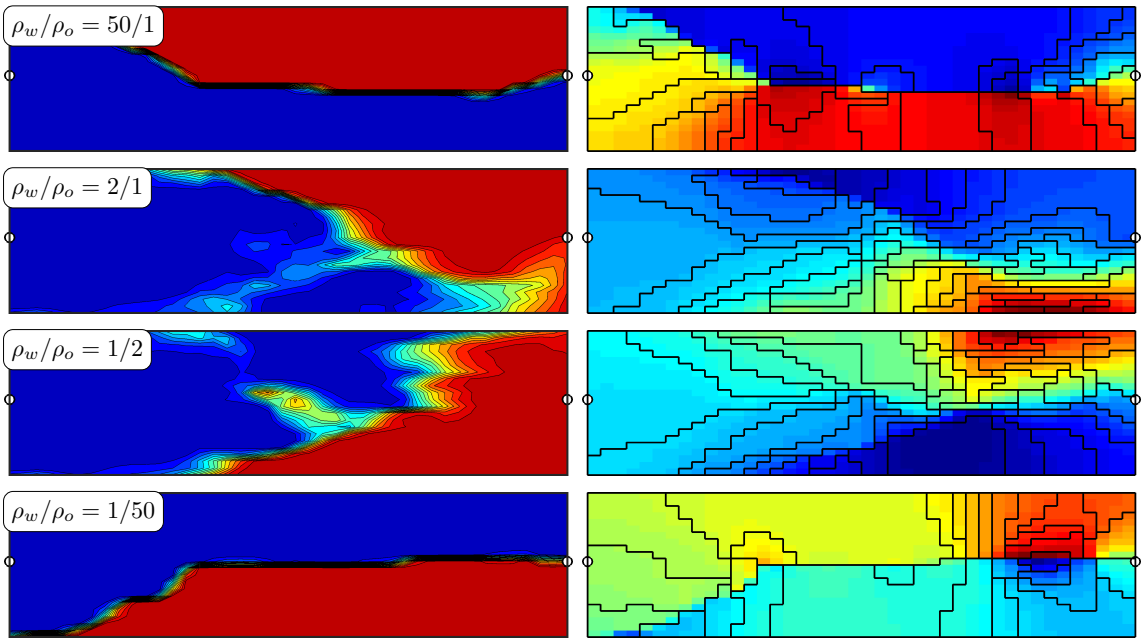


Fig. 7 The left column shows saturation profiles after 900 days for four different density ratios in Example 1. The right column shows the corresponding dynamic partitions (black lines), with colors indicating the magnitude of the pressure update.

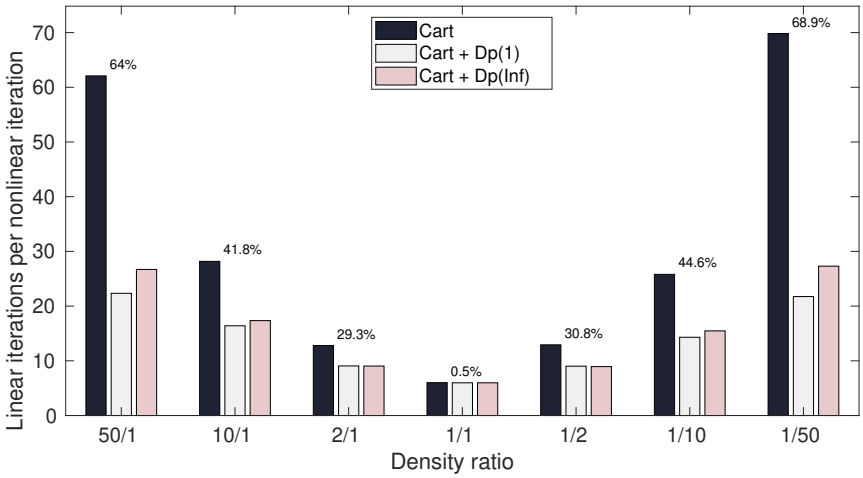


Fig. 8 Average number of linear iterations per nonlinear iteration used by the three solvers for different density ratios in Example 1. Dp(1) indicates that we recompute dynamic basis functions before each nonlinear iteration, whereas Dp(Inf) indicates that we only recompute before nonlinear iteration one and two. Percentages indicate the relative difference between the solvers with best and worst performance.

have also compared the effect of recomputing the dynamic basis functions before each nonlinear iteration instead of only before the first and second iterations. The solver with update before each iteration is denoted Dp(1), whereas the solver with update before iteration one and two is denoted Dp(Inf). Interestingly, we observe no reduction in the number of iterations by recomputing before each time step, and even an increase for large viscosity ratios. A possible explanation to this is that the last Newton updates of each time step are very small in most of the domain, so that the construction of dynamic partitions is influenced by noise.

Next, we set equal viscosities and look at the effect of varying the density ratio in the presence of gravity; that is, we vary the ratio ρ_w/ρ_o from 50/1 to 1/50, normalized so that the less dense fluid has a density of 10 kg/m^3. Gravity is aligned with the negative y-direction. Figure 7 shows the saturations after 900 days for different density ratios. We clearly see how the dynamic partitions adapt to the interface between the segregated fluids, especially for the highest and lowest density ra-

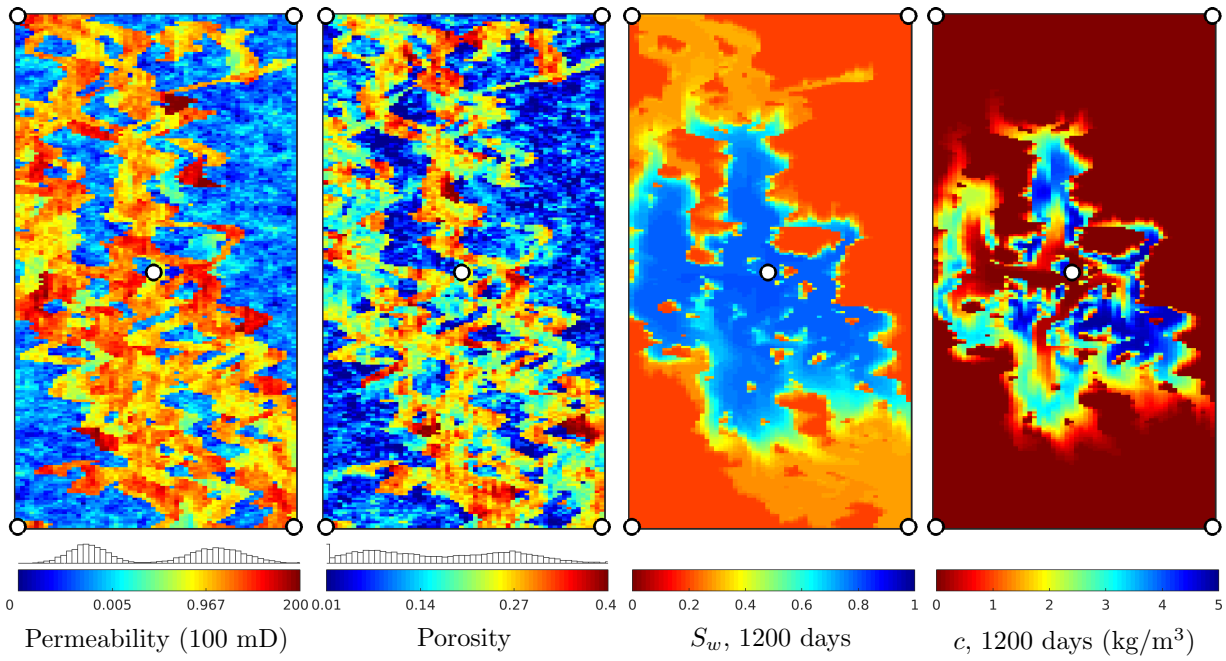


Fig. 9 Petrophysical properties for the case in Example 2, along with water saturation and polymer concentration after 1200 days of injection.

tios. Figure 8 reports the average number of linear iterations per nonlinear iteration used by the two solvers. Again, we see that adding a dynamic basis improves the convergence rate of the linear solver except for the case with unit density ratio. Moreover, compared with the case with varying viscosity ratios, the relative reduction in iterations is bigger and depends more strongly on the density ratio. For the extreme density ratios of $\rho_w/\rho_o = 50/1$ and $1/50$, the gravity segregation will take place very rapidly, giving sharp fluid interfaces where the pressure update is nearly discontinuous. These interfaces are nicely captured by the dynamic basis functions. In this case, we observe a slight reduction in the number of linear iterations by recomputing the dynamic basis functions before each nonlinear iteration (Dp(1)).

6.2 Example 2: Polymer injection

We consider polymer injection in Layer 52 of SPE10 Model 2 with wells placed in an inverted quarter five-spot pattern. The center well injects water at a constant rate of 9.14 m^3/day over a period of 2000 days, and the four corner wells operate at fixed bottom-hole pressures of 275 bar. The injected water will rapidly fill the high-permeable fluvial channels. To deviate water into the low-permeable zones, we inject a polymer slug (5 kg/m^3) from 400 to 800 days. The polymer is assumed to mix completely with water [36], increasing the viscosity of the injected phase from 0.3 cP to 60 cP. The polymer model also accounts for adsorption of polymer onto the porous rock. Shear effects and inaccessible pore space can also be accounted for in the model, but this is for simplicity not included in the example. See [3] for further details. The polymer effect on viscosity introduces a strong coupling between pressure and water saturation/polymer concentration. Figure 9 shows petrophysical properties, well positions, water saturation and polymer concentration after 1200 days. Unlike in the previous example, the system has compressibility and we thus use a CPR solver.

Figure 10 shows the different types of partitions we use to generate basis functions for the CPR preconditioner. The general basis consists of MsRSB basis functions defined on a partition constructed by METIS [17] with one-sided transmissibilities as edge weights. We also consider two partitions that honor the high permeability contrast and channeled structure of the reservoir, generated by an ad-hoc algorithm that agglomerates cells into coarse blocks based on a flow indicator [12]. The first partition uses permeability as flow indicator, whereas the second uses an incompressible velocity field computed a priori for the same grid and well setup. For both we use MsRSB basis functions to construct the corresponding prolongation operator. In addition, we use well bases and a set of dynamic basis functions with the pressure update Δp from the previous Newton iteration as indicator to agglomerate the grid cells.

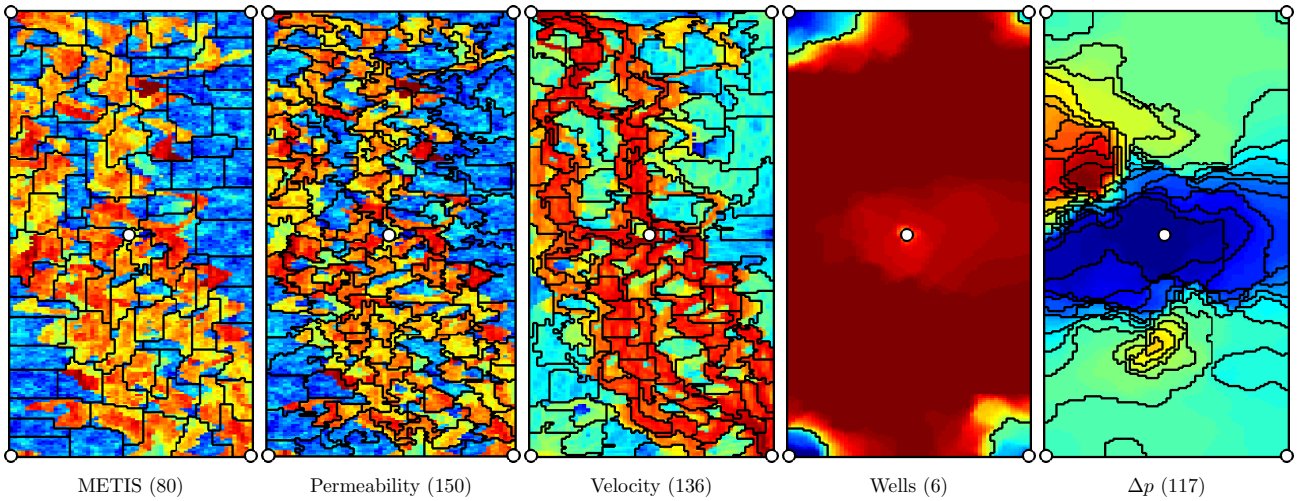


Fig. 10 Partitions for the different basis functions used in the polymer flooding in Example 2. The number of coarse cells for each partition are indicated in parentheses. The colors show the indicator used to generate the different partitions, with the exception of the well partition, where the actual basis functions are shown.

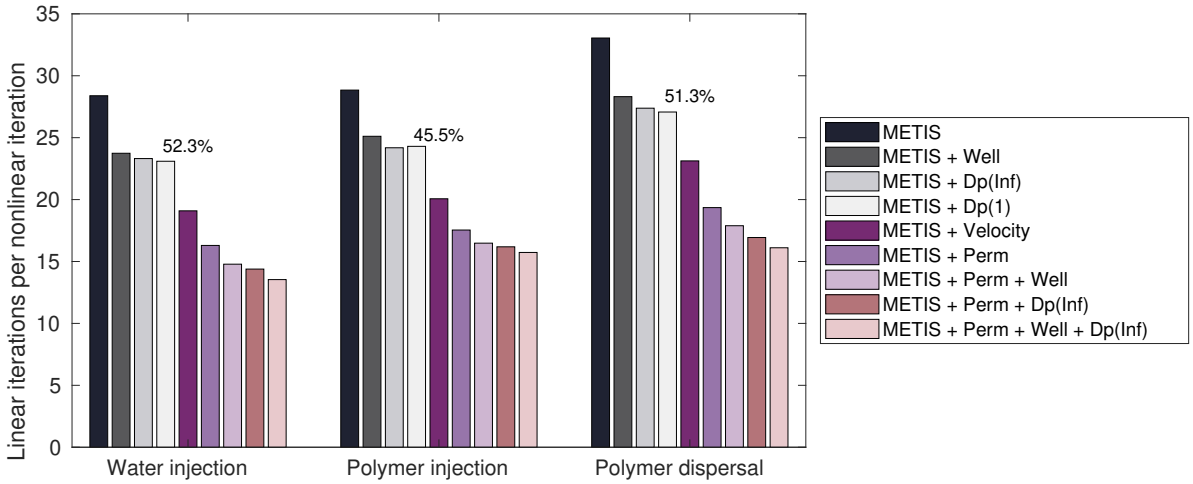
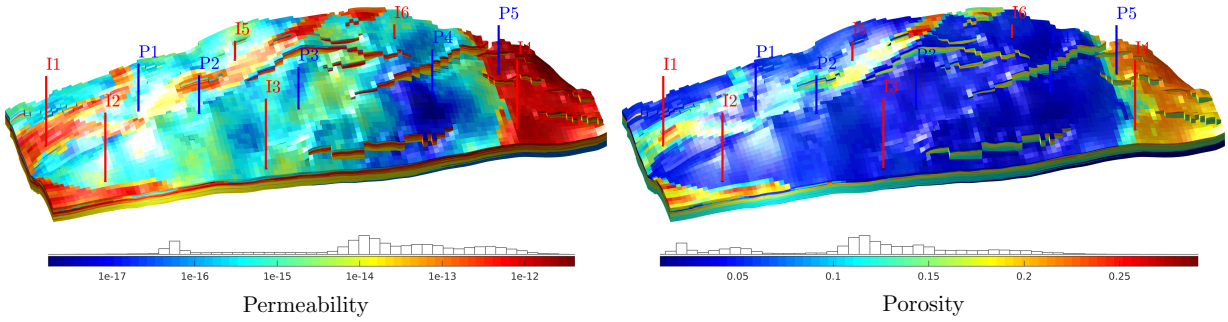


Fig. 11 Average number of linear iterations per nonlinear iteration with different combinations of multiscale operators for the CPR preconditioner in Example 2. Dp(1) indicates recomputation of dynamic basis functions before each nonlinear iteration, whereas Dp(Inf) indicates recomputation before nonlinear iteration one and two. Percentages indicate the relative difference between the best and worst performing solver.

Figure 11 reports the average number of linear iterations per nonlinear iteration for the different solvers during three separate phases of the recovery process: initial water injection, injection of the polymer slug, and the final water-injection period that disperses the injected polymer slug. The most significant reduction in iterations comes from adding static partitions that adapt to the high-flow channels, identified either by permeability or single-phase velocities. Using well basis functions typically gains 3–4 iterations. This comes at a very low computational cost, since the basis only consists of six coarse blocks (one for each well plus one to ensure partition of unity). The dynamic partitions have a similar reduction effect, but are generally more costly than the well bases. The dynamic bases reduce the number of iterations a bit more during the polymer dispersal phase, when the effective viscosity ratio between the displaced and displacing fluid is at its highest. Similar to the previous example, we observe only a slight reduction in the number of linear iterations when recomputing the dynamic basis functions before each nonlinear iteration (Dp(1)).

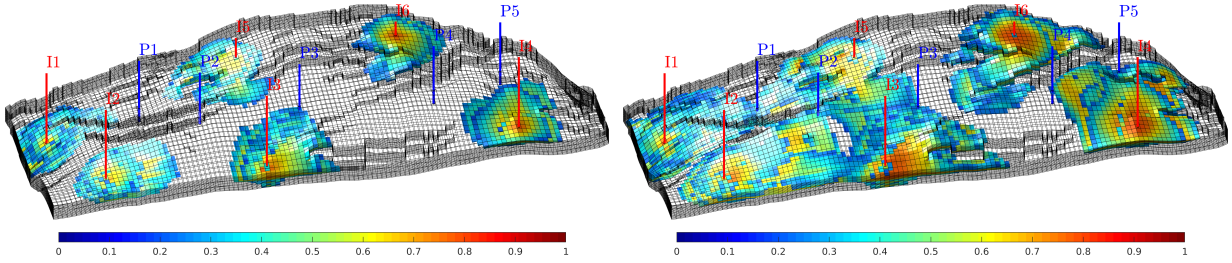
6.3 Example 3: Field model (SAIGUP)

The SAIGUP study [27] developed several realistic models of shallow-marine oil reservoirs based on data from real shoreface deposition sites, here represented by a



Permeability Porosity

Fig. 12 Petrophysical properties and well configuration for the SAIGUP field in Example 3.



Water saturation after initial water injection Water saturation after WAG + final water injection

Fig. 13 Water saturation before and after WAG + water injection in Example 3.

$40 \times 120 \times 20$ corner-point grid that spans a lateral area of approximately 9×3 km^2. The model has 78,720 active cells and describes several major faults. Permeability and porosity follow multimodal distributions, and the model identifies six different rock types with clearly distinct petrophysical characteristics. The reservoir also contains large amounts of mud-drapes that reduce the vertical permeability. Figure 12 shows porosity and lateral permeability and the location of the injection and production wells.

We consider a water-alternating-gas (WAG) injection with six injectors operating at a constant rate and five producers operating at constant bottom-hole pressure of 50 bar. WAG is a popular enhanced oil recovery strategy in which small volumes of water and gas are injected in a cycle. The injected gas will dissolve into the reservoir oil, which improves sweep efficiency and releases much of the trapped residual oil. Water is injected between the gas volumes to uphold the mobility of the displacement front. This technique is typically employed after an initial water flood. In our case, we start by injecting 0.8 pore volumes of water over a period of two years. Over the next two years we inject 0.8 pore volumes of alternating water and gas, followed by another 0.8 pore volumes of water.

Oil and gas are slightly compressible, with quadratic Brooks-Corey relative permeabilities. Densities and viscosities are 1000, 800, and 100 kg/m^3 and 1, 3, and 0.4 cP at reference pressure for the water, oil, and gas phases, respectively. The injected solvent gas has a density of 100 kg/m^3 and a viscosity of 0.5 cP, and mixes with the reservoir oil and gas according to a modified version of the Todd–Longstaff mixing rule [36]. This model is the same as implemented in a commercial simulator [9]. With only reservoir oil and reservoir gas, we have a traditional black-oil behavior, whereas the reservoir oil is assumed to be fully miscible with the solvent gas when no reservoir gas is present. In the intermediate region, we interpolate between the two extremes. The oil/solvent mixture has much lower density and viscosity than pure oil. In addition, solvent gas is assumed to lower the residual oil saturation from $S_{or,i} = 0.2$ in the immiscible case to $S_{or,m} = 0.1$ in the fully miscible case. The strong coupling between solvent gas saturation and reservoir fluid mobility means that the solution is strongly affected by the solvent saturation. As a consequence, the two-stage CPR preconditioner may not be as effective as for a standard waterflooding scenario. Figure 13 reports water saturations after initial water injection and at the end of the simulation.

Figure 14 shows the multiscale partitions used for the CPR preconditioner. These include a logically Cartesian partition and a partition generated using one-sided transmissibilities to measure coupling strengths between cells in the METIS graph-partitioning algorithm. These serve as general partitions. We include a static partition to honor the large variations in permeability and another static well partition that accounts for changes

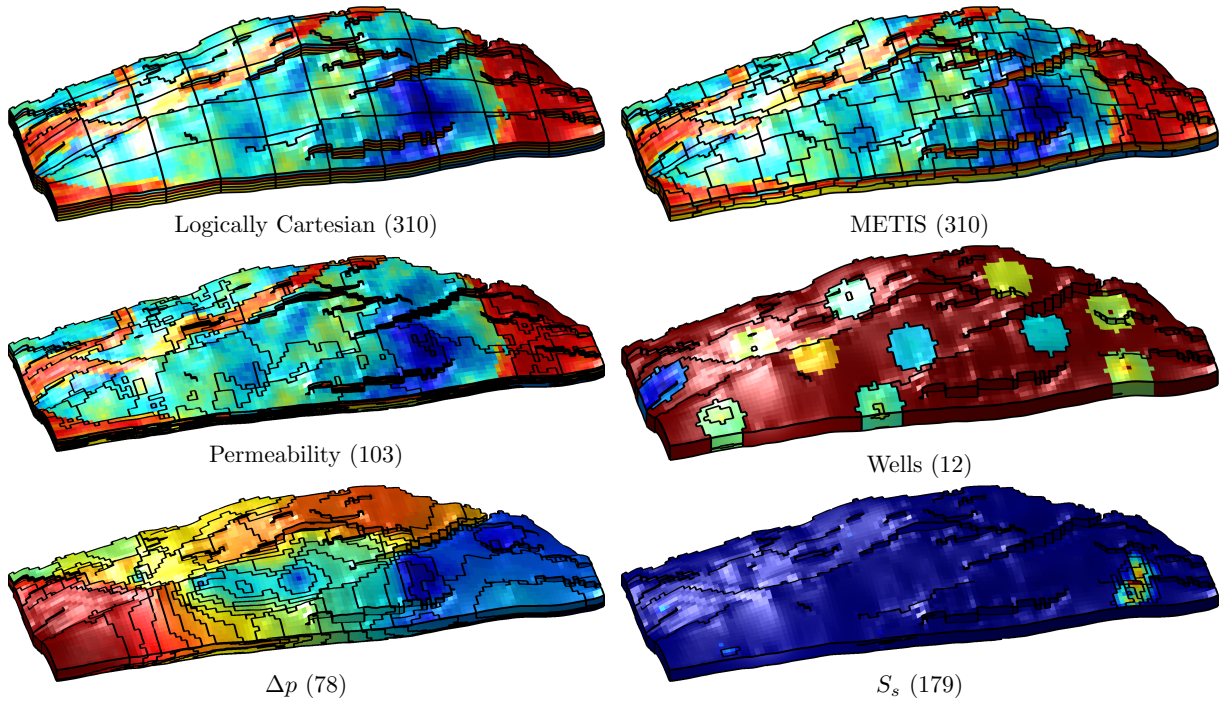


Fig. 14 The six different partitions used for the WAG scenario in Example 3. The number of coarse blocks for each partition is indicated in parentheses. Colors show the indicator used to generate each partition, except for the well partition, where we show the actual basis functions.

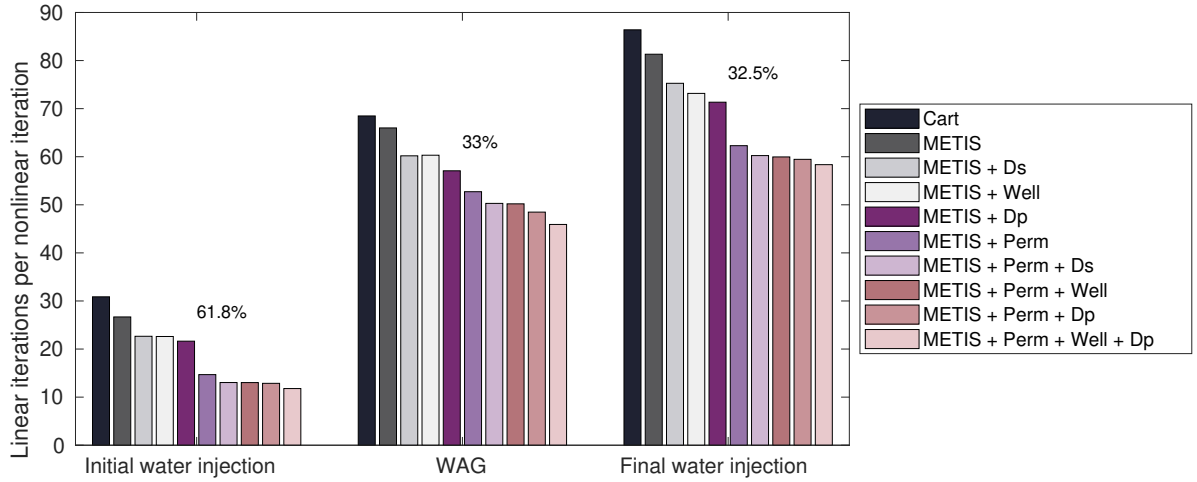


Fig. 15 Number of linear iterations per nonlinear iteration used by the different CPR solvers during each of the three production stages in Example 3. Percentages indicate the relative difference between the best and worst performing solver.

in the well control. Finally, we use two dynamic partitions agglomerated with the pressure update and the solvent gas saturation from the previous nonlinear iteration as indicators. Based on observations from the preceding examples, we only recompute the dynamic basis functions after nonlinear iteration one and two in this example. Figure 15 reports average number of linear iterations per nonlinear iteration for different combinations of multiscale operators. All solvers consume significantly more iterations during the WAG and the final water injection period. The presence of solvent gas gives denser systems, amplified by the fact that the coupling between pressure and solvent saturation increases as the solvent gas sweeps the reservoir and mixes with the resident fluids.

Using METIS to group cells with similar permeabilities gives slightly faster convergence than a Cartesian partition in index space. Adding extra static and/or dynamic partitions reduces the iterations further, and in this particular example, adapting the static partition

to permeability is the most effective. Trends are similar during all three stages of the production period, but the relative differences between the solvers are smaller during WAG and final water injection than during the initial water injection.

6.4 Computational efficiency

To assess the computational costs of the methods discussed above, we compare the theoretical number of operations required for one full linear iteration. From Equation (11), we see that this involves the following steps:

Smooth: We use incomplete LU-factorization with zero fill-in (ILU(0)) as our smoother. With one pressure unknown for each of the N cells, the smoothing step consists of solving two triangular systems and consumes $\mathcal{O}(2Nd)$ operations, where $d \ll N$ is an upper bound on the number of nonzero elements in each row of $\mathbf{A}$. Computing the linearized residual, $\mathbf{q} - \mathbf{A}\mathbf{x}$, amounts to $Nd + N$ operations for the matrix-vector product and the subtraction. Updating the state vector $\mathbf{x}$ adds another N operations.

Restrict: This amounts to computing $\mathbf{d}_\ell = \mathbf{R}^\ell(\mathbf{q}-\mathbf{A}\mathbf{x})$. The associated cost is $\mathcal{O}(Nd + N)$ operations to compute the linear residual and $\mathcal{O}(b_\ell N)$ operations for multiplication by $\mathbf{R}^\ell$. Here, $b_\ell = 1$ for a finite-volume restriction operator, whereas $1 < b_\ell \ll M_\ell$ is the maximum number of basis functions with support in a single cell for a Galerkin restriction.

Solve for $\mathbf{p}_c$: Inverting the coarse matrix $(\mathbf{A}_c^\ell)^{-1}\mathbf{d}$ typically takes $\mathcal{O}(M_\ell^p)$ operations with $p \approx 2$. The worst case is $p = 3$ for a dense matrix inverted by direct Gaussian elimination.

Prolongate and update: Prolongating the coarse pressure back to the fine grid, $\mathbf{P}^\ell\mathbf{p}_c$, consumes $\mathcal{O}(b_\ell N)$ operations and updating the state $\mathbf{x}$ adds another N operations.

To summarize, we assume for simplicity that each of the N_p coarse partitions consists of M blocks and that all basis functions have the same maximum overlap b. Then, we have that one cycle involves the following number of operations

$$N_p\mathcal{O}\big((Nd + N + 2Nd + N) + (Nd + N + bN) + M^p + (bN + N)\big) \approx \mathcal{O}\big((4d + 2b + 4)N_pN\big)$$

if we assume that the cost of inverting the multiscale system is negligible or of order $\mathcal{O}(N)$.

Updating the residual after the predictor step involves multiplication of the approximated pressure $\Delta\mathbf{x}_p$ by the first N columns and all NN_c rows of $\mathbf{J}^*$ (in which each row will have at most d nonzero elements) and subtracting the result from the residual. All in all, this involves $\mathcal{O}(N_cNd + N_cN)$ operations. Finally, the corrector step consists of an ILU(0) smoothing step applied to the full system with NN_c unknowns for a model with N_c components. This system will also have approximately N_c as many nonzero elements on each row. Updating the result gives N_cN more operations. Altogether, this amounts to a total of $\mathcal{O}\left(2dN_c^2N\right) + N_cN$ operations.

To sum up, given that we perform only one cycle in the iterative multiscale multibasis solver, the cost of one linear iteration using N_p multiscale partitions is approximately

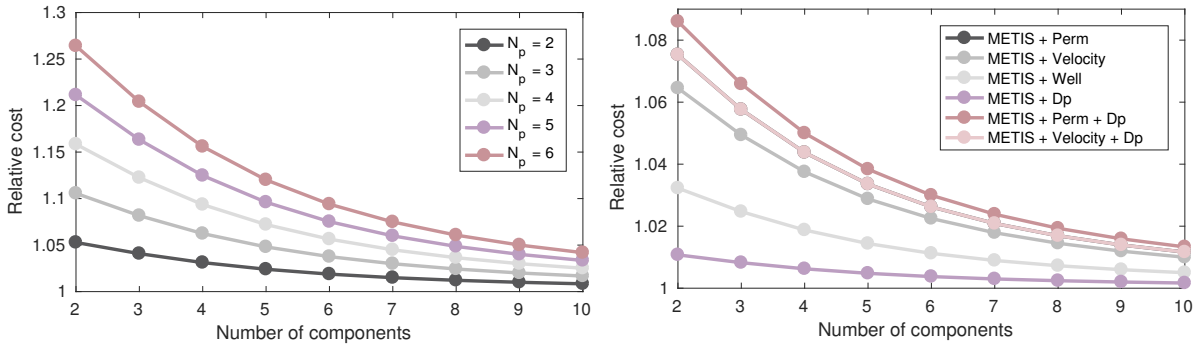
$$c(N_p) = N\big[N_p(4d + 2b + 4) + N_c(d + 1) + N_c(2dN_c + 1)\big]. \tag{12}$$

Figure 16 exemplifies the complexity analysis by reporting the ratio between the cost $c(N_p)$ of using N_p different multiscale partitions and the cost $c(1)$ of using only one multiscale partition for two different cases. In the first case, we assume that all N_p coarse partitions have the same maximum overlap b, and that $d = 15$, which is not uncommon for a realistic reservoir model with many faults and other types of non-neighboring connections. The second case uses numbers from Example 2 discussed above. We see that the worst-case scenario consists of problems with few components (primary unknowns per cell) and highly unstructured grids with many cell neighbors. As expected, we also see that the relative cost of using a multibasis CPR preconditioner diminishes with increasing number of components.

7 Concluding remarks

We have reported a preliminary study of how multiple operators that adapt to static and/or dynamic features can improve the efficiency of multiscale-CPR preconditioning in fully implicit simulators. A number of numerical experiments show that one can reduce the total number of linear iterations required to solve the full discrete system by applying extra multiscale operators to the reduced pressure system.

Adapted multiscale operators typically have few degrees of freedom relative to operators corresponding to general partitions. Applying one more iteration with such an operator to the reduced pressure system is therefore inexpensive compared with the cost of a linear iteration for the full system. From our simple analysis of the operational count, we conclude that the additional cost of using multiple bases for a typical scenario should be at most 10–15%.



Theoretical: equal overlap $b = d = 15$ Actual numbers from Example 2 (Layer 52, SPE10)

Fig. 16 The relative added computational cost for a full linear step using multiple instead of a single multiscale basis in the approximate CPR preconditioner reported as function of the number of components, i.e., primary unknowns per cell.

Basis functions can be adapted and combined in many different ways, and our results are somewhat inconclusive with respect to which specific combination is the most effective. In general, we expect that the best choice of multiscale operator(s) will vary from case to case and depend on the reservoir geology (level and type of heterogeneity), the drive mechanisms, and the degree of coupling between pressure and the other variables. Our experiments nevertheless give strong indication that it is beneficial to add static partitions honoring large permeability contrasts and/or add basis functions that better resolve near-well regions. Likewise, it also seems beneficial to add dynamic partitions adapting to pressure updates whenever these are located along propagating fluid fronts and/or fluid phase interfaces. Our overall conclusion is that using a multibasis CPR preconditioner may give a 10–60% speedup compared with using a multiscale-CPR preconditioner with only a single set of basis functions.

Finally, we note that the idea of using dynamic basis functions may be beneficial in a predictor-corrector solution framework where the system is first solved sequentially in a predictor step, and then corrected by solving the fully implicit system using the sequential solution as an initial guess, see e.g., [30]. In this case, the sequential pressure solution will typically be close to the fully implicit solution, and dynamic basis functions based on this pressure may be very well suited for a multiscale pressure solver.

8 Acknowledgements

The work of Klemetsdal and Lie is funded by the Research Council of Norway through grant no. 244361. Møyner is funded by VISTA, a basic research program funded by Equinor and conducted in close collaboration with the Norwegian Academy of Science and Letters.

References

1. J. E. Aarnes, S. Krogstad, and K.-A. Lie. A hierarchical multiscale method for two-phase flow based upon mixed finite elements and nonuniform coarse grids. *Multiscale Modeling & Simulation*, 5(2):337–363, 2006. doi: 10.1137/050634566.
2. R. E. Bank, T. F. Chan, W. M. Coughran, and R. K. Smith. The alternate-block-factorization procedure for systems of partial differential equations. *BIT*, 29(4):938–954, Dec 1989. doi: 10.1007/BF01932753.
3. K. Bao, K.-A. Lie, O. Møyner, and M. Liu. Fully implicit simulation of polymer flooding with MRST. *Comput. Geosci.*, 21(5):1219–1244, Dec 2017. doi: 10.1007/s10596-017-9624-5.
4. J. B. Bell, J. A. Trangenstein, and G. R. Shubin. Conservation laws of mixed type describing three-phase flow in porous media. *SIAM J. Appl. Math.*, 46(6):1000–1017, 1986. doi: 10.1137/0146059.
5. M. A. Christie and M. J. Blunt. Tenth SPE comparative solution project: a comparison of upscaling techniques. *SPE Reservoir Eval. Eng.*, 4:308–317, 2001. doi: 10.2118/72469-PA.
6. K. H. Coats. A note on impes and some impes-based simulation models. *SPE J.*, 5:14–17, 2000. doi: 10.2118/65092-PA.
7. M. Cusini, A. A. Lukyanov, J. R. Natvig, and H. Hajibeygi. Constrained pressure residual multiscale (CPR-MS) method for fully implicit simulation of multiphase flow in porous media. *J. Comput. Phys.*, 299:472–486, 2015. doi: 10.1016/j.jcp.2015.07.019.
8. Y. Efendiev and T. Y. Hou. *Multiscale Finite Element Methods*, volume 4 of *Surveys and Tutorials in the Applied Mathematical Sciences*. Springer Verlag, New York, 2009.
9. GeoQuest, Schlumberger. *ECLIPSE reservoir simulator, Manual and technical description*, 2010.
10. S. Gries, K. Stüben, G. L. Brown, D. Chen, and D. A. Collins. Preconditioning for efficiently applying algebraic multigrid in fully implicit reservoir simulations. *SPE J.*, 19(04):726–736, 2014. doi: 10.2118/163608-PA.
11. H. Hajibeygi, G. Bonfigli, M. A. Hesse, and P. Jenny. Iterative multiscale finite-volume method. *J. Comput. Phys.*, 227(19):8604–8621, 2008. doi: 10.1016/j.jcp.2008.06.013.
12. V. L. Hauge. *Multiscale methods and flow-based gridding for flow and transport in porous media*. PhD thesis,

Norwegian University of Science and Technology, 2010.
13. V. L. Hauge, K.-A. Lie, and J. R. Natvig. Flow-based coarsening for multiscale simulation of transport in porous media. *Comput. Geosc.*, 16(2):391–408, 2012. doi: 10.1007/s10596-011-9230-x.
14. T. Y. Hou and X.-H. Wu. A multiscale finite element method for elliptic problems in composite materials and porous media. *J. Comput. Phys.*, 134(1):169–189, 1997. doi: 10.1006/jcph.1997.5682.
15. P. Jenny, S. H. Lee, and H. A. Tchelepi. Multi-scale finite-volume method for elliptic problems in subsurface flow simulation. *J. Comput. Phys.*, 187(1):47–67, 2003. doi: 10.1016/S0021-9991(03)00075-5.
16. P. Jenny, S. H. Lee, and H. A. Tchelepi. Adaptive fully implicit multi-scale finite-volume method for multi-phase flow and transport in heterogeneous porous media. *J. Comput. Phys.*, 217(2):627–641, 2006. doi: 10.1016/j.jcp.2006.01.028.
17. G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392, 1998. doi: 10.1137/S1064827595287997.
18. A. Kozlova, Z. Li, J. R. Natvig, S. Watanabe, Y. Zhou, K. Bratvedt, and S. H. Lee. A real-field multiscale black-oil reservoir simulator. *SPE J.*, 21(6):2049–2061, 2016. doi: 10.2118/173226-PA.
19. A. Kozlova, D. Walsh, S. Chittireddy, Z. Li, J. Natvig, S. Watanabe, and K. Bratvedt. A hybrid approach to parallel multiscale reservoir simulator. In *ECMOR XV – 15th European Conference on the Mathematics of Oil Recovery*, Amsterdam, The Netherlands, 29 August–1 September 2016. EAGE. doi: 10.3997/2214-4609.201601889.
20. S. Krogstad, K.-A. Lie, O. Møyner, H. M. Nilsen, X. Raynaud, and B. Skaflestad. MRST-AD – an open-source framework for rapid prototyping and evaluation of reservoir simulation problems. In *SPE Reservoir Simulation Symposium*, 2015. doi: 10.2118/173317-MS.
21. S. Lacroix, Y. V. Vassilevski, and M. F. Wheeler. Iterative solvers of the implicit parallel accurate reservoir simulator (IPARS), i: Single processor case. Technical report, Numer. Linear Algebra Appl., 2000.
22. S. Lacroix, Y. Vassilevski, J. Wheeler, and M. Wheeler. Iterative solution methods for modeling multiphase flow in porous media fully implicitly. *SIAM J. Sci. Comput.*, 25(3):905–926, 2003. doi: 10.1137/S106482750240443X.
23. K.-A. Lie. *An Introduction to Reservoir Simulation Using MATLAB: User guide for the MATLAB Reservoir Simulation Toolbox (MRST)*. Cambridge University Press, 2019.
24. K.-A. Lie, K. Kedia, B. Skaflestad, X. Wang, Y. Yang, X.-H. Wu, and N. Hoda. A general non-uniform coarsening and upscaling framework for reduced-order modeling. In *SPE Reservoir Simulation Conference*. Society of Petroleum Engineers, February 2017. doi: 10.2118/182681-MS.
25. K.-A. Lie, O. Møyner, and J. R. Natvig. A feature-enriched multiscale method for simulating complex geomodels. *SPE J.*, 22(06):1929–1945, 2017. doi: 10.2118/182701-PA.
26. K.-A. Lie, O. Møyner, J. R. Natvig, A. Kozlova, K. Bratvedt, S. Watanabe, and Z. Li. Successful application of multiscale methods in a real reservoir simulator environment. *Comput. Geosci.*, 21(5):981–998, 2017. doi: 10.1007/s10596-017-9627-2.
27. T. Manzocchi et al. Sensitivity of the impact of geological uncertainty on production from faulted and unfaulted shallow-marine oil reservoirs: objectives and methods. *Petrol. Geosci.*, 14(1):3–15, 2008. doi: 10.1144/1354-079307-790.
28. O. Møyner and K.-A. Lie. A multiscale restriction-smoothed basis method for compressible black-oil models. *SPE J.*, 21(6):2079–2096, 2016. doi: 10.2118/173265-PA.
29. O. Møyner and K. A. Lie. A multiscale restriction-smoothed basis method for high contrast porous media represented on unstructured grids. *J. Comput. Phys.*, 304:46–71, 2016. doi: 10.1016/j.jcp.2015.10.010.
30. O. Møyner and A. Moncorgé. Nonlinear domain decomposition scheme for sequential fully implicit formulation of compositional multiphase flow. In *ECMOR XVI – 16th European Conference on the Mathematics of Oil Recovery*, Barcelona, Spain, September 3–6, 2018 2018. EAGE. doi: 10.3997/2214-4609.201802128.
31. O. Møyner and H. A. Tchelepi. A mass-conservative sequential implicit multiscale method for general compositional problems. *SPE J.*, 23(06):2376–2393, 2018. doi: 10.2118/182679-PA.
32. A. S. Odeh. Comparison of solutions to a three-dimensional black-oil reservoir simulation problem. *J. Petrol. Techn.*, 33(1):13–25, 1981. doi: 10.2118/9723-PA.
33. Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, 1986. doi: 10.1137/0907058.
34. S. M. Sheth and R. M. Younis. Localized solvers for general full-resolution implicit reservoir simulation. In *SPE Reservoir Simulation Conference*. Society of Petroleum Engineers, 2017. doi: 10.2118/182691-MS.
35. K. Stüben, T. Clees, H. Klie, B. Lu, and M. F. Wheeler. Algebraic multigrid methods (AMG) for the efficient solution of fully implicit formulations in reservoir simulation. In *SPE Reservoir Simulation Conference*. Society of Petroleum Engineers, February 2007. doi: 10.2118/105832-MS.
36. M. R. Todd and W. J. Longstaff. The development, testing, and application of a numerical simulator for predicting miscible flood performance. *J. Petrol. Tech.*, 24(7):874–882, 1972. doi: 10.2118/3484-PA.
37. J. A. Trangenstein and J. B. Bell. Mathematical structure of the black-oil model for petroleum reservoir simulation. *SIAM J. Appl. Math.*, 49(3):749–783, 1989. doi: 10.1137/0149044.
38. U. Trottenberg, C. W. Oosterlee, and A. Schuller. *Multigrid*. Academic press, 2000.
39. J. R. Wallis. Incomplete Gaussian elimination as a preconditioning for generalized conjugate gradient acceleration. In *SPE Reservoir Simulation Symposium*, 1983. doi: 10.2118/12265-MS.
40. J. R. Wallis, R. P. Kendall, and T. E. Little. Constrained residual acceleration of conjugate residual methods. In *SPE Reservoir Simulation Symposium*, 1985. doi: 10.2118/13536-MS.
41. Y. Wang, H. Hajibeygi, and H. A. Tchelepi. Algebraic multiscale solver for flow in heterogeneous porous media. *J. Comput. Phys.*, 259:284–303, 2014. doi: 10.1016/j.jcp.2013.11.024.
42. J. Watts. A compositional formulation of the pressure and saturation equations. *SPE Reserv. Eng.*, 1(03):243–252, 1986. doi: 10.2118/12244-PA.