Computational Geosciences (2022) 26:1045–1063
https://doi.org/10.1007/s10596-021-10090-x

ORIGINAL PAPER

A numerical study of the additive Schwarz preconditioned exact Newton method (ASPEN) as a nonlinear preconditioner for immiscible and compositional porous media flow

Øystein Klemetsdal[1] · Arthur Moncorgé[2] · Olav Møyner[1] · Knut-Andreas Lie[1]

Received: 7 January 2021 / Accepted: 19 August 2021 / Published online: 16 September 2021

Abstract

Domain decomposition methods are widely used as preconditioners for Krylov subspace linear solvers. In the simulation of porous media flow there has recently been a growing interest in nonlinear preconditioning methods for Newton's method. In this work, we perform a numerical study of a spatial additive Schwarz preconditioned exact Newton (ASPEN) method as a nonlinear preconditioner for Newton's method applied to both fully implicit or sequential implicit schemes for simulating immiscible and compositional multiphase flow. We first review the ASPEN method and discuss how the resulting linearized global equations can be recast so that one can use standard preconditioners developed for the underlying model equations. We observe that the local fully implicit or sequential implicit updates efficiently handle the local nonlinearities, whereas long-range interactions are resolved by the global ASPEN update. The combination of the two updates leads to a very competitive algorithm. We illustrate the behavior of the algorithm for conceptual one and two-dimensional cases, as well as realistic three dimensional models. A complexity analysis demonstrates that Newton's method with a fully implicit scheme preconditioned by ASPEN is a very robust and scalable alternative to the well-established Newton's method for fully implicit schemes.

Keywords Nonlinear solvers · Nonlinear preconditioning · ASPEN · Reservoir simulation

1 Introduction

Simulation of multiphase and multicomponent flow has many applications within the subsurface sciences, including oil and gas recovery, carbon dioxide sequestration, groundwater and subsurface remediation, and geothermal energy management. The basic flow models consist of conservation equations for each species where the fluid velocity is given by Darcy's law, which together form a nonlinear system of parabolic partial differential equations. This system will usually have a mixed elliptic–hyperbolic sub-character and generally describe delicate balances among capillary, gravity, and viscous forces that vary over time and throughout space. If you combine this with orders of magnitude multiscale variations in rock properties, grids with skew geometries and large aspect ratios, nonsmooth and hysteretic rock–fluid properties, and large variations in temporal and spatial scales, you end up with numerical models that are surprisingly challenging to simulate efficiently.

The standard approach in reservoir simulation (and in simulation of carbon storage) is to use a fully implicit discretization and iteratively solve for all the primary unknowns at once using Newton's method. This requires repeated solves of large, ill-conditioned linearized systems of equations, for which domain-decomposition methods are well established as a means to accelerate the solution process [18]. These techniques can be categorized into methods for variable decomposition and spatial decomposition. State-of-the-art simulators use so-called constrained

✉ Øystein Klemetsdal
oystein.klemetsdal@sintef.no

Arthur Moncorgé
arthur.moncorge@totalenergies.com

Olav Møyner
olav.moyner@sintef.no

Knut-Andreas Lie
knut-andreas.lie@sintef.no

[1] SINTEF Digital, Oslo, Norway

[2] TotalEnergies, Pau, France

pressure residual (CPR) preconditioners [5, 24, 48, 49], which fall into the first category and are designed to exploit the mixed elliptic–hyperbolic character of the system by first solving for a pressure-like variable by an efficient algebraic multigrid (AMG) preconditioner [10, 12, 45, 46], followed by a broadband smoother like incomplete lower-upper factorization with zero fill-in (ILU0) applied to the whole system. Standard domain additive and multiplicative Schwarz decomposition methods have been used for decades as a parallelization strategy [6, 16, 18, 24]. In addition, recent multiscale methods (see [27] for an overview) can be interpreted as domain decomposition methods [40]).

Domain decomposition is also very appealing as a nonlinear solution strategy in reservoir simulation: The system of nonlinear equations tends to be strongly coupled, highly nonlinear, and unbalanced (i.e., a safe step length for Newton's method is determined by a small subset of the full variable set [3]), so that unless we take very short timesteps, the initial guess is typically far from the solution. A particularly popular nonlinear variable-decomposition approach is sequential splitting of the full problem into a pressure subproblem and a set of transport subproblems [33, 34, 38, 47, 50], which also facilitates using efficient multiscale methods for the pressure subproblem [13, 27, 36].

Nonlinear spatial decomposition methods, on the other hand, have so far seen limited use. A main reason for this could be that even though nonlinear spatial domain decomposition is excellent at handling unbalanced nonlinearities, it tends to converge slowly for the full residual. Cai and Keyes [3] therefore proposed to use domain decomposition as a nonlinear preconditioner and devised an additive Schwarz preconditioned inexact Newton (ASPIN) method. The method has later been applied to a range of flow problems, including single-phase Navier–Stokes cavity flow [4], two-phase porous media flow [43], and single-phase Forchheimer flow with a restricted, exact Newton version (RASPEN) [9]. Skogestad et al. [43] also demonstrated that the performance of ASPIN is better than that of the linear preconditioning counterpart. Moreover, Liu et al. [30] applied ASPIN with variable decomposition to sequential splitting for two-phase porous media flow, and later also devised a multiplicative (MSPIN) method [29, 31]. Other recent work on variable-decomposition MSPIN for subsurface flow includes geothermal applications [51] and simulations with multisegment wells [25].

Herein, we study preconditioning of the fully implicit method with the additive Schwarz preconditioned (in)exact Newton (ASPEN/ASPIN) method on a variety of flow problems in reservoir simulation, including both compressible black-oil type and compositional models. In the following, we will for simplicity refer to both methods as ASPEN. These preconditioners can be seen as a two-step algorithm. During the first step, we converge all the local subdomains independently with an additive Schwarz algorithm. For each subdomain, we freeze the values outside of the subdomain and solve the fully implicit equations with Newton's method locally. When all variables have been updated, the residual of the equations is zero inside each domain, whereas nonzero values can be observed at the boundaries between the subdomains. The second step of the ASPEN algorithm is to minimize a global function with Newton's method. We show that the local fully implicit solves accurately account for all local couplings, that the global ASPEN solves resolve the global interactions well, and that the combination of these two updates leads to a very robust and scalable alternative to Newton's method when applied to fully implicit schemes.

The paper is organized as follows: Section 2 describes the model, Section 3 reviews the ASPEN framework, and in Section 4 we report and discuss the results of a series of numerical experiments.

2 Governing equations

We consider a generic set of flow equations for a multiphase, multicomponent system that has been discretized in time by a finite-difference method and in space by a finite-volume scheme

$$\mathbf{R}_i^{n+1} = \frac{1}{\Delta t^n}(\mathbf{M}_i^{n+1} - \mathbf{M}_i^n) + \mathtt{div}(\mathbf{V}_i^{n+1}) - \mathbf{Q}_i^{n+1} = 0, \quad i = 1, ..., m. \quad (1)$$

Here, $\mathbf{M}_i$ is the vector containing the conserved quantity of component i in all the grid cells, $\mathbf{V}_i$ is the vector of flow rates for component i across each cell interface, and $\mathbf{Q}_i$ is the vector of source terms. Superscript n refers to the timestep and `div` is a discrete analogue of the standard divergence operator defined over a volumetric grid that maps face values to cells; see, e.g., [26, §4.4] for details on how this operator is defined. We do not make any special assumptions on the grid, except for assuming that it consists of a finite collection of polytopal cells with matching faces.

We use a standard Peaceman-type model to relate the effective source terms $\mathbf{Q}_i$ in each perforated cell to the difference between the wellbore pressure and average cell pressure. This involves solving one extra conservation equation for each of the well phase fluxes. Well controls on bottom-hole pressure or fluid rates are (weakly) enforced by an additional control equation. The detailed formulation used herein is described in detail in [26, §4.3.2 and 12.2.4].

By collecting the individual equations in Eq. 1, we obtain a nonlinear system of discrete residual equations, $\mathbf{R}(\mathbf{u}) = \mathbf{0}$. This system is typically solved using Newton's method: We

assume that for a value $\mathbf{u}$ there exists some update $\Delta\mathbf{u}$ so that $\mathbf{u}+\Delta\mathbf{u}$ solves the residual equation and linearize around $\mathbf{u}$,

$$\mathbf{0} = \mathbf{R}(\mathbf{u}+\Delta\mathbf{u}) = \mathbf{R}(\mathbf{u}) + \tfrac{\partial\mathbf{R}}{\partial\mathbf{u}}\Delta\mathbf{u} + \mathcal{O}(\|\Delta\mathbf{u}\|^2). \tag{2}$$

Neglecting higher-order terms and solving for $\Delta\mathbf{u}$ gives us the iterative Newton's method in its basic form

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta\mathbf{u}, \quad -\frac{\partial\mathbf{R}}{\partial\mathbf{u}}\Delta\mathbf{u} = \mathbf{R}(\mathbf{u}^k). \tag{3}$$

In most simulators, the Newton increment $\Delta\mathbf{u}$ is not applied directly but is modified to ensure that the updated state is physically meaningful, and at the same time aiming at stable convergence. One method to this end is to let $\Delta\mathbf{u}$ define the search direction for an inexact line-search method or for more sophisticated trust-region approaches [14, 21, 35]. Likewise, we typically need to precondition the linear system made up of the Jacobian and residual in order to solve it efficiently. To keep the presentation as simple as possible, we disregard these issues in the following as the specific choices are not essential to the ASPEN solver. Before continuing to discuss the ASPEN framework, we briefly outline some typical examples of Eq. 1.

2.1 Immiscible flow

In subsurface flow, we typically consider three phases: aqueous (a), liquid (ℓ), and vapor (v). For immiscible flow, each phase is assumed to consist of one component only and Eq. 1 thus describes conservation of mass for each phase. The system of residual equations is formulated using

$$M_\alpha = \phi\rho_\alpha S_\alpha, \qquad V_\alpha = \rho_\alpha v_\alpha, \qquad Q_\alpha = \rho_\alpha q_\alpha, \tag{4}$$

where $\alpha = a, \ell, v$. Here, ϕ is porosity, ρ_α denotes densities, S_α is saturation, v_α is phase flow rate, and q_α represents volumetric sources of phase α. To close the system, we need relationships for the flow rates v_α, saturations S_α, relative permeabilities $k_{r\alpha}$, and phase pressure p_α

$$\begin{aligned} v_\alpha &= -\frac{k_{r\alpha}}{\mu_\alpha}\mathsf{K}\big(\mathtt{grad}\,(p_\alpha) - \rho_\alpha g\,\mathtt{grad}\,(z)\big), \\ S_a + S_\ell + S_v = 1, &\quad p_\ell - p_a = p_c^a(S_a), \\ k_{r\alpha} = k_{r\alpha}(S_\alpha), &\quad p_v - p_\ell = p_c^v(S_v). \end{aligned} \tag{5}$$

Here, K is the absolute permeability tensor, $\mathtt{grad}$ is a discrete gradient, μ_α is viscosity, ρ_α is density, g is the gravity acceleration, z is the vertical coordinate, and p_c^α are capillary-pressure curves.

2.2 Compositional flow

In compositional models, we use mass fractions $x_{\alpha,i}$ to account for the amount of component i contained in phase α. In this case, Eq. 1 represents mass balances for each component, so that

$$\begin{aligned} M_i &= \sum_{\alpha=a,\ell,v} M_\alpha x_{\alpha,i}, \\ V_i &= \sum_{\alpha=a,\ell,v} V_\alpha x_{\alpha,i}, \qquad \text{where} \quad \sum_{i=1}^{m} x_{\alpha,i} = 1. \\ Q_i &= \sum_{\alpha=a,\ell,v} Q_\alpha x_{\alpha,i}, \end{aligned} \tag{6}$$

Note that the closure relationships (5) are expanded by the requirement that the mass fractions must sum to unity for each phase α. In addition, the partition of the components between the phases is determined by an equation-of-state; see, e.g., [41, 42, 44]. We assume a simple water treatment where the aqueous phase is synonymous with the water component and the liquid and vapor phases can contain any combination of the non-water components.

2.3 Sequential splitting

A fully implicit method solves the full nonlinear system of residual (1), with Eqs. 4–5 for immiscible flow and Eqs. 4–6 for compositional flow, for all unknowns simultaneously in each timestep. In a sequential implicit method, we perform a variable decomposition of $\mathbf{u}$ into flow variables $\mathbf{u}_p$ (pressure) and transport variables $\mathbf{u}_t$ (saturations and/or mass fractions) and then use this to formulate a flow equation and a (system of) transport equation(s). These are then solved implicitly in consecutive steps. The flow equation is derived as a weighted sum of the component conservation (1),

$$\mathbf{R}_p^{n+1} = \sum_{i=1}^{m} \omega_i \mathbf{R}_i^{n+1} = \mathbf{0}. \tag{7}$$

The weights ω_i are chosen so that all partial derivatives of the resulting equation with respect to transport variables disappear; i.e., $\sum_i \partial_v(\omega_i \mathbf{R}_i^{n+1}) = 0$ for all $v \in \mathbf{u}_t$. The transport equation, $\mathbf{R}_t^{n+1} = \mathbf{0}$, consists of the original component conservation equations, but with fixed pressure variables and the flux (V_α or V_i) formulated in terms of the total velocity; see [38] for details.

3 ASPEN framework

We want to find the solution $\mathbf{u}$ to the system of nonlinear residual equations $\mathbf{R}(\mathbf{u}) = \mathbf{0}$, where $\mathbf{u}$ and $\mathbf{R}(\mathbf{u})$ are vectors of real numbers. In the following, $\mathbf{R}$ will be one of three possible types of equations: (i) the full system of residual component (1), which has a mixed elliptic–hyperbolic character; (ii) the flow equation with $\mathbf{R} = \mathbf{R}_p^{n+1}$,

which is parabolic but with an elliptic character; and (iii) the transport equation with $\mathbf{R} = \mathbf{R}_t^{n+1}$, which is either hyperbolic or parabolic with a strong hyperbolic character. The additive Schwarz method for solving such problems is generally not very robust by itself, but may be a very efficient nonlinear preconditioner.

3.1 Brief review of the method

To explain the method, we first partition the full problem into two subproblems. Without loss of generality, we assume that variables and equations are ordered so that this can be written on the form

$$\mathbf{R}(\mathbf{u}) = (\mathbf{R}_1(\mathbf{u}_1, \mathbf{u}_2), \mathbf{R}_2(\mathbf{u}_1, \mathbf{u}_2)) = \mathbf{0}, \tag{8}$$

were $\mathbf{u} = (\mathbf{u}_1, \mathbf{u}_2)$ is a partition of the unknowns into two non-overlapping subdomains with corresponding residual equations $\mathbf{R}_1$ and $\mathbf{R}_2$ that have their accumulation terms defined in the same non-overlapping partitions. In an additive Schwarz method, these are solved independently while keeping the unknowns in the other domain fixed. Formally, we introduce the following solution operator

$$\mathscr{L}(\mathbf{u}) = (\mathscr{L}_1(\mathbf{u}), \mathscr{L}_2(\mathbf{u}))\,, \tag{9}$$

where $\mathscr{L}_i(\mathbf{u})$ is defined as the solution to subproblem i,

$$\mathbf{R}_1\big(\mathscr{L}_1(\mathbf{u}), \mathbf{u}_2\big) = \mathbf{0}, \quad \text{and} \quad \mathbf{R}_2\big(\mathbf{u}_1, \mathscr{L}_2(\mathbf{u})\big) = \mathbf{0}. \tag{10}$$

Solving $\mathbf{R}(\mathbf{u}) = \mathbf{0}$ is then equivalent to finding the fixed point $\mathbf{u}$ such that $\mathbf{u} = \mathscr{L}(\mathbf{u})$. This fixed-point iteration scheme tends to have poor convergence properties, but can be accelerated by means of nonlinear acceleration techniques; Jiang and Tchelepi [15] compares some of these techniques in the context of subsurface flow. One popular class of acceleration techniques are quasi-Newton methods, which reformulate the fixed-point iteration as a residual equation and use information from previous iterates to construct an approximate Jacobian. ASPEN uses a similar approach, but computes the exact Jacobian instead of an approximation. We write the fixed-point equation on residual form,

$$\mathbf{F}(\mathbf{u}) = \mathbf{u} - \mathscr{L}(\mathbf{u}) = \mathbf{0}, \tag{11}$$

and derive a Newton method equivalent to Eq. 3,

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta\mathbf{u}, \quad -\frac{\partial \mathbf{F}}{\partial \mathbf{u}}\Delta\mathbf{u} = \mathbf{F}(\mathbf{u}^k), \quad \text{where} \quad \frac{\partial \mathbf{F}}{\partial \mathbf{u}} = \mathbf{I} - \begin{bmatrix} \frac{\partial \mathscr{L}_1}{\partial \mathbf{u}} \\ \frac{\partial \mathscr{L}_2}{\partial \mathbf{u}} \end{bmatrix}. \tag{12}$$

The question is now how to compute the Jacobian $\partial\mathbf{F}/\partial\mathbf{u}$. The derivatives of the implicitly defined solution operator $\mathscr{L}$ do not have a simple closed form in the general case and cannot be directly computed by means of conventional techniques such as automatic differentiation. However, by taking the derivative of the first domain equation $\mathbf{R}_1(\mathbf{u}) = \mathbf{0}$ with respect to $\mathbf{u}$, we find that

$$\frac{\partial \mathbf{R}_1}{\partial \mathbf{u}} = \frac{\partial \mathbf{R}_1}{\partial \mathbf{u}_1}\frac{\partial \mathscr{L}_1}{\partial \mathbf{u}} + \frac{\partial \mathbf{R}_1}{\partial \mathbf{u}_2}\frac{\partial \mathbf{u}_2}{\partial \mathbf{u}} = \mathbf{0}. \tag{13}$$

Note that this is evaluated in $(\mathbf{u}_1, \mathbf{u}_2) = (\mathscr{L}_1(\mathbf{u}), \mathbf{u}_2)$. We rearrange to obtain

$$\frac{\partial \mathscr{L}_1}{\partial \mathbf{u}} = -\left(\frac{\partial \mathbf{R}_1}{\partial \mathbf{u}_1}\right)^{-1}\frac{\partial \mathbf{R}_1}{\partial \mathbf{u}_2}\frac{\partial \mathbf{u}_2}{\partial \mathbf{u}}, \tag{14}$$

and similarly,

$$\frac{\partial \mathscr{L}_2}{\partial \mathbf{u}} = -\left(\frac{\partial \mathbf{R}_2}{\partial \mathbf{u}_2}\right)^{-1}\frac{\partial \mathbf{R}_2}{\partial \mathbf{u}_1}\frac{\partial \mathbf{u}_1}{\partial \mathbf{u}}. \tag{15}$$

Here, Eq. 14 is evaluated in $(\mathbf{u}_1, \mathbf{u}_2) = (\mathscr{L}_1(\mathbf{u}), \mathbf{u}_2)$, whereas Eq. 15 is evaluated in $(\mathbf{u}_1, \mathbf{u}_2) = (\mathbf{u}_1, \mathscr{L}_2(\mathbf{u}))$. Altogether, this means that we are able to compute the Jacobian of the solution operator $\mathscr{L}$, and thereby the Jacobian of $\mathbf{F}$, using only known quantities. Moreover, part of the computation is already done, because solving the subproblems (10) necessarily requires that we compute $\partial\mathbf{R}_i/\partial\mathbf{u}_i$ (provided that we use Newton's method). The coupling terms $\partial\mathbf{R}_i/\partial\mathbf{u}_j$ must be computed after the subdomain solves. Note also that $\partial\mathbf{u}_i/\partial\mathbf{u}$ applied to a vector $\mathbf{v}$ simply acts as a restriction onto subdomain i,

$$\frac{\partial \mathbf{u}_i}{\partial \mathbf{u}}\mathbf{v} = \mathbf{e}_i^T\mathbf{v} = \mathbf{v}_i. \tag{16}$$

The method naturally extends to the case where we have m subdomains, each defined by their corresponding non-overlapping subset. For this, we define a non-overlapping partition of the full problem

$$\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_m), \quad \mathbf{R}(\mathbf{u}) = \big(\mathbf{R}_1(\mathbf{u}), \ldots, \mathbf{R}_m(\mathbf{u})\big) = \mathbf{0},$$

with corresponding solution operators $\mathscr{L}_1, \ldots, \mathscr{L}_m$ such that

$$\mathbf{R}_i\big(\mathbf{u}_1, \ldots, \mathbf{u}_{i-1}, \mathscr{L}_i(\mathbf{u}), \mathbf{u}_{i+1}, \ldots, \mathbf{u}_m\big) = \mathbf{0}. \tag{17}$$

The expression for the Jacobian of each solution operator is analogous to the two-subdomain case and is found in the same manner:

$$\frac{\partial \mathscr{L}_i}{\partial \mathbf{u}} = -\left(\frac{\partial \mathbf{R}_i}{\partial \mathbf{u}_i}\right)^{-1}\left(\sum_{j=1, j\neq i}^{m}\frac{\partial \mathbf{R}_i}{\partial \mathbf{u}_j}\frac{\partial \mathbf{u}_j}{\partial \mathbf{u}}\right). \tag{18}$$

Wells often constitute the parts of the model equations that have the strongest coupling and thus represent a primary hindrance to nonlinear convergence. Equilibration within the wellbore is usually assumed to be (almost) instant. In our experience, it is thus important that all the variables of a single well, or all well variables that are subject to the same group or field control, belong to the same subdomain. Grouping variables from wells that are far apart into a single subdomain does not pose any conceptual difficulties for ASPEN, because the method does not require spatially contiguous subdomains.

The near-well region usually contains a relatively large pressure drop and tends to exhibit strongly coupled and nonlinear behavior. We therefore group all well variables *and* the reservoir variables from a region surrounding each well into a single subdomain. By default, the surrounding region is set to be the perforated cells plus a padding layer, defined so that each perforated cell has at least one topological neighbor between itself and the nearest subdomain boundary, but in some examples we use somewhat larger domains. Using at least one level of cell padding also means that the formulation discussed in the previous section does not need any modification to account for extra coupling terms that potentially could have been introduced by wells, because the residual well equations only have nonzero derivatives with respect to reservoir variables from the perforated cells. Subdomains must be merged if they overlap in space or if the wells are subject to a common control, which is not the case for any of the examples considered herein.

3.2 Some properties of the linearized system

In contrast to the Jacobian $\partial\mathbf{R}/\partial\mathbf{u}$ of the original problem, the Jacobian $\partial\mathbf{F}/\partial\mathbf{u}$ is generally dense, and efficient preconditioners are therefore challenging to construct. However, a breakdown of the blocks in Eq. 14 reveals that

$$\frac{\partial\mathscr{L}_1}{\partial\mathbf{u}_1} = \mathbf{0}, \quad \frac{\partial\mathscr{L}_1}{\partial\mathbf{u}_2} = -\left(\frac{\partial\mathbf{R}_1}{\partial\mathbf{u}_1}\right)^{-1}\frac{\partial\mathbf{R}_1}{\partial\mathbf{u}_2}, \tag{19}$$

and similarly for the second subdomain. The first block, $\partial\mathscr{L}_1/\partial\mathbf{u}_1$, is zero because $\mathscr{L}_1$ gives the solution in subdomain one regardless of the initial guess $\mathbf{u}_1$. However, this solution will obviously depend on the boundary conditions imposed from the constant values $\mathbf{u}_2$ in subdomain two, and consequently, $\partial\mathscr{L}_1/\partial\mathbf{u}_2$ is generally nonzero. Looking at Eq. 12, it follows that the diagonal blocks of $\partial\mathbf{F}/\partial\mathbf{u}$ are simply the identity, and

$$\frac{\partial\mathbf{F}}{\partial\mathbf{u}} = \begin{bmatrix} \frac{\partial\mathbf{R}_1}{\partial\mathbf{u}_1} & \mathbf{0} \\ \mathbf{0} & \frac{\partial\mathbf{R}_2}{\partial\mathbf{u}_2} \end{bmatrix}^{-1} \frac{\partial\mathbf{R}}{\partial\mathbf{u}} \equiv \mathbf{D}^{-1}\frac{\partial\mathbf{R}}{\partial\mathbf{u}}. \tag{20}$$

In this linear system, $\partial\mathbf{R}_1/\partial\mathbf{u}_1$ and the rows of $\partial\mathbf{R}/\partial\mathbf{u}$ corresponding to the residual equation in subdomain one are evaluated in $\mathbf{u} = (\mathscr{L}_1(\mathbf{u}), \mathbf{u}_2)$, whereas $\partial\mathbf{R}_2/\partial\mathbf{u}_2$ and the rows of $\partial\mathbf{R}/\partial\mathbf{u}$ corresponding to subdomain two are evaluated in $\mathbf{u} = (\mathbf{u}_1, \mathscr{L}_2(\mathbf{u}))$. This formula enables us to interpret the linearized system as

$$-\frac{\partial\mathbf{F}}{\partial\mathbf{u}}\Delta\mathbf{u} = \mathbf{F}(\mathbf{u}) \quad \Longleftrightarrow \quad -\frac{\partial\mathbf{R}}{\partial\mathbf{u}}\Delta\mathbf{u} = \mathbf{DF}(\mathbf{u}). \tag{21}$$

This is just the usual linearization (3) of the original problem without preconditioning, but with a different right-hand side. In other words, this interpretation brings us back to home ground, where we know very well what linear preconditioner to use. In the examples reported shortly, the increments $\Delta\mathbf{u}$ are either computed with a direct or an iterative linear solver, depending on the problem size. We nonetheless refer to the resulting method as ASPEN,

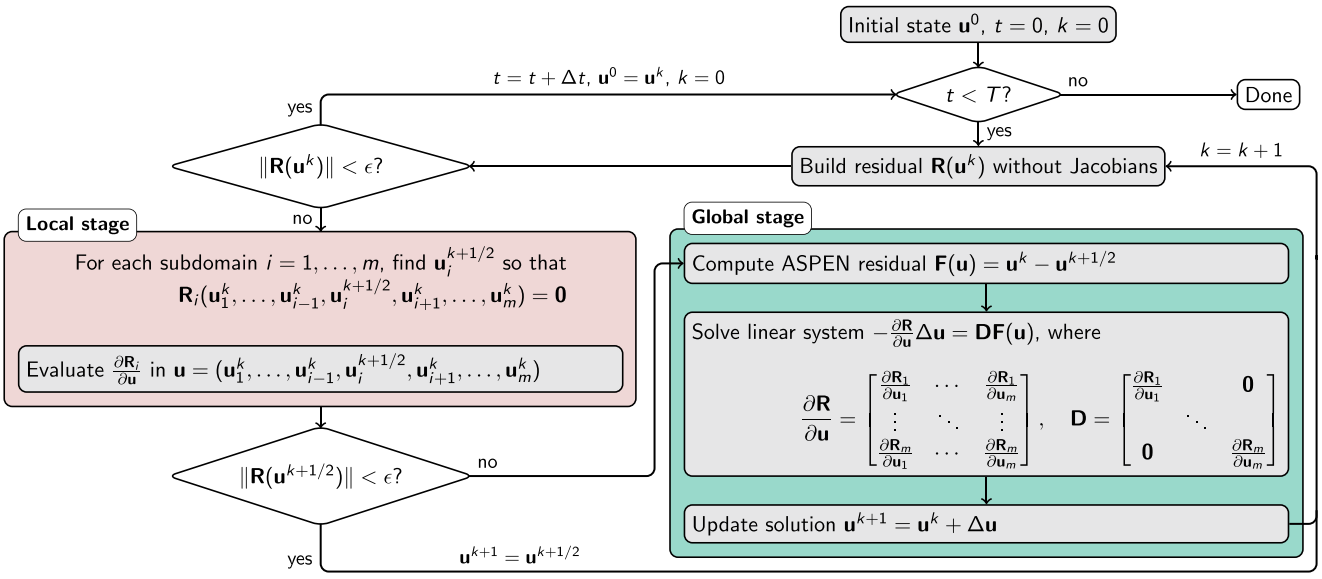

Fig. 1 Simulation flowchart for the ASPEN method applied to a fully implicit method. Each nonlinear iteration starts with a local stage, where we solve for the unknowns in each subdomain, keeping all other subdomain unknowns fixed. After the subdomain solve, we compute the corresponding Jacobian rows with the current solution. We then check if the global residual has converged, and proceed to the next timestep if this is the case. If not, we proceed to the global stage. Here, we compute the ASPEN residual as the difference between the current iterate and the combined subdomain solution, and solve for the Newton update using the ASPEN residual as initial guess. We then update the solution. This continues until the residual has converged

because the iterative solvers always use strict convergence tolerances.

3.3 Solution procedure and concurrency potential

Figure 1 illustrates the entire simulation workflow, from an initial state until the final time horizon is reached. The figure highlights important aspects of an ASPEN implementation and therefore merits some discussion. Because all subdomains are solved with fixed quantities from the previous nonlinear iteration, they can be solved concurrently. In this sense, the local stage is perfectly parallel. Note that we also check for convergence after the local stage. This can constitute significant computational savings, because it opens up for circumventing the global stage altogether. If the global residual is not converged after the local stage, we proceed to the global stage. This stage obviously requires some communication but can nonetheless be implemented very efficiently by utilizing the fact that all Jacobian blocks $\partial \mathbf{R}_i / \partial \mathbf{u}_j$ for subdomain i will be contained on the same worker (or processor). During the global stage, each worker will in effect have access to an entire row of Jacobian blocks. To compute the right-hand side, we therefore only need to communicate $\mathbf{F}(\mathbf{u})$ to all the workers. Moreover, if we use a Krylov-type iterative solver like GMRES, we only need to compute matrix-vector products on the form $-(\partial \mathbf{R}/\partial \mathbf{u})\mathbf{v}$. This can be efficiently implemented by communicating the linear iterate to all workers, multiplying it by its local row of Jacobian blocks, and gathering the results.

3.4 Complexity analysis

Assume N cells and an average of $n = N/m$ cells in each subdomain. Assume that one row in the Jacobian has d nonzero entries. The dominant part of a nonlinear iteration is the linear solve, which has complexity $(Nd)^\gamma$, where γ may vary from 1.2 for modern iterative linear solvers to 3 for direct Gaussian elimination. For spatially contiguous subdomains, including the well subdomains considered herein, the only off-diagonal elements neglected in the subdomain Jacobians correspond to cells outside the subdomain. To be on the conservative side, we therefore say that also the subdomain Jacobians have d entries per row.

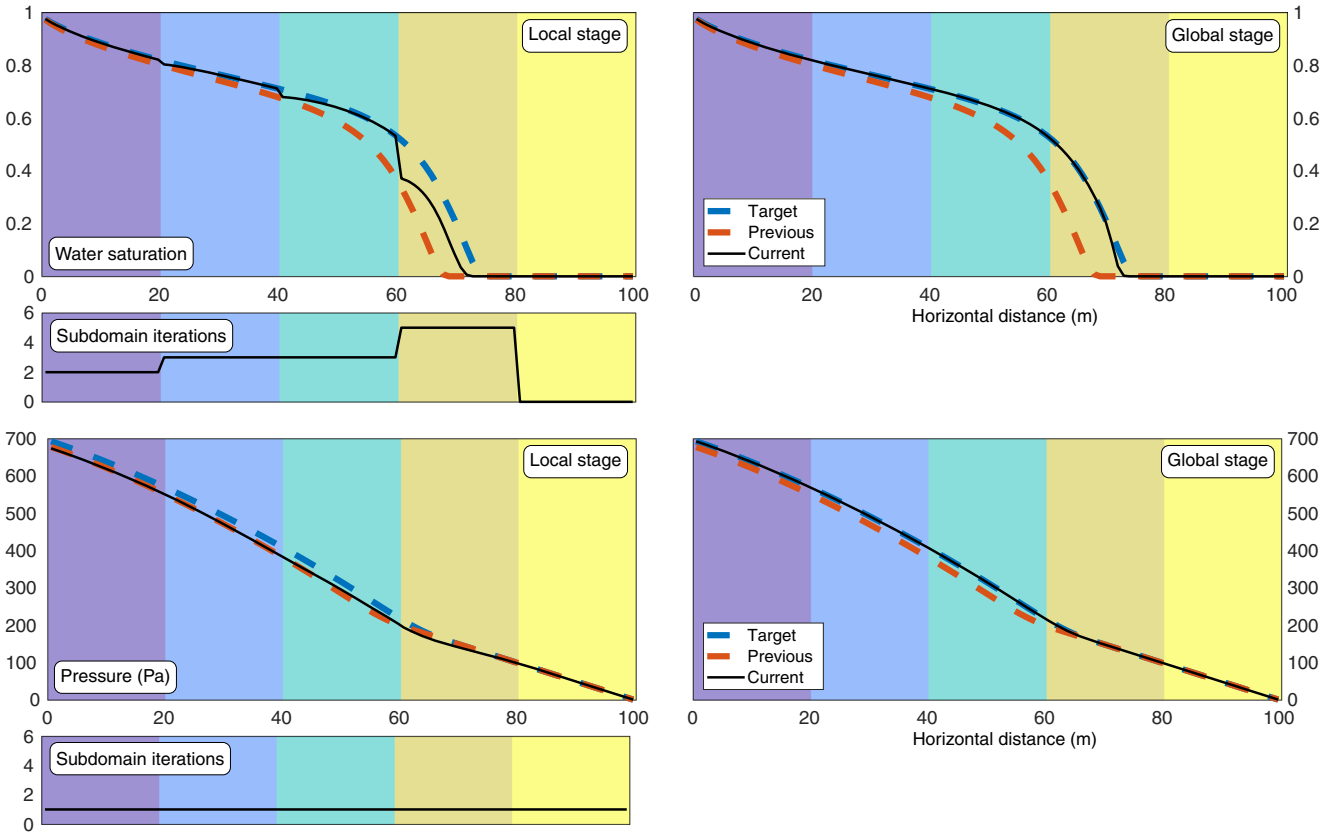

Fig. 2 Nonlinear iterates for saturation (top) and pressure (bottom) after the local stage (left) and the global stage (right) of an ASPEN iteration for a sequential solver. Colors distinguish subdomains, dashed lines show the solution at the previous timestep (red) and the target solution for the timestep (blue), whereas the solid, black lines show the current iterates. The bottom plots in the first column report the number of iterations used to solve each subdomain

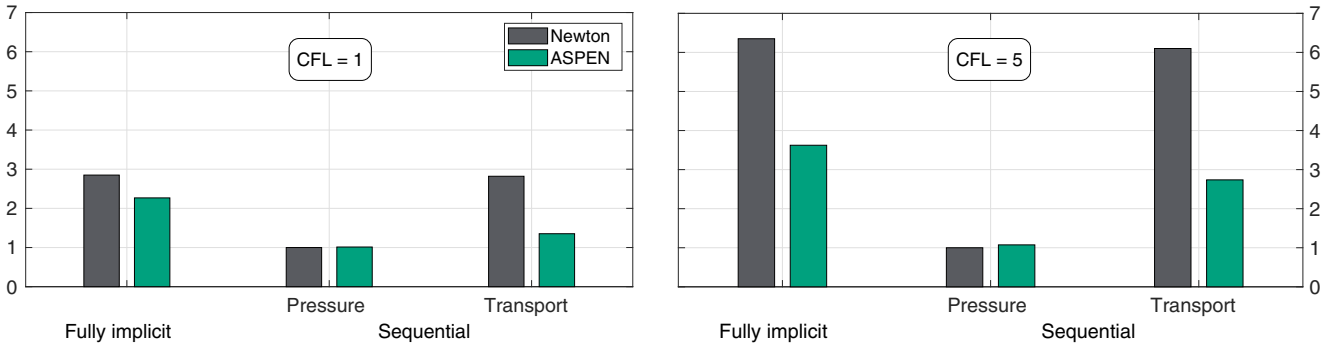

Fig. 3 Average number of nonlinear iterations per timestep for the Buckley–Leverett displacement. For Newton, the average is computed over timesteps, whereas for ASPEN we count each outer iteration as C_c "iterations" (C_c is defined in Eq. 22) and average the result over the timesteps

Hence, a linear solve will have complexity $(nd)^\gamma$. The outer iteration of ASPEN will be of the same complexity as a standard Newton iteration, e.g., be dominated by the linear solver with complexity $(Nd)^\gamma$. Assuming k is an upper bound for the number of nonlinear iterations used to solve one subdomain, the cost of one ASPEN iteration relative to the cost of one regular Newton iteration can be written as

$$C_c = \frac{k(nd)^\gamma + (Nd)^\gamma}{(Nd)^\gamma} = 1 + k\left(\frac{n}{N}\right)^\gamma = 1 + \frac{k}{m^\gamma} \quad (22)$$

when all the local subdomains are solved concurrently, and

$$C_s = \frac{mk(nd)^\gamma + (Nd)^\gamma}{(Nd)^\gamma} = 1 + k\left(\frac{n}{N}\right)^{\gamma-1} = 1 + \frac{k}{m^{\gamma-1}}, \quad (23)$$

when the local subdomains are solved serially. If the global residual has converged after the local stage (c.f. Fig. 1), we do not solve the global system and thus only count the added cost of the local solves (i.e., the last term). This analysis does not account for the cost of assembly, which scales linearly with the number of cells and primary variables, and that we for sufficiently small subdomains use a direct linear solver, which is generally more robust than iterative solvers.

4 Numerical examples

The ASPEN method is implemented using the automatic differentiation (AD-OO) simulator framework of MRST [26]. In the following, we report the performance of ASPEN on a number of examples using both fully implicit and sequential solution strategies and compare its efficiency with results from a nonlinear Newton–Raphson solver with damping strategies used in commercial simulators, but without preconditioning. We refer to the former method as ASPEN and the latter as Newton. The examples include both conceptual setups constructed to challenge the nonlinear solver and realistic setups with industry-grade geology and fluid properties. Applying ASPEN in combination with a sequential solver means herein that we solve both the pressure and transport subproblems with ASPEN, using the same spatial domain decomposition. For simplicity, we do not include outer iterations, and the sequential results are thus not guaranteed to converge to the fully implicit solution.

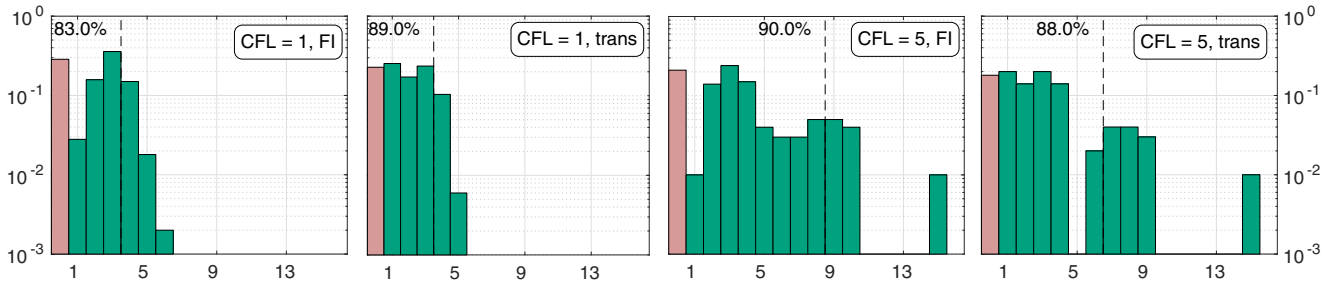

Fig. 4 Normalized histogram of the nonlinear subdomain iterations for the Buckley–Leverett example. Iterations are reported for the fully implicit (FI) simulations, and transport part (trans) of the sequential implicit simulations. The x-axis represents number of nonlinear subdomain iterations and the y-axis fraction of subdomain solves that required this number of iterations over the entire simulation. The leftmost bar in each histogram represents zero iterations. In other words, 20–30% of all subdomain solves required zero iterations, because they were already converged with the previous solution as the initial guess. Approximate 90 percentiles are represented as dashed vertical lines

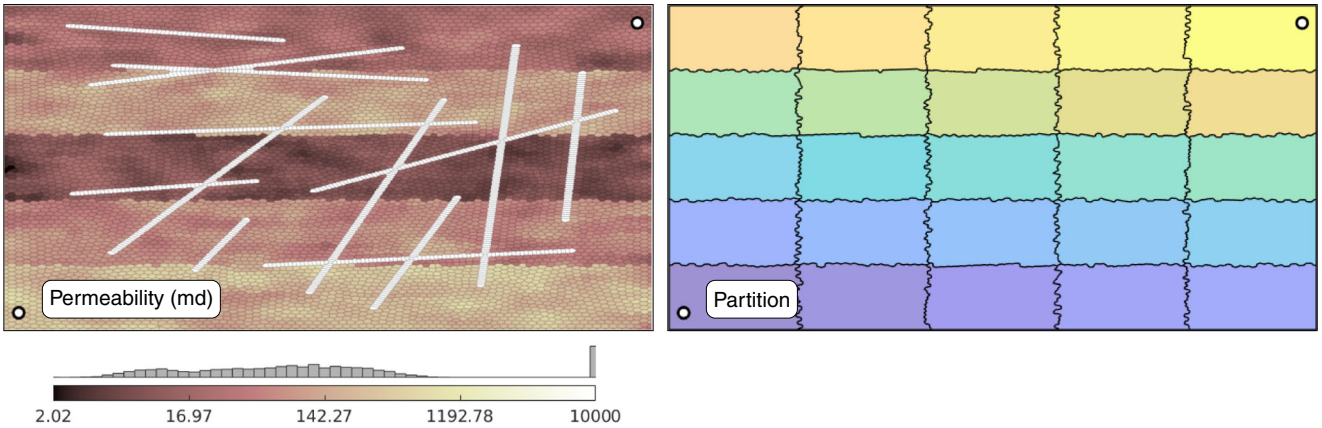

Fig. 5 Fractured reservoir permeability and partition. The permeability has a banded structure with lognormal permeabilities with a distinct mean in each band. The fracture corridors have a permeability approximately 100 times higher than the matrix

4.1 Example 1: Buckley–Leverett displacement

We consider water injected at a constant rate into a horizontal, one-dimensional, oil-filled reservoir with fluids produced at a constant pressure at the other end. Relative permeabilities are quadratic for both phases, whereas all other rock and fluid properties are set to unity. We subdivide the 100-cell grid into five subdomains of 20 cells each and simulate the problem with fully implicit and sequentially implicit solution strategies.

The top part of Fig. 2 reports the saturation after the local and global stages of an ASPEN iteration for a sequentiall simulation with CFL number equal 5. After the first local solve, the intermediate solution clearly exhibits kinks across the subdomain boundaries. The global iteration effectively levels out these kinks, so that after a full ASPEN iteration, the transport problem has almost converged. We also see that the subdomain iterations are localized to the propagating wave (shock followed by rarefaction wave). For the pressure update, the first set of subdomain solves has limited effect due to the elliptic nature of the pressure equation, but the global step effectively converges the pressure.

Figure 3 compares the average number of nonlinear iterations consumed per timestep for ASPEN and standard Newton for fully implicit and sequential solution strategies. Because ASPEN uses both subdomain and global iterations, we use C_c from Eq. 22, averaged over all timesteps, to define a number that represents a comparable computational cost to the Newton iterations. This is done for all examples in the following. We see that the ASPEN solver successfully reduces the effective number of nonlinear iterations for the fully implicit and transport problems compared with Newton. The pressure subproblem is linear for this example

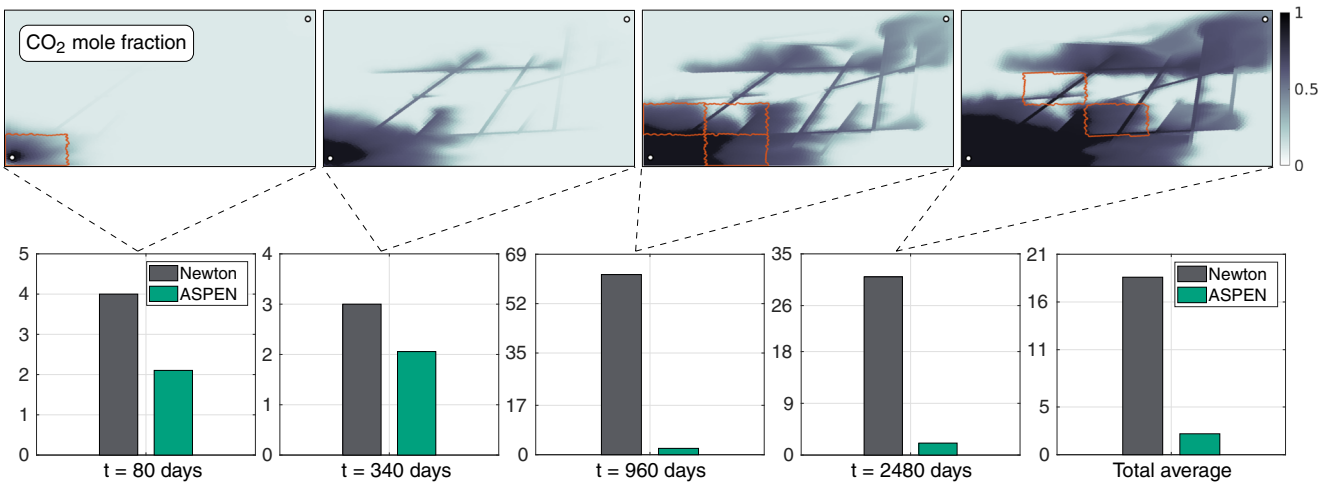

Fig. 6 Mole fraction of carbon dioxide after 80, 340, 960, and 2 480 days of injection. The bar charts report the corresponding nonlinear iterations required to solve the timestep by Newton and ASPEN. The rightmost bar chart reports the average number of nonlinear iterations per timestep. At each timestep, subdomains that required more than three iterations to converge the full problem are outlined in red

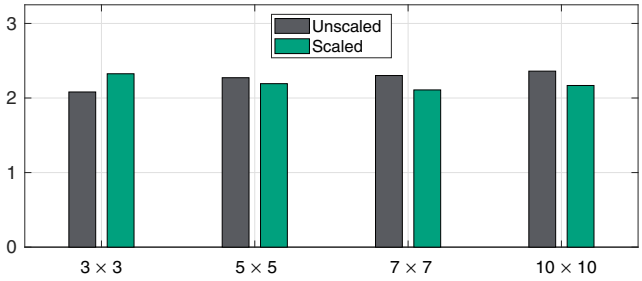

Fig. 7 Average number of nonlinear iterations per timestep for ASPEN with four different partitions in the fractured reservoir example. Scaled iterations are computed as C_c in Eq. 22, whereas unscaled iterations refer to the actual number of global outer iterations

and both Newton and ASPEN use a single iteration to converge.

It is also of interest to see the spread in the number of nonlinear subdomain iterations for ASPEN. Figure 4 reports a normalized histogram of the number of local iterations used per subdomain throughout the whole simulation. Because each ASPEN iteration involves converging every subdomain over an entire timestep, the maximum number of subdomain iterations per timestep (i.e., k in Eq. 22) will almost always be larger than the number of outer ASPEN iterations. However, keep in mind that a subdomain iteration is a lot less expensive compared to a global iteration due to the reduced size of the subdomains and the corresponding linearized systems.

4.2 Example 2: Fractured reservoir

Our next test is a slightly modified variant of an example from [38]. We consider a 1000 × 500 m^2 reservoir cross-section containing thirteen high-permeability fracture channels. The background sand consists of five bands in the east–west direction. Within each band, the permeability is drawn from a lognormal distribution with a distinct mean. We discretize the domain using conforming Voronoi cells generated by the `upr` module in MRST [1], and use a volumetric representation for the fracture channels (see Fig. 5).

We assume a two-phase, liquid–gas model with n-decane, carbon dioxide, and methane, with phase behavior defined by the cubic Peng–Robinson equation-of-state [41]. The reservoir is initially filled with a mixture of the three components. A well in the southwest corner injects a mixture of n-decane and carbon dioxide, whereas the producer in the northeast corner operates at a bottom-hole pressure of 50 bar. We simulate 2 555 days of injection with timesteps that gradually increase to 20 days, as a reasonable compromise between too high CFL numbers in the fractures and too low CFL numbers in the matrix. Rapid fingering of injected fluids through the fracture network, combined with initial reservoir pressure just below bubble-point nevertheless makes this a very challenging test case for the nonlinear solver.

Figure 6 reports the mole fraction of carbon dioxide at four selected timesteps along with the number of nonlinear iterations required to solve these timesteps with Newton and ASPEN. Before the injected carbon dioxide reaches the producer after approximately 900 days, both Newton and ASPEN converge steadily, using on average 3.7 and 2.0 nonlinear iterations per timestep, respectively. From 900 days and until the end of the simulation, Newton struggles significantly, and as a result, cuts the timestep in half multiple times, giving a large number of wasted iterations. ASPEN, on the other hand, continues to converge steadily. This is evident from the two last reported timesteps, for which Newton uses 62 and 31 nonlinear iterations to converge, whereas ASPEN converges in 2.2 and 2.1 scaled iterations, respectively. Averaged over all timesteps, Newton consequently uses 18.6 nonlinear iterations, whereas ASPEN requires only 2.2.

To investigate how the number of subdomains affects the nonlinear iteration count, we also run the same setup with 3 × 3, 7 × 7, and 10 × 10 partitions. Figure 7 reports the average number of nonlinear iterations per timestep. To give an idea of how computational effort shifts from the local to the global stage, the bar plot shows both the observed number of global ASPEN iterations and iterations scaled according to Eq. 22. As expected,

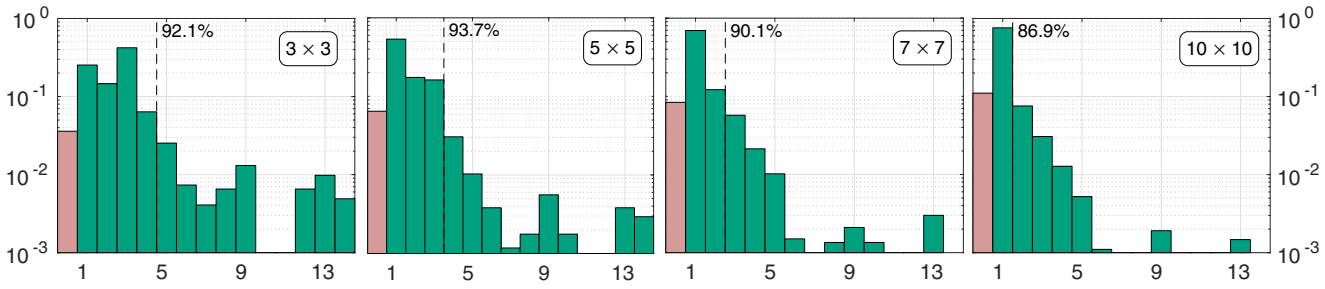

Fig. 8 Normalized histogram of nonlinear subdomain iterations for each partition of the fractured reservoir example. The leftmost bar in each chart indicates subdomains with zero iterations, whereas approximate 90 percentiles are shown as dashed vertical lines

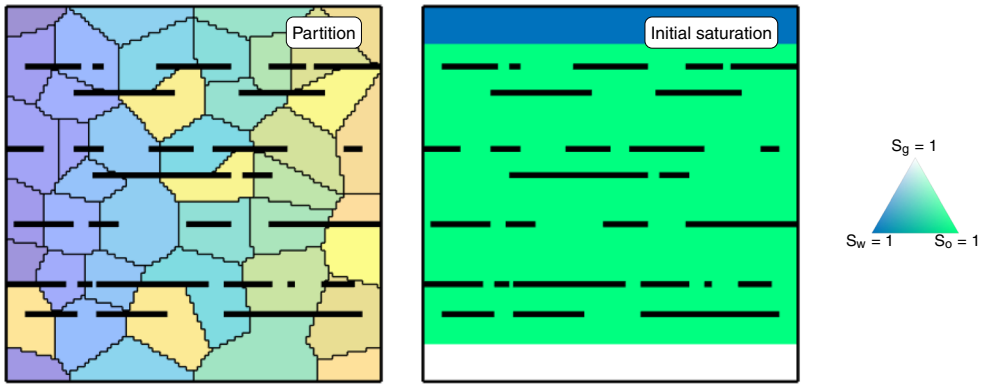

Fig. 9 Subdomain partition and initial saturation for the gravity segregation example. The black horizontal lines indicate sealing barriers

the two seem to converge to a constant difference, but from opposite sides: Unscaled iterations increase slightly with the number of subdomains because fewer subdomains means fewer kinks in the solution after the local stage (c.f. Fig. 2), and consequently less global iterations to resolve long-range interactions (think of the trivial case with a single subdomain, for which the solution converges in one global iteration). Scaled iterations decrease slightly with the number of subdomains because the theoretical cost of solving each subdomain tends to zero with the relative subdomain size. (In practice, each solve also involves a startup cost, which is not accounted for in our simplified analysis). The scaled iterations converge to a number slightly smaller than the unscaled ones because some of the ASPEN iterations converged after the local stage, so that we have only counted the cost of the local stage for these.

The normalized histograms of subdomain iterations reported in Fig. 8 show that approximately 90% of all subdomain solves required five, four, three, and one iteration or less to converge for the 3×3, 5×5, 7×7, and 10×10 partitions, respectively. A few subdomains require up to 15 iterations, but because these solves are localized, the cost is significantly less than for Newton, for which lack of convergence in a single cell will cause the solver to continue iterating over the whole domain. Note also that more than 10% of the subdomain solves do not require any iterations to converge for the 10×10 partition.

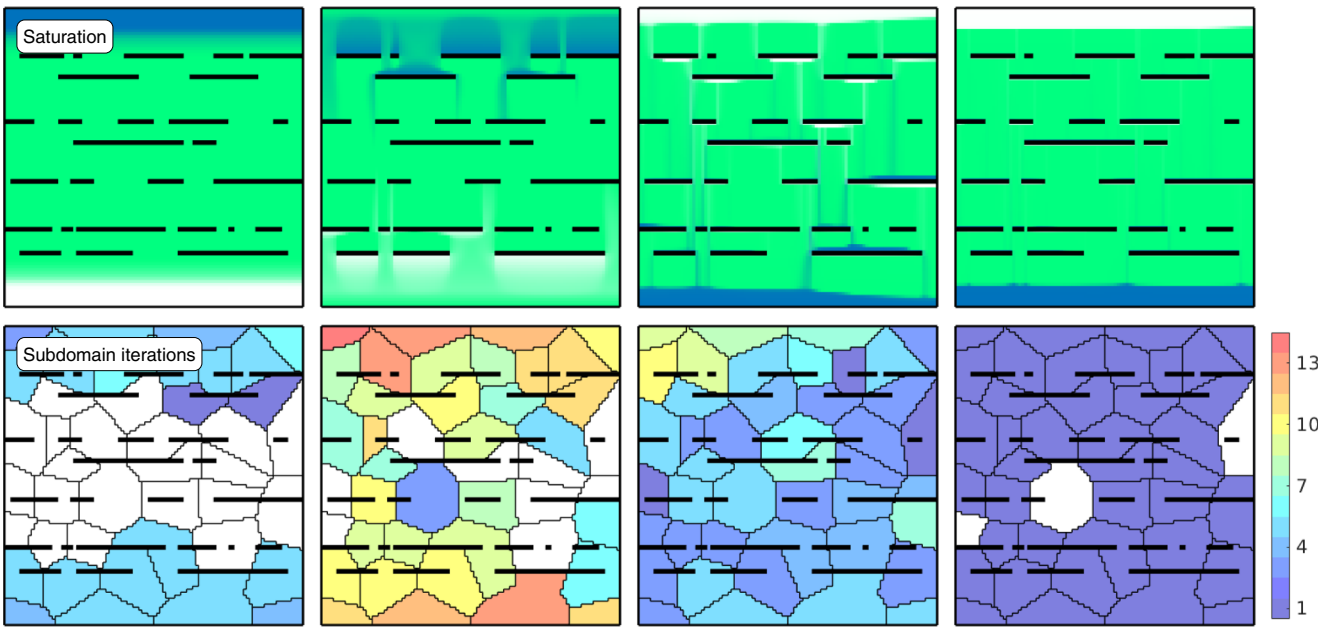

Fig. 10 Saturation and total subdomain iterations at four selected timesteps for the gravity segregation example. Subdomains with zero iterations are colored white

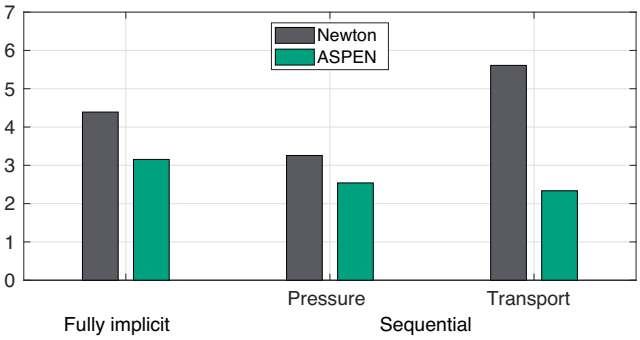

Fig. 11 Average number of nonlinear iterations per timestep for the gravity segregation example

4.3 Example 3: Gravity segregation with horizontal barriers

The example is inspired by a test case from [11]. A vertical 100×100 m^2 reservoir cross-section with homogeneous permeability of 1 mD and porosity of 0.3 is initially filled with an immiscible three-phase fluid. A heavy aqueous phase (density 1 500 kg/m^3, viscosity 1 cP) occupies the top 10% whereas the bottom 10% is occupied by a light gaseous phase (density 500 kg/m^3, viscosity 1 cP). An oleic phase with density 1000 kg/m^3 and viscosity 2 cP fills the remaining 80% of the domain. We use quadratic Brooks–Corey relative permeabilities for all phases [2]. Density differences will cause the aqueous phase to gradually exchange places with the light gaseous phase. The reservoir has several horizontal, partially sealing barriers that will deviate flow and give rise to a complex, gravity-driven flow pattern with fast-flow regions around barrier openings and stagnant regions close to barrier centers. Figure 9 shows the setup, with barriers and initial saturation.

To minimize the coupling among subdomains, we partition the domain so that regions with fast vertical flow lie close to the center of each subdomain. To this end, we first solve an incompressible pressure-drop problem from bottom to top and use the resulting interface fluxes to identify fast-flow regions. We use the center of each fast-flow region, together with points along the boundary, to construct a Delaunay triangulation and its dual Voronoi diagram. The partition is then defined by matching cell centroids in the grid to encompassing Voronoi cells. Finally, we split subdomains that are divided by a sealing barrier and merge the smaller part into a neighboring subdomain. The left part of Fig. 9 shows the subdomain partition.

Figure 10 reports the saturation at four selected timesteps along with the corresponding total number of subdomain iterations for the fully implicit ASPEN solver. As expected, more subdomain iterations are required in regions with large changes in the flow, but localizing iterations also means that some subdomains are already converged, so that no iterations are needed.

Figure 11 compares the average number of nonlinear iterations for the fully implicit and sequential implicit strategies with Newton and ASPEN. The transport subproblem in this example is particularly challenging for the nonlinear solver and is also where ASPEN gives the largest iteration reduction. Because the nonlinearities are resolved locally, the global ASPEN update will be very accurate.

The pressure subproblem only benefits slightly from the domain decomposition. We believe the main reason is that the pressure remains transient throughout large parts of the simulation as a result of changes induced by the upward and downward movement of phases with different density and subsequent change in gravity head. The main argument for using ASPEN for the pressure problem would thus be better parallel scalability.

Compared with the previous example, we see that a significantly larger fraction of the subdomain solves require more than seven iterations both in the fully implicit and the sequential algorithm (see Fig. 12). The distribution is nonetheless skewed toward the lower end of the scale: 6–7% of the subdomain solves do not require any iterations at all and the approximate 90 percentile is found at eight iterations for both solvers.

4.4 Example 4: Field-scale model (SAIGUP)

Moving on from conceptual examples, we next consider a field-scale model with realistic geology (Fig. 13). The

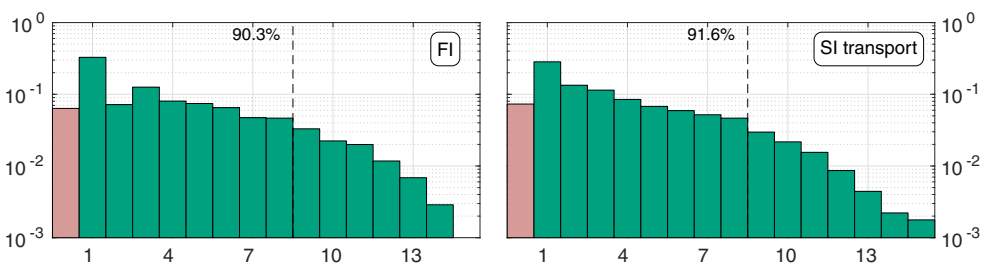

Fig. 12 Normalized histogram of nonlinear subdomain iterations for the gravity segregation example. The leftmost bar in each chart indicates subdomains with zero iterations, whereas the approximate 90 percentiles are represented as dashed vertical lines

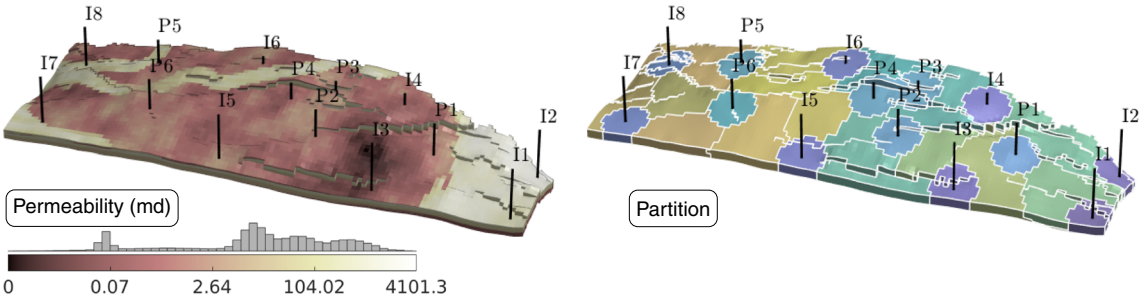

Fig. 13 Permeability and subdomain partition for the SAIGUP field model

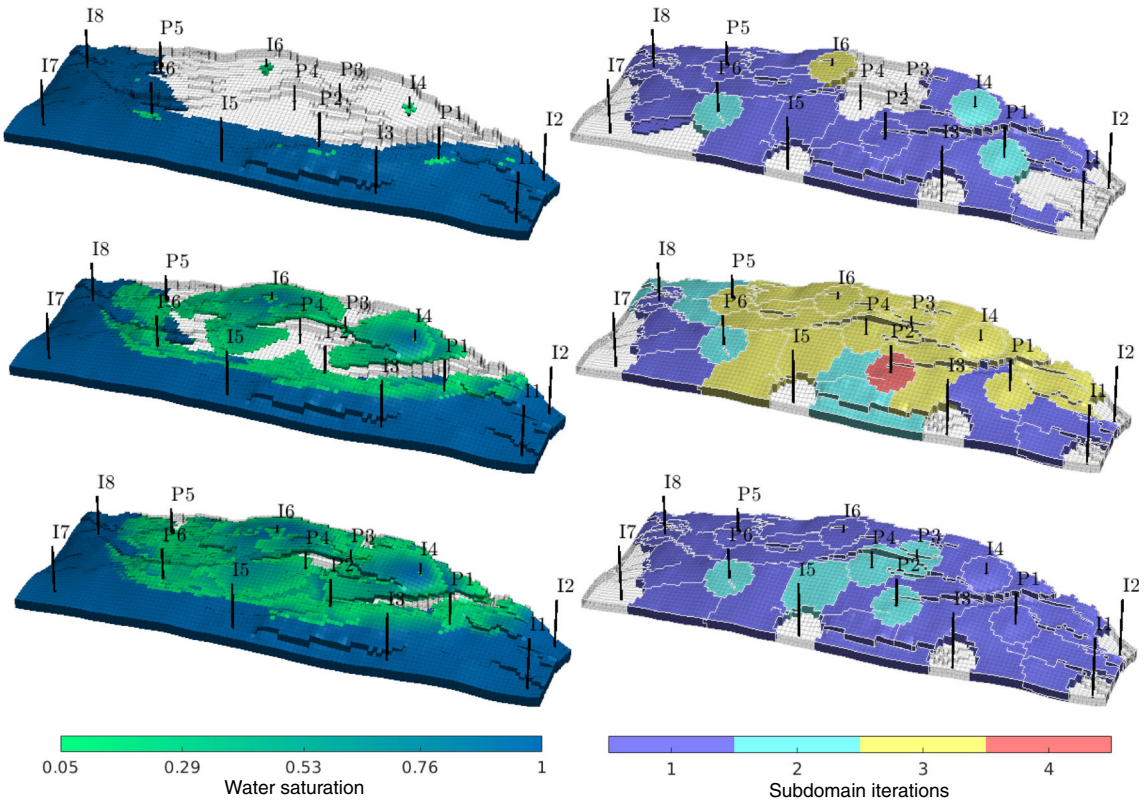

Fig. 14 Water saturation after 0.02, 4.7, and 18.4 years of injection, along with the total number of subdomain iterations used at each timestep. Empty cells correspond to subdomains with zero iterations

geomodel is a realization of a 9 × 3 km^2 shallow-marine oil field from the SAIGUP study [32], modelled as a 40 × 120 × 20 corner-point grid with 78,720 active cells. Permeabilities and porosities are drawn from multimodal distributions to model mud drapes and fast-flow regions. The reservoir also has multiple faults delineating different flow compartments.

We consider a simple two-phase oil/water model, with viscosity of 1 and 5 cP and density of 1014 and 859 kg/m^3 for the aqueous and oleic phase, respectively. Both phases are slightly compressible with quadratic Brooks–Corey relative permeabilities [2]. Initially at hydrostatic rest, the shallow regions of the reservoir are filled with

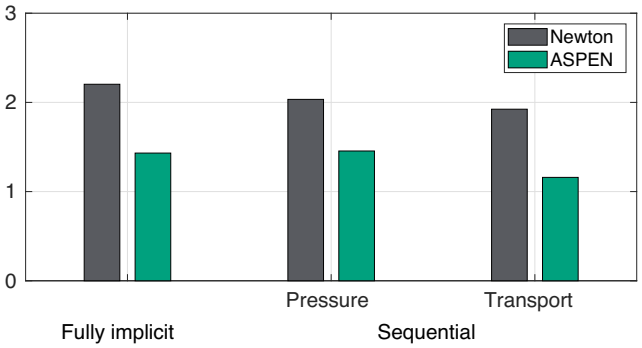

Fig. 15 Average number of nonlinear iterations per timestep for the SAIGUP example

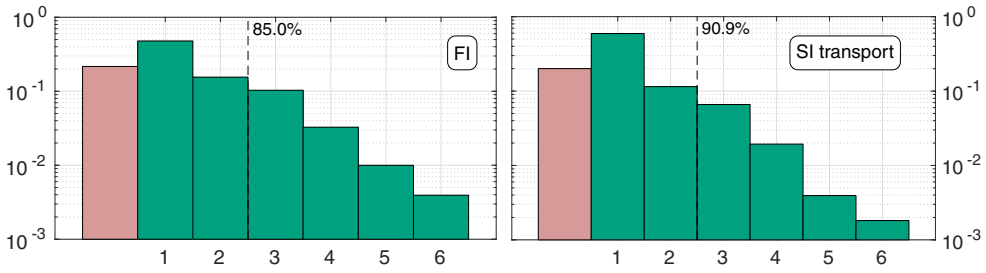

Fig. 16 Normalized histogram of subdomain iterations for the SAIGUP example. The leftmost bar in each chart indicates zero iterations, whereas the approximate 90 percentiles are represented as dashed vertical lines

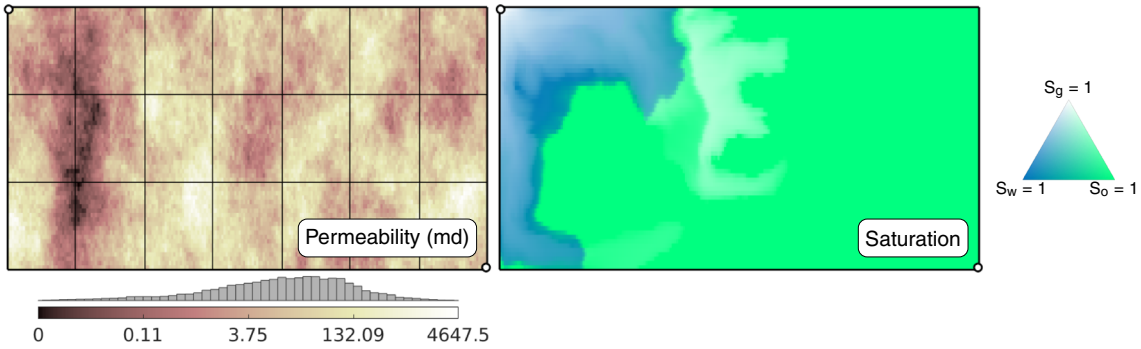

Fig. 17 Permeability and final saturation for the WAG example. The reservoir is rotated 90° clockwise

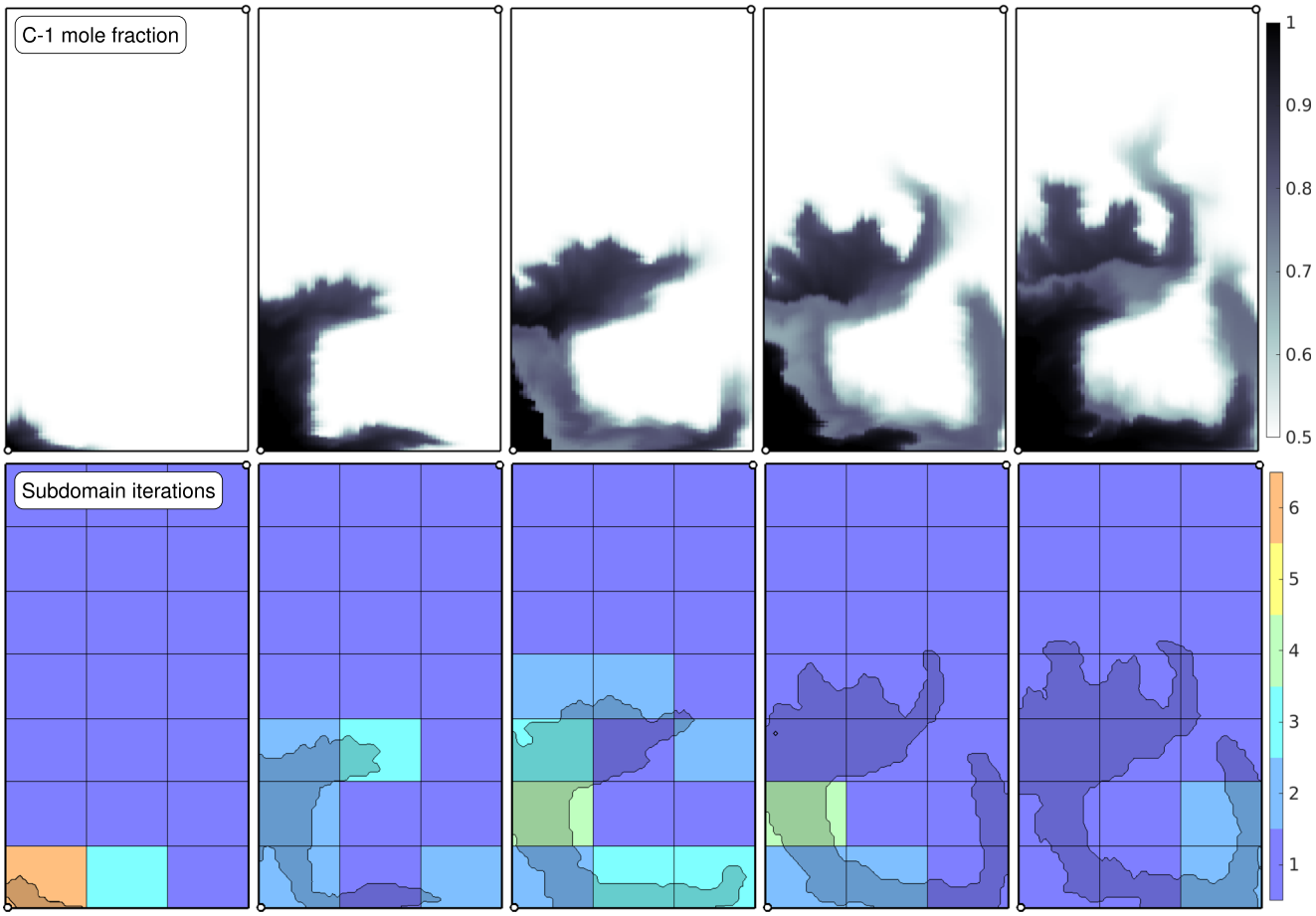

Fig. 18 C-1 mole fraction and subdomain iterations at five selected timesteps for the WAG example. The shaded areas overlaying the subdomain iterations indicate regions where all three phases are present

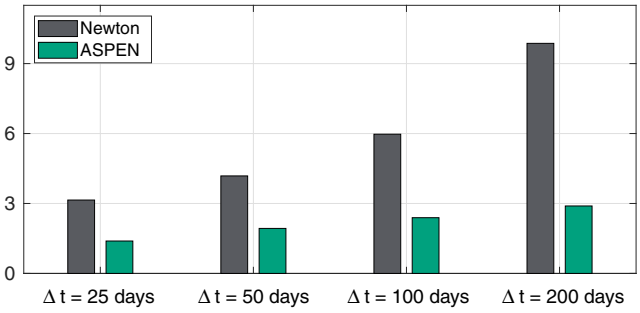

Fig. 19 Average number of nonlinear iterations per timestep for the WAG example

oil, whereas the remaining deeper regions are saturated with formation water. Seven injectors are placed along the reservoir perimeter, each injecting water at a constant rate in the range of 500–2 000 m^3 per day. The injectors encompass six producers operating at a fixed bottom-hole pressure of 200 bar. We simulate 30 years of production using timesteps that gradually increase to 100 days. To subdivide the domain, we first define a tube of consisting of all cells within a radius of 400 meters of each wellbore and merge any intersecting tubes into one subdomain. The remaining cells then are partitioned using the METIS graph partitioning algorithm [17] with face transmissibilities as edge weights (see Fig. 13).

Figure 14 reports water saturation after 0.02, 4.7, and 18.4 years of injection, along with the corresponding total subdomain iterations used to solve each timestep. Notice that several subdomains require zero iterations for all the reported timesteps.

Figure 15 reports the average number of nonlinear iterations per timestep with Newton and ASPEN for the fully implicit and sequential implicit solution strategies. The setup in this example results in only moderate CFL numbers, and Newton therefore performs near optimal, using approximately two iterations per timestep on average. We nonetheless see that nonlinear preconditioning with ASPEN is able to reduce the iteration count further for all solution strategies. Moreover, the histogram of subdomain iterations in Fig. 16 reveals that 85.0% and 90.9% of the subdomain solves required two iterations or less for the fully-implict and transport solvers, respectively. Altogether, this indicates that ASPEN is also well-suited for more typical simulation setups commonly seen in reservoir engineering applications, which is arguably more important than excellent performance on challenging corner cases.

4.5 Example 5: Water-alternating gas injection

After an initial water flood, a popular tertiary recovery technique is to inject solvent gas into the reservoir to dissolve and mobilize trapped residual oil. The gas is usually injected in smaller volumes, with water injection in between the gas volumes to uphold a favorable mobility ratio. We revisit an example from [33] posed on the first layer of Model 2 from the 10th SPE Comparative Solution Project [7], which is initially filled with a mixture of carbon species (C1, C3, C6, C10, C15, and C20), all in the liquid phase. We use a three-phase model with six components plus water and assume that water is immiscible. A well in the lower-left corner injects C1 gas for 5000 days, followed by water for 5000 days, followed by C1 gas for 5000 days. Fluids are produced from a well in the upper-right corner operating at a fixed bottom-hole pressure of 275 bar. Figure 17 shows the setup and final saturation, along with a 3×7 rectilinear subdomain partition used to decompose the domain.

A recurring issue in reservoir simulation is how to select appropriate timesteps. In practice, one compromises between timesteps long enough to get the simulation done in reasonable time and short enough to not impede nonlinear convergence. Emphasis tends to be on the latter, because a timestep that does not converge within the prescribed

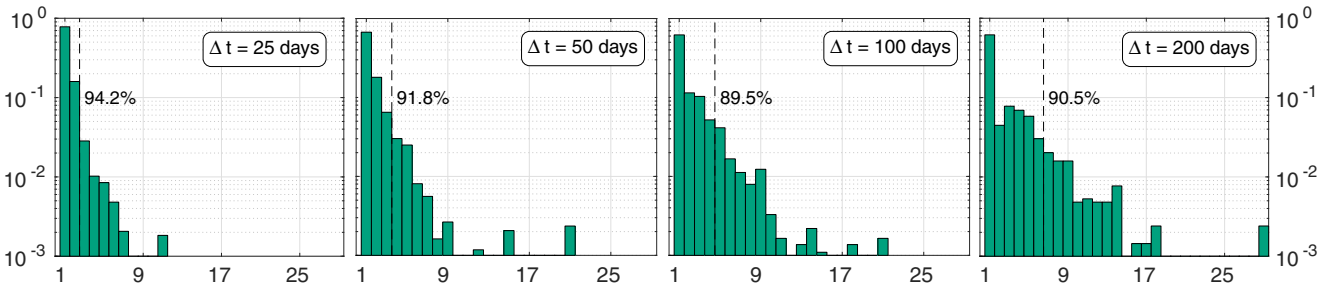

Fig. 20 Histogram of subdomain iterations with different timesteps for the WAG example. Approximate 90 percentiles are represented as dashed vertical lines

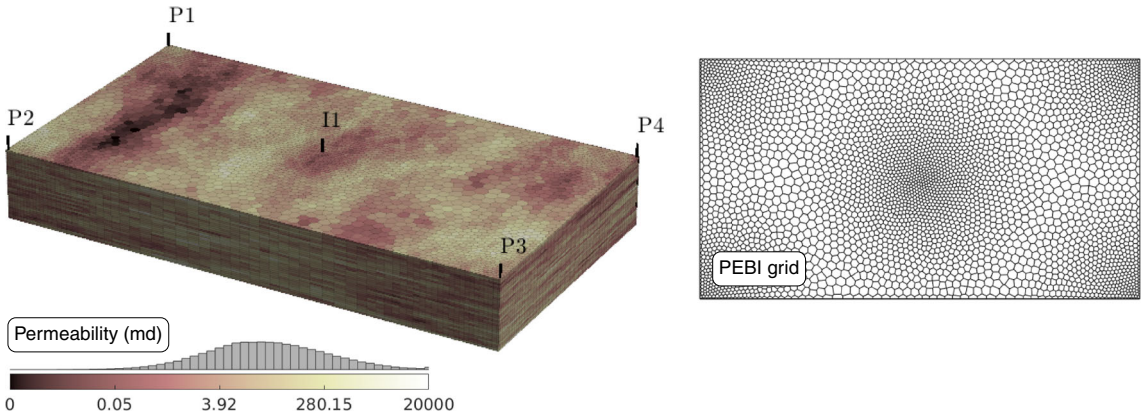

Fig. 21 Permeability and 2.5 D PEBI grid with near-well refinement for the Tarbert example

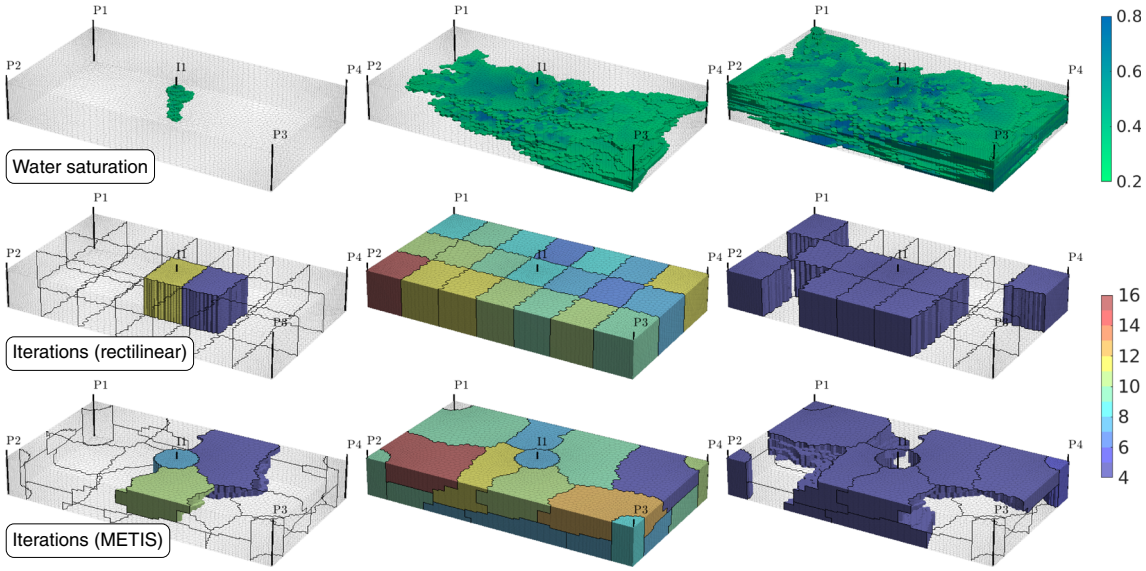

Fig. 22 Water saturation and corresponding subdomain iterations with rectilinear and METIS partitions for the Tarbert example. Water saturations are only plotted in cells with $S_a > 0.3$, whereas iterations are only reported in subdomains that used four iterations or more to solve the timestep

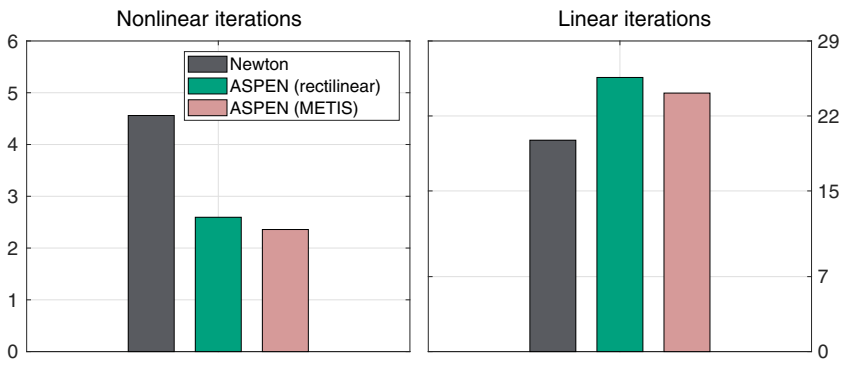

Fig. 23 Number of nonlinear iterations (left) and average number of linear iterations per nonlinear iteration (right) for the Tarbert example

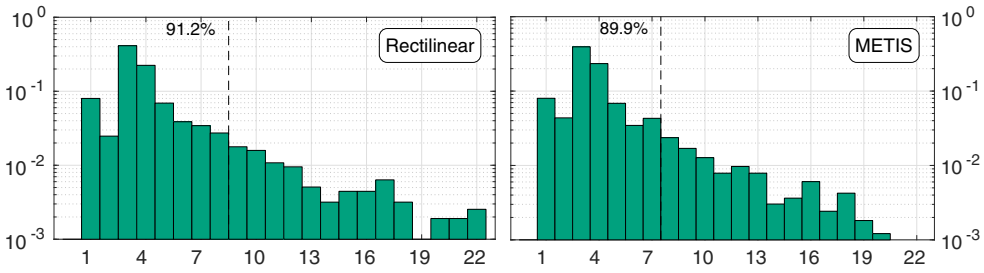

Fig. 24 Histogram of subdomain iterations for the Tarbert example. Approximate 90 percentiles are represented as dashed vertical lines

iteration limit is reduced and restarted, resulting in a potentially large amount of wasted computational effort. In this example, we compare Newton and ASPEN for timestep targets of 25, 50, 100, and 200 days. At the beginning, and when changing from/to water to/from gas injection, we use eight timesteps that gradually increase to the timestep target, resulting in a total of 624, 324, 174, and 99 timesteps for the different setups. The Newton solver is allowed to perform a maximum of 25 iterations without converging before the timestep is halved and restarted. Figure 18 shows the C1 mole fraction at five times for the setup with finest timesteps, together with corresponding subdomain iterations.

Figure 19 reports the average number of nonlinear iterations per timestep. As expected, the iteration count increases with the timestep length for both solvers: from 3.1 to 9.9 for Newton, and from 1.4 to 2.9 for ASPEN. As in the SAIGUP example, we see that ASPEN performs similar to Newton when this solver performs near optimal. Moreover, the increase in iteration count with timestep length is significantly smaller for ASPEN than for Newton.

From the subdomain iterations in Fig. 20 we observe that the distribution shifts towards the right (i.e., the subdomain solves consume more iterations) for larger timesteps: 94.2% of all subdomain solves consume two iterations or less for the setup with timesteps of 25 days, whereas the (approximate) 90% percentile is at three, four, and six iterations for the setups with timesteps of 50, 100, and 200 days, respectively. Together with the very modest increase in average nonlinear iterations reported in Fig. 19, this demonstrates how ASPEN efficiently resolves unbalanced nonlinearities locally, which makes the method very robust with respect to timestep length. This robustness is very valuable when simulating complex recovery scenarios.

4.6 Example 6: Unstructured grid

As an example of an unstructured grid, we consider an inverted five-spot pattern in a rectangular domain of 300 × 600 m represented on a 2.5D perpendicular bisector (PEBI) grid. Petrophysical properties representative of a shallow-marine reservoir are sampled from the first 35 layers of Model 2 from the 10th SPE Comparative Solution Project [7]; see Fig. 21. We use two different partitions: a simple 3 × 7 rectilinear partition and a partition strategy with cylindrical subdomains around each well and METIS for the rest of the reservoir (see Example 4). We simulate injection of water at a constant rate for 2000 days; the upper row of Fig. 22 shows three snapshots of the resulting displacement.

From the iteration counts reported in Figs. 22 to 24, we see that the nonlinear subdomain iterations follow the same trend for both partitions and that the performance of ASPEN appears to be almost invariant to partition type, albeit with a slight preference to the METIS partition, in which the subdomains are adapted to the heterogeneous rock.

As discussed in the complexity analysis, solving the linearized systems tends to be the dominant part of a nonlinear iteration. In this example, we therefore also look at linear solver performance. We use GMRES with CPR preconditioning [8] as the linear solver, both for ASPEN and Newton. Figure 23 reports average linear iterations per

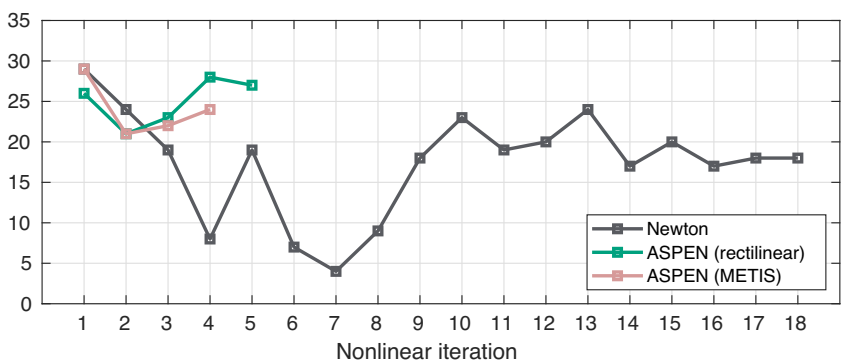

Fig. 25 Linear iterations per nonlinear iteration for a selected timestep for the Tarbert example in which Newton required 18 nonlinear iterations and the two ASPEN solvers required 5 and 4 nonlinear iterations

nonlinear iteration. On average, using ASPEN increases the number of linear iterations per nonlinear iteration by 29.7% and 22.3%, respectively. Such an increase should be expected, even if the linear system itself has the same structure as for regular Newton, because the ASPEN variable updates are necessarily larger in magnitude. Figure 25 illustrates this effect for a timestep for which Newton uses 18 nonlinear iterations, whereas ASPEN uses 5 (rectilinear) and 4 (METIS). The first two nonlinear updates of Newton consume as many linear iterations as the ASPEN solvers, while the last ones consume less. All in all, however, we can conclude that ASPEN does not seem to significantly impede linear convergence.

5 Concluding remarks

This work presents a highly flexible implementation of nonlinear spatial-domain decomposition preconditioning with ASPEN that can handle industry-grade compositional fluid models and general, polytopal grids. Through a series of test cases with different types of flow physics, heterogeneity, and grid types, we have demonstrated that ASPEN can be a competitive nonlinear solution strategy, especially for complex injection scenarios with unbalanced nonlinearities.

A key problem with standard Newton's method is that local convergence problems have to be resolved through expensive global iterations. (Such problems are typically encountered in the near-well region, near strong displacement fronts, in areas with strong gas expansion, when wells are opened or shut in, and in regions with high media contrasts combined with deviated cell geometries, etc.) With ASPEN, convergence problems are usually handled locally in the nonlinear preconditioning stage, which enables the method to focus the computational effort and avoid wasting iterations on converged subsets of variables. In many cases, a significant fraction of the cells remain invariant from one step to the next, typically in unswept areas for secondary and tertiary recoveries, and with ASPEN one avoids iterating these cells. As a result of the localization, ASPEN also appears to be more robust with respect to timestep lengths. Likewise, the method is almost as efficient, and in some cases more efficient than Newton's method at its best. This is important in practice, because it is generally difficult to select timesteps that are sufficiently long to ensure computational efficiency and at the same time not so long that they cause convergence problems in the nonlinear solver.

These conclusions are based on using nonlinear (and linear) iterations, accompanied by an estimate of computational costs based on a complexity analysis, as our measure of computational efficiency. This is obviously not a totally reliable metric, and a full assessment including CPU times is necessary to draw firm conclusions on performance over the standard Newton method. We are currently implementing a version of this algorithm in a C++ reservoir simulator to assess the runtime performance in practice. The observed reductions in nonlinear iterations for our prototype implementation are nonetheless significant, giving a clear indication of enhanced nonlinear solver efficiency. Moreover, our reformulation of the global linearized system enables us to leverage well-established, efficient linear solution strategies. This is particularly important for large simulation problems, for which lack of efficient linear preconditioners would render computation of the global update prohibitively expensive. In line with previous work, we also argue that ASPEN is very well suited for parallelization. In fact, with our reformulation of the global linearized system, ASPEN may be seen as a way to parallelize Newton's method itself.

Another advantage of the ASPEN framework is its treatment of wells, which are often a main cause for nonlinear convergence issues. The overall numerical strategy presented in this paper is sufficiently general to allow for various decomposition and solution options, including solving wells as a separate subdomain (akin to nonlinear facility solvers found in some simulation tools). For the test cases studied herein, we have used a slightly different strategy that assigns all perforated cells, padded by at least one layer of extra cells, to a single subdomain. (Note that there is no requirement that ASPEN subdomains are connected, so that a well subdomain can consist of multiple disconnected parts.) The rest of the domain can then be decomposed by means of a suitable graph partitioning algorithm. The drawback of this approach is that the resulting subdomain can represent a large fraction of the reservoir cells for reservoirs with high well density. Subdivision is straightforward if wells are localized and operated independently, but further research is necessary to devise good strategies for handling cases with long and complex (multibranch) trajectories and/or wells subject to group or field constraints/controls, e.g., to ensure that constraint violations are handled correctly across subdomains.

We have tested our ASPEN implementation both for fully implicit problems and for pressure and transport subproblems in sequential implicit strategies. As expected, we observe significantly higher efficiency gain for transport problems than for pressure problems. A natural extension is to use ASPEN as a nonlinear transport solver in combination with local timestepping techniques [28] and spatial adaptivity with dynamic coarsening [20, 23]. As argued in [19], nonlinear transport solvers localized by optimal cell ordering [21, 39] are multiplicative Schwarz methods. Our results encourage further development of such solvers as

multiplicative Schwarz preconditioning techniques (MSPEN). Another possible approach is to combine ASPEN with sequential fully implicit strategies [37], and in [22] we discuss how the choice of fully implicit versus sequential implicit methods can be localized to individual subdomains with ASPEN as an overall framework.

Acknowledgements The authors thank Halvor Møll Nilsen for fruitful discussions and TOTAL E&P for funding and allowing the publication of this work.

Funding Open access funding provided by SINTEF AS.

References

1. Berge, R.L., Klemetsdal, Ø.S., Lie, K.-A.: Unstructured Voronoi grids conforming to lower dimensional objects. Comput. Geosci. **23**(1), 169–188 (2019). https://doi.org/10.1007/s10596-018-9790-0
2. Brooks, R.H., Corey, A.T.: Properties of porous media affecting fluid flow. J Irrigation Drainage Div. **92**(2), 61–90 (1966). https://doi.org/10.1061/JRCEA4.0000425
3. Cai, X.-C., Keyes, D.E.: Nonlinearly preconditioned inexact Newton algorithms. SIAM J. Comput. Sci. **24**(1), 183–200 (2002). https://doi.org/10.1137/s106482750037620x
4. Cai, X.C., Keyes, D.E., Marcinkowski, L.: Non-linear additive schwarz preconditioners and application in computational fluid dynamics. Int. J. Numer. Meth. Fluids **40**(12), 1463–1470 (2002). https://doi.org/10.1002/fld.404
5. Cao, H., Tchelepi, H.A., Wallis, J., Yardumian, H.: Parallel scalable unstructured CPR-type linear solver for reservoir simulation. In: SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers (SPE) (2005). https://doi.org/10.2118/96809-ms
6. Chien, H., Mark, C., Tchelepi, H.A., Yardumian, H.E., Chen, W.H.: Scalable parallel multi-purpose reservoir simulator. In: SPE Symposium on Reservoir Simulation, pp 17–30 (1997). https://doi.org/10.2118/37976-MS
7. Christie, M.A., Blunt, M.J.: Tenth SPE comparative solution project: A comparison of upscaling techniques. SPE Reserv Eval Eng **4**(04), 308–317 (2001). https://doi.org/10.2118/72469-pa
8. Demidov, D.: AMGCL: An efficient, flexible, and extensible algebraic multigrid implementation. Lobachevskii J. Math. **40**(5), 535–546 (2019). https://doi.org/10.1134/S1995080219050056
9. Dolean, V., Gander, M.J., Kheriji, W., Kwok, F., Masson, R.: Nonlinear preconditioning: How to use a nonlinear Schwarz method to precondition Newton's method. SIAM J. Sci. Comput. **38**(6), A3357–A3380 (2016)
10. Gries, S., Stüben, K., Brown, G.L., Chen, D., Collins, D.A.: Preconditioning for efficiently applying algebraic multigrid in fully implicit reservoir simulations. SPE J. **19**(4), 726–736 (2014). https://doi.org/10.2118/163608-PA
11. Hamon, F.P., Mallison, B.T., Tchelepi, H.A.: Implicit hybrid upwind scheme for coupled multiphase flow and transport with buoyancy. Comput. Methods Appl. Mech. Eng. **311**, 599–624 (2016). https://doi.org/10.1016/j.cma.2016.08.009
12. Henson, V.E., Yang, U.M.: BoomerAMG: A parallel algebraic multigrid solver and preconditioner. Appl. Numer. Math. **41**(1), 155–177 (2002). https://doi.org/10.1016/S0168-9274(01)00115-5
13. Jenny, P., Lee, S.H., Tchelepi, H.A.: 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). https://doi.org/10.1016/j.jcp.2006.01.028
14. Jenny, P., Tchelepi, H.A., Lee, S.H.: Unconditionally convergent nonlinear solver for hyperbolic conservation laws with s-shaped flux functions. J Comput. Phys. **228**(20), 7497–7512 (2009). https://doi.org/10.1016/j.jcp.2009.06.032
15. Jiang, J., Tchelepi, H.A.: Nonlinear acceleration of sequential fully implicit (SFI) method for coupled flow and transport in porous media. Comput. Methods Appl. Mech. Eng. **352**, 246–275 (2019). https://doi.org/10.1016/j.cma.2019.04.030
16. Kaarstad, T., Froyen, J., Bjorstad, P., Espedal, M.: A massively parallel reservoir simulator. In: SPE Reservoir Simulation Conference. Society of Petroleum Engineers (SPE) (1995). https://doi.org/10.2118/29139-MS
17. Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comput. **20**(1), 359–392 (1998). https://doi.org/10.1137/S1064827595287997
18. Killough, J.E., Wheeler, M.F.: Parallel iterative linear equation solvers: An investigation of domain decomposition algorithms for reservoir simulation. In: SPE Symposium on Reservoir Simulation (1987). https://doi.org/10.2118/16021-MS
19. Klemetsdal, Ø.S.: Efficient Solvers for Field-Scale Simulation of Flow and Transport in Porous Media. Phd thesis, Norwegian University of Technology and Science (2019)
20. Klemetsdal, Ø.S., Lie, K.-A.: Dynamic coarsening and local reordered nonlinear solvers for simulating transport in porous media. SPE J., (January), 1–20. https://doi.org/10.2118/201089-PA (2020)
21. Klemetsdal, Ø.S., Møyner, O., Lie, K.-A.: Robust nonlinear Newton solver with adaptive interface-localized trust regions. SPE J. **24**(04), 1576–1594 (2019). https://doi.org/10.2118/195682-PA
22. Klemetsdal, Ø.S., Moncorgé, A., Nilsen, H.M., Møyner, O., Lie., K.-A.: An adaptive sequential fully implicitdomain-decomposition solver. SPE J. In press (2021a)
23. Klemetsdal, Ø.S., Møyner, O., Moncorgé, A., Nilsen, H.M., Lie, K.-A.: High-resolution compositional reservoir simulation with dynamic coarsening and local timestepping for unstructured grids. SPE J. In press (2021b)
24. Lacroix, S., Vassilevski, Y., Wheeler, J., Wheeler, M.F.: Iterative solution methods for modeling multiphase flow in porous media fully implicitly. SIAM J. Sci Comput. **25**(3), 905–926 (2003). https://doi.org/10.1137/S106482750240443X
25. Li, J., Wong, Z.Y., Tomin, P., Tchelepi, H.: Sequential implicit Newton method for coupled multi-segment wells. In: SPE Reservoir Simulation Conference (2019). https://doi.org/10.2118/193833-MS
26. Lie, K.-A.: An Introduction to Reservoir Simulation Using MATLAB/GNU Octave: User guide for the MATLAB Reservoir

Simulation Toolbox (MRST). Cambridge University Press, Cambridge (2019). https://doi.org/10.1017/9781108591416
27. Lie, K.-A., Møyner, O., Natvig, J.R., Kozlova, A., Bratvedt, K., Watanabe, S., Li, Z.: Successful application of multiscale methods in a real reservoir simulator environment. Comput. Geosci. **21**(5-6), 981–998 (2017). https://doi.org/10.1007/s10596-017-9627-2
28. Linga, G., Møyner, O., Møll, H., Moncorgé, A., Lie, K.-A.: An implicit local time-stepping method based on cell reordering for multiphase flow in porous media. J. Comput. Phys.: X **6**, 100051 (2020). https://doi.org/10.1016/j.jcpx.2020.100051
29. Liu, L., Keyes, D.E.: Field-split preconditioned inexact Newton algorithms. SIAM J. Comput. Sci **37**(3), A1388–A1409 (2015). https://doi.org/10.1137/140970379
30. Liu, L., Keyes, D.E., Sun, S.: Fully implicit two-phase reservoir simulation with the additive Schwarz preconditioned inexact Newton method. In: SPE Reservoir Characterization and Simulation Conference and Exhibition, 16-18 September, Abu Dhabi, UAE. Society of Petroleum Engineers (SPE) (2013)
31. Luo, L., Cai, X.-C., Keyes, D.E.: Nonlinear preconditioning strategies for two-phase flows in porous media discretized by a fully implicit discontinuous galerkin method. SIAM J. Sci. Comput. (0), S317–S344 (2021)
32. Manzocchi, T. 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). https://doi.org/10.1144/1354-079307-790
33. Moncorgé, A., Tchelepi, H.A., Jenny. P.: Sequential fully implicit formulation for compositional simulation using natural variables. J. Comput. Phys. **371**, 690–711 (2018). https://doi.org/10.1016/j.jcp.2018.05.048
34. Moncorgé, A., Møyner, O., Tchelepi, H.A., Jenny, P.: Consistent upwinding for sequential fully implicit multiscale compositional simulation. Comput. Geosci. **24**, 533–550 (2020). https://doi.org/10.1007/s10596-019-09835-6
35. Møyner, O.: Nonlinear solver for three-phase transport problems based on approximate trust regions. Comput. Geosci. **21**(5-6), 999–1021 (2017)
36. Møyner, O., Lie, K.-A.: A multiscale restriction-smoothed basis method for compressible black-oil models. SPE J. **21**(06), 2079–2096 (2016). https://doi.org/10.2118/173265-PA
37. Møyner, O., Moncorgé A.: Nonlinear domain decomposition scheme for sequential fully implicit formulation of compositional multiphase flow. Comput. Geosci., 789–806. https://doi.org/10.1007/s10596-019-09848-1 (2019)
38. Møyner, O., Tchelepi, H.A.: A mass-conservative sequential implicit multiscale method for isothermal equation of state compositional problems. SPE J. (23), 2376–2393. https://doi.org/10.2118/182679-PA (2018)
39. Natvig, J.R., Lie, K.-A.: Fast computation of multiphase flow in porous media by implicit discontinuous galerkin schemes with optimal ordering of elements. J Comput. Phys. **227**(24), 10108–10124 (2008). https://doi.org/10.1016/j.jcp.2008.08.024
40. Nordbotten, J., Bjørstad, P.: On the relationship between the multiscale finite-volume method and domain decomposition preconditioners. Comput. Geosci. **12**(3), 367–376 (2008)
41. Peng, D.-Y., Robinson, D.B.: A new two-constant equation of state. Ind. Eng. Chem. Res. **15**(1), 59–64 (1976). https://doi.org/10.1021/i160057a011
42. Redlich, O., Kwong, J.N.S.: On the thermodynamics of solutions. V. An equation of state. Fugacities of gaseous solutions. Chem Rev **44**(1), 233–244 (1949). https://doi.org/10.1021/cr60137a013
43. Skogestad, J.O., Keilegavlen, E., Nordbotten, J.M.: Domain decomposition strategies for nonlinear flow problems in porous media. J Comput. Phys. **234**, 439–451 (2013). https://doi.org/10.1016/j.jcp.2012.10.001
44. Soave, G.: Equilibrium constants from a modified Redlich–Kwong equation of state. Chem Eng Sci **27**(6), 1197–1203 (1972). https://doi.org/10.1016/0009-2509(72)80096-4
45. Stueben, K.: An introduction to algebraic multigrid. Appendix in book "Multigrid" by U. Trottenberg, C.W. Oosterlee and A. Schueller, pp. 413–532. Academic Press, Cambridge (2001)
46. Stueben, K., Clees, T., Klie, H., Lu, B., Wheeler, M.F.: Algebraic multigrid methods (AMG) for the efficient solution of fully implicit formulations in reservoir simulation. In: SPE Reservoir Simulation Symposium. Society of Petroleum Engineers (SPE) (2007). https://doi.org/10.2118/105832-ms
47. Trangenstein, J.A., Bell, J.B.: Mathematical structure of the black-oil model for petroleum reservoir simulation. SIAM J. Appl Math. **49**(3), 749–783 (1989). https://doi.org/10.1137/0149044
48. Wallis, J.R.: Incomplete Gaussian elimination as a preconditioning for generalized conjugate gradient acceleration. In: SPE Reservoir Simulation Symposium, 15-18 November, San Francisco, California. Society of Petroleum Engineers (1983). https://doi.org/10.2118/12265-MS
49. Wallis, J.R., Kendall, R.P., Little, T.E.: Constrained residual acceleration of conjugate residual methods. In: PE Reservoir Simulation Symposium (1985). https://doi.org/10.2118/13536-MS
50. Watts, J.W.: A compositional formulation of the pressure and saturation equations. SPE Reserv. Eng. **1**(03), 243–252 (1986). https://doi.org/10.2118/12244-pa
51. Wong, Z.Y.: Sequential-Implicit Newton's Method for Geothermal Reservoir Simulation. PhD thesis, Stanford University (2018)