Computational Geosciences (2024) 28:253–272
https://doi.org/10.1007/s10596-023-10238-x

ORIGINAL PAPER

Block constrained pressure residual preconditioning for two-phase flow in porous media by mixed hybrid finite elements

Stefano Nardean[1,2] · Massimiliano Ferronato[2] · Ahmad Abushaikha[1]

Received: 16 November 2022 / Accepted: 10 July 2023 / Published online: 28 July 2023

Abstract
This work proposes an original preconditioner that couples the Constrained Pressure Residual (CPR) method with block preconditioning for the efficient solution of the linearized systems of equations arising from fully implicit multiphase flow models. This preconditioner, denoted as Block CPR (BCPR), is specifically designed for Lagrange multipliers-based flow models, such as those generated by Mixed Hybrid Finite Element (MHFE) approximations. An original MHFE-based formulation of the two-phase flow model is taken as a reference for the development of the BCPR preconditioner, in which the set of system unknowns comprises both element and face pressures, in addition to the cell saturations, resulting in a 3×3 block-structured Jacobian matrix with a 2×2 inner pressure problem. The CPR method is one of the most established techniques for reservoir simulations, but most research focused on solutions for Two-Point Flux Approximation (TPFA)-based discretizations that do not readily extend to our problem formulation. Therefore, we designed a dedicated two-stage strategy, inspired by the CPR algorithm, where a block preconditioner is used for the pressure part with the aim at exploiting the inner 2×2 structure. The proposed preconditioning framework is tested by an extensive experimentation, comprising both synthetic and realistic applications in Cartesian and non-Cartesian domains.

Keywords Constrained pressure residual · Block preconditioning · Two-phase flow in porous media · Mixed hybrid finite elements

Mathematics Subject Classification (2010) 65F08 · 65M22 · 65N08

1 Introduction

The physical processes involving the flow of multiple fluids in porous media are mathematically described by a set of coupled nonlinear Partial Differential Equations (PDEs), which are usually solved numerically. A typical solution approach consists of addressing the PDEs' inner coupling in a Fully Implicit (FI) manner, while a Newton scheme deals with the intrinsic nonlinearity. Although this strategy is robust and unconditionally stable, it demands the solution of several large-size linearized systems of equations with the Jacobian matrix at each time step, until satisfactory convergence is attained. Numerical evidence shows that this is the most time- and resource-consuming task in a simulation, typically requiring between 60 and 80% of the total CPU time [1–3]. While the use of other solvers than iterative Krylov subspace methods is in practice unfeasible, given the size of the systems generated in real-world applications, their performance depends mainly on the robustness and efficiency of the preconditioning strategy being supplied and the interplay that develops with the solver itself. This observation explains the importance of equipping Krylov solvers with efficient off-the-shelf preconditioners or developing a customized tool tailored to the application at hand whenever existing techniques are not robust. Either way, achieving good efficiency of the linear solver allows reducing the runtime and the

✉ Ahmad Abushaikha
aabushaikha@hbku.edu.qa

Stefano Nardean
stefano.nardean@unipd.it

Massimiliano Ferronato
massimiliano.ferronato@unipd.it

1 Division of Sustainable Development, College of Science and Engineering, Qatar Foundation, Hamad Bin Khalifa University, Doha, Qatar

2 Department of Civil, Environmental and Architectural Engineering, University of Padova, Padova, Italy

consumption of computational resources, while attaining the same solution accuracy.

The Constrained Pressure Residual (CPR) preconditioner [4, 5] is one of the most established strategies for both academic simulators and industrial software (see, for instance, [3, 6–17]). In its original structure, CPR is a two-stage technique based on the sequential application of two preconditioners for the pressure subproblem (local stage) and the whole Jacobian matrix (global stage), respectively. This strategy is underpinned by a physics-based intuition, i.e., the different characters of the pressure and saturation problems, the former being nearly elliptic and the latter hyperbolic. A consolidated scalable option for the preconditioning of the pressure block is Algebraic MultiGrid (AMG) [18, 19], which is very well-suited for elliptic problems, while incomplete factorization with zero fill-in (ILU(0)) is the usual choice for the global stage. Preliminary decoupling of pressure from saturation is also an option to improve the accuracy of the algorithm, with the possible downside of undermining the ellipticity of the original pressure problem [20].

Despite being almost forty-year-old, CPR is still the subject of extensive research (see, for instance, the recent review in [21]). One of the main topics consists of extending the CPR algorithm to general purpose reservoir simulators, e.g., [8, 22]. Roy et al. [22], in particular, developed a two-stage CPR-like algorithm for the nonisothermal dead-oil flow model, denoted as Constrained Pressure-Temperature Residual (CPTR). As the name suggests, temperature is elected as a primary variable together with pressure, so a restricted pressure/temperature problem is addressed in the local stage of the CPTR algorithm. Because of the inner 2×2 structure of this subproblem, a block preconditioner was developed, instead of relying on a single AMG approximation that can possibly prove ineffective. This is also pointed out in the work by Cremon et al. [8], where such a preconditioning approach was tested. In this article, which focuses on compositional simulations with thermal effects and reactions, another CPR-like algorithm, denoted as CPTR3, was proposed. It consists of three stages with two local AMG preconditioners for temperature and pressure alone, followed by a global ILU(0) sweep. The CPTR3 preconditioner has been benchmarked against the standard CPR algorithm, showing promising results.

Another research path fostering the integration of the CPR method and block preconditioners concerns the preconditioning of coupled flow/poromechanics problems addressed in an FI fashion. In this context, CPR can be included as a local preconditioner for the flow part within a global block preconditioning framework (see, for instance, [23, 24]).

Although the two-phase isothermal flow problem was the original target of the CPR development back in the 80s', it still represents a challenging bench test whenever novel and advanced schemes are used for the discretization of the problem PDEs. In this work, we considered a Mixed Hybrid Finite Element (MHFE) [25] approximation of Darcy's law, coupled with a standard time-implicit Finite Volume (FV) discretization of the mass balance. Following the approach developed in [26], and further applied in [27] and [28], in which the continuity of fluxes across the grid interfaces is strongly imposed to ensure the mass balance, the resulting problem is characterized by a 3×3 block-structured Jacobian matrix, owing to the simultaneous presence of pressure variables defined on both cells and faces, in addition to the saturation unknowns computed on elements. This peculiar matrix format differs from the usual 2×2 structure generated by Two-Point-Flux-Approximation (TPFA)-based discretizations.

Applying the classical two-stage CPR algorithm, which was precisely designed for TPFA, to such a problem would require approximating the non-symmetric 2×2 pressure subproblem with an AMG preconditioner, which is often not expected to be a robust option. On the contrary, its internal block structure can be exploited with the introduction of a block preconditioner as a local stage within a broader CPR-like algorithm. The resulting method is denoted as Block CPR (BCPR). A typical strategy for the preconditioning of the systems of equations stemming from single-phase MHFE-based flow simulations, whose 2×2 block structure resembles that of our pressure subproblem, consists of eliminating the element pressure unknowns by performing static condensation (see, for instance, [29, 30]). Then, the resulting matrix can be preconditioned by means of off-the-shelf preconditioners. However, this strategy cannot be applied to our problem formulation and, more generally, to problems with compressible fluids since, unlike the single-phase flow setting, the element pressure block is not diagonal.

In this work, we propose an original approach for addressing the element and face pressure equations. Building on our previous work on block preconditioners for the single-phase flow problem [31, 32], we developed a tool based on the exploitation of approximate versions of the Jacobian matrix decoupling factors to obtain an inexact version of the Schur complement. Although fully parallelizable, approximating such factors is the most expensive task in our algorithm; however, it does not need to be performed at each nonlinear iteration, but only once at the outset of the simulation. Therefore, the associated cost can be amortized throughout the entire simulation.

This article extends the piece of work first presented at the ECMOR 2022 conference in [33]. A breakdown of the paper structure is as follows. The preconditioning methodology is described and tested in four applications after the mathematical and numerical model problems are presented. The paper concludes with a discussion of the findings and some preliminary suggestions for further investigation.

2 Mathematical problem: two-phase flow in porous media

The two-phase flow in porous media, underpinned by some simplifications of the physics, is the model problem considered in this study. The fluids (oil, o, and water, w), flowing in a compressible medium under isothermal conditions, are assumed to be immiscible and incompressible. Capillarity is also neglected in the model, resulting in the equality of the fluid pressures, i.e., $p_o = p_w = p$. Pressure in the reservoir, p, the well flowing pressure, p_{wf}, and saturation of the wetting phase (water), S_w, are elected as model unknowns, consistently with the natural variables' formulation [34].

2.1 Governing equations

The set of PDEs describing the flow of multiple fluids in porous media consists of the mass balance equation

$$\frac{\partial \phi S_\alpha}{\partial t} + \nabla \cdot \boldsymbol{v}_\alpha = f_\alpha, \qquad \alpha = o, w \quad \text{in } \Omega \times \mathbb{T}, \tag{1}$$

and Darcy's law

$$\boldsymbol{v}_\alpha = -\lambda_\alpha K \nabla (p - \gamma_\alpha z), \qquad \alpha = o, w \quad \text{in } \Omega \times \mathbb{T}, \tag{2}$$

written for each phase α in the space and time domains, Ω and $\mathbb{T}$, respectively. In Eq. 1, ϕ denotes the porosity of the rock, S_α and $\boldsymbol{v}_\alpha$ are the saturation and velocity of phase α, t is time, and f_α is the source/sink term, while, in Eq. 2, K defines the permeability tensor, γ_α the specific weight, and z is depth. Furthermore, $\lambda_\alpha = k_{r\alpha}(S_\alpha)/\mu_\alpha$ denotes the phase mobility factor, where $k_{r\alpha}(S_\alpha)$ and μ_α are the relative permeability and phase dynamic viscosity, respectively. The factor λ_α depends on saturation, however, due to the fluid incompressibility assumption, it does not change with pressure, i.e., μ_α is constant. The term f_α introduces in the reservoir model the action of wells, which are reproduced using the classical Peaceman formulation [35, 36].

Additional relationships need to be provided to mathematically close the problem outlined by Eqs. 1 and 2. Specifically, we relied on the analytical Corey's model [37, 38] for the relative permeability and the classical relationship

$$\phi = \phi^0 \left[1 + c_r \left(p - p^0\right)\right], \tag{3}$$

which relates porosity to the change in pore pressure experienced by the rock [39]. Here, c_r is the solid phase compressibility and the superscript 0 denotes the initial conditions of the relevant quantities. The constraint on the sum of the phase saturations

$$\sum_{\alpha=o,w} S_\alpha = 1 \tag{4}$$

is also an equation of the system. Ultimately, appropriate initial conditions on pressure and saturation,

$$p|_{t=0} = p^0 \qquad \text{in } \overline{\Omega}, \tag{5a}$$

$$S_w|_{t=0} = S_w^0 \qquad \text{in } \overline{\Omega}, \tag{5b}$$

along with boundary conditions on pressure and fluid fluxes,

$$p = \overline{p} \qquad \text{on } \Gamma_p \times \mathbb{T}, \tag{6a}$$

$$-\lambda_\alpha K \nabla (p - \gamma_\alpha z) \cdot \boldsymbol{n} = \overline{v_{n\alpha}} \qquad \text{on } \Gamma_{\boldsymbol{v}} \times \mathbb{T}, \tag{6b}$$

need to be prescribed to give rise to a well-posed mathematical model. In Eqs. 5 and 6, Γ_p and $\Gamma_{\boldsymbol{v}}$ are the portions of the boundary, Γ, where Dirichlet and Neumann conditions are enforced, respectively, with $\Gamma_p \cup \Gamma_{\boldsymbol{v}} = \Gamma$, and $\overline{\Omega} = \Omega \cup \Gamma$, while $\boldsymbol{n}$ denotes the outer normal vector to Γ.

2.2 Weak formulation

Let $\mathcal{E}^h$ be the collection of non-overlapping hexahedral cells discretizing the reservoir body with $\mathcal{F}^h$ denoting the set of grid faces. The weak form of Darcy's law in Eq. 2 is constructed by means of the MHFE method on the lowest-order $\mathbb{RT}_0$ space [40], which requires appending pressure unknowns to the center of gravity of both elements and faces, p^E and π, respectively. Saturation is instead solely computed on the cells centroid. Specifically, the $\mathbb{P}_0$ space with support on the collection of cells is used to reproduce the element pressure and water saturation fields, whereas another $\mathbb{P}_0$ space, defined on faces, plays a similar role for the interface pressure. The 3-D velocity field is approximated in the $\mathbb{RT}_0$ space, which is spanned by piecewise trilinear vector functions, $\boldsymbol{\eta}_i^E(x, y, z)$, defined for every face i and element E with support restricted to the cell itself. Darcy's velocity on E is thus expressed in the form:

$$\boldsymbol{v}_\alpha^h|_E = \sum_{\ell \in \partial E} \boldsymbol{\eta}_\ell^E(x, y, z) q_{\alpha,\ell}^E, \qquad \alpha = o, w, \tag{7}$$

where the weight, $q_{\alpha,\ell}^E$, represents the α-phase flux across face ℓ and ∂E is the collection of element interfaces. The face unknowns π are the hallmark of the MHFE approach over the original Mixed finite element scheme, whose introduction was intended to guarantee the continuity of the normal components of the interface fluxes, and they operate as Lagrange multipliers.

The MHFE weak form of Eq. 2, expressed for every element of the grid, describes the fluid fluxes across its interfaces based on the pressure and gravitational gradients experienced within the cell itself as [26]:

$$\boldsymbol{q}_\alpha^E = \lambda_\alpha^* \left(B^E\right)^{-1} \left[p^E \mathbf{1} - \boldsymbol{\pi}^E - \gamma_\alpha \left(z^E \mathbf{1} - \boldsymbol{\zeta}^E\right)\right], \qquad \alpha = o, w. \qquad (8)$$

Here, the vectors $\boldsymbol{q}_\alpha^E$, $\boldsymbol{\pi}^E$, $\boldsymbol{\zeta}^E \in \mathbb{R}^{n_f^E}$ collect the fluxes of phase α, face pressures and relevant depths, respectively, with n_f^E being the number of element faces. Furthermore, $\mathbf{1} \in \mathbb{R}^{n_f^E}$ denotes the vector of ones, λ_α^* is the diagonal matrix containing the fluid mobility of the *upstream* element for each face, and $B^E \in \mathbb{R}^{n_f^E \times n_f^E}$ is the MHFE elementary matrix. The entries of B^E are defined as:

$$B_{ij}^E = \int_{\Omega^E} \left(\boldsymbol{\eta}_i^E\right)^T \left(K^E\right)^{-1} \boldsymbol{\eta}_j^E \, d\Omega, \qquad \forall i, j \in \partial E, \qquad (9)$$

where Ω^E denotes the cell volume. The expression for the interface fluxes in Eq. 8 is all in all similar to that generated by a Mimetic finite difference discretization (see, for instance, [27, 41]).

In this section, we limited ourselves to presenting just the main aspects of the MHFE method applied to Eq. 2 in a multiphase setting; the interested reader is referred, for instance, to the works [10, 32, 42–44] for the full mathematical derivation and further insights.

On the other hand, a classical FV method in space, coupled with a first-order implicit Euler scheme for the time discretization, is used to build the weak form of the mass balance (1). This is a rather standard approach resulting in the approximate form:

$$\Omega^E \frac{\phi\left(p_n^E\right) S_{\alpha,n}^E - \phi\left(p_{n-1}^E\right) S_{\alpha,n-1}^E}{\Delta t_{n-1}} + \sum_{i \in \partial E_r} q_{\alpha,i}^{E,E'} - \Omega^E f_\alpha^E = 0, \qquad \forall E \in \mathcal{E}^h, \qquad (10)$$

where $n-1$ and n denote the previous and actual time steps, respectively, $\Delta t_{n-1} = t_n - t_{n-1}$ is the time increment, and ∂E_r is the set of faces of E not included in the boundary. This implies that the reservoir is isolated from the surrounding rock, which is a frequent assumption in reservoir modeling. Moreover,

$$q_{\alpha,i}^{E,E'} = \lambda_{\alpha,i}^* \frac{\left(B^{E'}\right)_{ii}^{-1} \Lambda_{\alpha,i}^E - \left(B^E\right)_{ii}^{-1} \Lambda_{\alpha,i}^{E'}}{\left(B^E\right)_{ii}^{-1} + \left(B^{E'}\right)_{ii}^{-1}} \qquad (11)$$

is the flux across the i-th face separating elements E and E', which results from strongly imposing the continuity of the local fluxes, expressed as in Eq. 8, on the two sides of interface i, i.e., $q_{\alpha,i}^E$ and $q_{\alpha,i}^{E'}$ [26]. In Eq. 11, we have

$$\Lambda_{\alpha,i}^E = L_{B_i^E}\left(p^E - \gamma_\alpha z^E\right) - \sum_{j \in \partial E \setminus \{i\}} \left(B^E\right)_{ij}^{-1} \left(\pi_j^E - \gamma_\alpha \zeta_j^E\right)$$

with $L_{B_i^E} = \sum_{j \in \partial E} \left(B^E\right)_{ij}^{-1}$.

Wells are reproduced using the standard Peaceman model. They can include multiple perforations and can be operated either by a fixed rate or Bottom Hole Pressure (BHP) control. Switching between the two controls is possible though not used in the following tests. The phase flux exchanged by the i-th perforation with the hosting cell reads [36]:

$$f_{\alpha,i} = \lambda_{\alpha,i}^* WI(p_{res} - p_i), \qquad (12)$$

where p_{res} is the reservoir pressure at the hosting block, and p_i is the well pressure in the perforation, including also the gravity contribution exerted by the fluids in the above portion of the well, evaluated with a segmented gravity head technique. In Eq. 12, WI is the well index which is defined as (assuming that the well is vertical):

$$WI = \frac{2\pi \Delta z \sqrt{k_x k_y}}{\ln\left(\frac{r_e}{r_w}\right) + s}. \qquad (13)$$

Here, Δz is the cell thickness, k_x and k_y are the permeability values along the x and y axes, s is the skin factor, r_w is the well radius and r_e is the equivalent radius:

$$r_e = 0.28 \frac{\left[\left(\frac{k_y}{k_x}\right)^{\frac{1}{2}} \Delta x^2 + \left(\frac{k_x}{k_y}\right)^{\frac{1}{2}} \Delta y^2\right]^{\frac{1}{2}}}{\left(\frac{k_y}{k_x}\right)^{\frac{1}{4}} + \left(\frac{k_x}{k_y}\right)^{\frac{1}{4}}}, \qquad (14)$$

where Δx and Δy are the x and y sizes of the hosting cell. Depending on the type of well control, the equations to be included in the discrete system read:

$$\begin{cases} p_{wf} - \overline{p_{wf}} = 0 & \text{(BHP control)} \\ \sum_{\alpha=o,w} \sum_{i=1}^{n_p} f_{\alpha,i} - \overline{f} = 0 & \text{(rate control)} \end{cases}, \qquad (15)$$

where $\overline{p_{wf}}$ and $\overline{f}$ are the prescribed BHP and total flow rate, respectively, and n_p is the number of perforations.

2.2.1 The MHFE-FV system of equations

The governing equations, introduced in the previous section, define a coupled nonlinear problem, $\boldsymbol{R}^n = \boldsymbol{0}$, at each time step, which is addressed here by an FI strategy. The system equations can be broken down into three groups: $\boldsymbol{R}_\pi \in \mathbb{R}^{n_f}$, $\boldsymbol{R}_p \in \mathbb{R}^{n_E+n_w}$, and $\boldsymbol{R}_s \in \mathbb{R}^{n_E}$, where n_f, n_E, and n_w are the number of faces, elements, and wells in the model. Note that the time step counter n has been dropped in the vectors above to compact the notation. The first set, $\boldsymbol{R}_\pi = \boldsymbol{0}$, aims at ensuring that the total fluxes remain constant across the grid faces, i.e.,

$$\sum_{\alpha=o,w} \left(q^E_{\alpha,i} + q^{E'}_{\alpha,i} \right) = 0, \qquad \forall i \in \mathcal{F}^h. \tag{16}$$

The second set, $\boldsymbol{R}_p = \boldsymbol{0}$, consists of the so-called Implicit Pressure Explicit Saturation (IMPES)-like *pressure equations*, obtained by summing Eq. 10 over the phase index for each cell, in addition to the well constraints, which are included here due to the expected low number of wells as compared to the number of elements. Ultimately, the set $\boldsymbol{R}_s = \boldsymbol{0}$ collects the so-called *saturation equations*, i.e., the n_E discretized mass balance equations (Eq. 10) written for the water phase. The overall size of the nonlinear system of equations, $\boldsymbol{R}^n = [\boldsymbol{R}_\pi,\ \boldsymbol{R}_p,\ \boldsymbol{R}_s]^{n,T} = \boldsymbol{0}$, is $2n_E + n_f + n_w$, with n_E, n_f, and n_w pressure unknowns computed on elements, faces, and wells, respectively, and n_E saturations.

The nonlinearity of $\boldsymbol{R}^n = \boldsymbol{0}$ is addressed by a classical Newton scheme

$$\boldsymbol{x}^{n,(m)} = \boldsymbol{x}^{n,(m-1)} + \delta \boldsymbol{x}, \tag{17}$$

where $\boldsymbol{x}^{n,(m)} = [\boldsymbol{x}_\pi,\ \boldsymbol{x}_p,\ \boldsymbol{x}_s]^{n,(m),T}$ is the full set of unknowns arranged in homogeneous groups and m is the Newton iteration counter. In Eq. 17, the solution update $\delta\boldsymbol{x}$ is computed by solving the linearized system of equations:

$$\mathcal{J}^{n,(m-1)} \delta \boldsymbol{x} = -\boldsymbol{R}^{n,(m-1)} \quad \Rightarrow \\ \begin{bmatrix} J_{\pi\pi} & J_{\pi p} & J_{\pi s} \\ J_{p\pi} & J_{pp} & J_{ps} \\ J_{s\pi} & J_{sp} & J_{ss} \end{bmatrix}^{n,(m-1)} \begin{bmatrix} \delta\boldsymbol{x}_\pi \\ \delta\boldsymbol{x}_p \\ \delta\boldsymbol{x}_s \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}_\pi \\ \boldsymbol{R}_p \\ \boldsymbol{R}_s \end{bmatrix}^{n,(m-1)}, \tag{18}$$

where $\mathcal{J}^{n,(m-1)} = \frac{\partial \boldsymbol{R}^{n,(m-1)}}{\partial \boldsymbol{x}}$ is the Jacobian matrix exhibiting a 3×3 block structure, and $\boldsymbol{R}^{n,(m-1)}$ is the residual vector, both taken at the $(m-1)$-th iteration. As to the size of the diagonal blocks, we have $J_{\pi\pi} \in \mathbb{R}^{n_f \times n_f}$, $J_{pp} \in \mathbb{R}^{(n_E+n_w)\times(n_E+n_w)}$, and $J_{ss} \in \mathbb{R}^{n_E \times n_E}$. Before applying the solution update in Eq. 17, $\delta\boldsymbol{x}$ is properly post-processed with the Appleyard chop technique [13, 45], which provides a sort of guidance and stabilization to the behavior of the nonlinear solver. The iterative process in Eq. 17 is stopped whenever an appropriate norm of the residual (see also Section 4 for the specific criterion implemented in the simulator) falls below a given threshold.

Inspection of the Jacobian matrix reveals that $J_{\pi s} = 0$ when gravity is neglected, otherwise all blocks are non-zero. Moreover, under the same condition, $J_{\pi p}$ and $J_{\pi\pi}$ do not depend on λ_α. This means that these blocks remain constant throughout the simulation with $J_{\pi\pi}$ being also Symmetric Negative Definite (SND), while the whole Jacobian is always non-symmetric, regardless of whether gravitational forces are included in the model or not. Block J_{pp} is mildly non-symmetric, positive definite and diagonally dominant, while J_{ss} is the typical matrix arising from a transport problem, therefore non-symmetric as well. For the design of the BCPR preconditioner, it is convenient to rearrange the Jacobian matrix into a 2×2 structure by grouping together the pressure blocks, yielding:

$$\mathcal{J} = \begin{bmatrix} J_{PP} & J_{Ps} \\ J_{sP} & J_{ss} \end{bmatrix}, \tag{19}$$

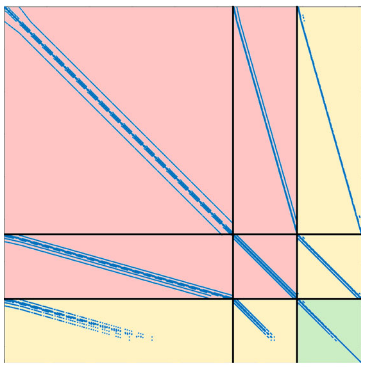
where

$$J_{PP} = \begin{bmatrix} J_{\pi\pi} & J_{\pi p} \\ J_{p\pi} & J_{pp} \end{bmatrix}, \qquad J_{Ps} = [J_{\pi s}\ J_{ps}]^T, \\ J_{sP} = [J_{s\pi}\ J_{sp}]. \tag{20}$$

A sketch of the non-zero pattern of $\mathcal{J}$ is shown in Fig. 1a.

3 The Block CPR (BCPR) preconditioner

The CPR-like family of preconditioners is one of the most successful tools for the efficient preconditioning of linearized systems of equations in reservoir simulations. The use of TPFA for the inter-element fluxes approximation, with a single pressure unknown per cell, influenced the structure of the CPR algorithm, where a computationally efficient and scalable choice for the approximation of the elliptic pressure block alone is offered by AMG. However, applying the classical CPR algorithm to the linearized system in Eq. 18 is not optimal since J_{PP} is a 2×2 non-symmetric block matrix, compounding both element and face variables, for which an approximation by means of existing off-the-shelf AMG functions is often ineffective. On the contrary, such a block structure can be exploited by replacing AMG for the whole pressure part with a block preconditioner relying, in turn, on AMG approximations for the application of the


(a) Non-zero pattern of $\mathcal{J}$ as per equations (19) and (20).

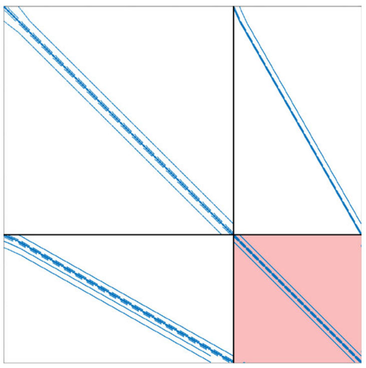

(b) Non-zero pattern of $\mathcal{J}$ after element-wise reordering of $\boldsymbol{x}_p$ and $\boldsymbol{x}_s$ unknowns.

Fig. 1 Non-zero pattern of $\mathcal{J}$ with different unknowns ordering. In panel (b), the block in position (2,2), highlighted in red, roughly corresponds to the system matrix obtained from a standard TPFA-FV discretization with its typical seven-point stencil. The off-diagonal blocks express the coupling between element and face unknowns. Notice that block (2,1) is denser than (1,2) as a result of strongly imposing the flux continuity in the mass balance equations

leading block and Schur complement. In essence, we use the classical two-stage CPR multiplicative framework:

$$\mathcal{M}_{\text{BCPR}}^{-1} = \mathcal{M}_2^{-1}\left[I - \mathcal{J}\mathcal{M}_1^{-1}\right] + \mathcal{M}_1^{-1}, \tag{21}$$

where

$$\mathcal{M}_1^{-1} = \text{diag}\,(\mathcal{J})^{-1} \tag{22}$$

and

$$\mathcal{M}_2^{-1} = \begin{bmatrix} \mathcal{M}_{PP}^{-1} & \\ & 0_s \end{bmatrix} \tag{23}$$

are the first- and second-stage preconditioners, respectively. In Eq. 22, diag $(\mathcal{J})^{-1}$ is the Jacobi preconditioner applied to $\mathcal{J}$, while, in Eq. 23, $\mathcal{M}_{PP}^{-1}$ denotes the block preconditioner for the overall pressure subproblem and 0_s is the zero matrix in the space of saturations.

By inspecting Eqs. 21-23, we notice that two modifications, in addition to the block preconditioner for the pressure subproblem, have been introduced with respect to the most classical CPR framework:

1. The order of the stages is inverted, i.e., the global stage is carried out before the local one. This is done following the suggestion in [22] (and previously applied in [46]) in order to provide some decoupling of pressure from saturation. The decoupling task, in fact, is not explicitly performed in a preliminary stage, as is often done in practice (for instance with the aid of IMPES or Quasi-IMPES strategies [16]), so as to preserve the original algebraic properties and stencil of the J_{PP} block, which is key for building $\mathcal{M}_{PP}^{-1}$;
2. The global stage is carried out by a simple Jacobi preconditioner instead of the classical ILU(0). The system in Eq. 18 embraces both element and face unknowns and their ordering makes it difficult to build a matrix with a compact band, which is an important property for incomplete factorizations. Typically, in FV-based models, this matrix structure is achieved by arranging the unknowns with an element-wise ordering, i.e., by interleaving the element pressure variables with the saturation unknowns, $[p_1, S_{w,1}, p_2, S_{w,2}, \ldots, p_{n_E}, S_{w,n_E}]$. Applying this ordering strategy to the set of $\boldsymbol{x}_p$ and $\boldsymbol{x}_s$ unknowns in the system of Eq. 18 (see Fig. 1b) does not provide an effective ILU(0) approximation and often may introduce significant round-off errors, as some numerical tests, not reported here, pointed out. Although other remedies may include allowing some fill-in in the ILU decomposition or applying algebraic reordering techniques, we found that the inexpensive Jacobi preconditioner can suffice for the satisfactory performance of the BCPR preconditioner, as we will comment more extensively in Section 4.1. This also appears to be favorable in view of the massive parallelization of the algorithm.

The starting point for the design of a block preconditioner for J_{PP} is its $\mathcal{LDU}$ decomposition. By introducing proper approximations for the application of the inverse of $J_{\pi\pi}$ and the Schur complement $S = J_{pp} - J_{p\pi}J_{\pi\pi}^{-1}J_{\pi p}$, the block preconditioner for the pressure subproblem takes the form:

$$\mathcal{M}_{PP}^{-1} = \mathcal{U}^{-1}\mathcal{D}^{-1}\mathcal{L}^{-1} = \begin{bmatrix} I_\pi & -\widetilde{J}_{\pi\pi}^{-1}J_{\pi p} \\ & I_p \end{bmatrix}\begin{bmatrix} \widetilde{J}_{\pi\pi}^{-1} & \\ & \widetilde{S}^{-1} \end{bmatrix}\begin{bmatrix} I_\pi & \\ -J_{p\pi}\widetilde{J}_{\pi\pi}^{-1} & I_p \end{bmatrix}, \tag{24}$$

where the superscript $\sim$ denotes an approximate term. Approximating the second part of the Schur complement is usually troublesome, as the unknown and dense term $J_{\pi\pi}^{-1}$ is involved. Building on the Explicit Decoupling Factor

Approximation (EDFA) preconditioner, developed in our previous works [31, 32], we rewrite the Schur complement as

$$S = J_{pp} + J_{p\pi} F, \tag{25}$$

where

$$F = -J_{\pi\pi}^{-1} J_{\pi p} \tag{26}$$

is the upper decoupling factor in Eq. 24. The F term can be computed by solving a series of Multiple Right-Hand Side (MRHS) systems with the columns of $J_{\pi p}$ and $J_{\pi\pi}$ as a matrix:

$$-J_{\pi\pi} F = J_{\pi p}. \tag{27}$$

Obviously, the exact calculation of the dense F factor is computationally intensive, so an approximation is sought. In particular, to preserve a workable sparsity, we compute each column of F only at certain locations belonging to a given non-zero pattern $\mathcal{P}$. Focusing on the approximation of column q of F, with $1 \leq q \leq n_E$, let $R_r^{(q)}$ be the restriction operator to the rows belonging to the q-th non-zero pattern, $\mathcal{P}^{(q)}$, and $\boldsymbol{l}^{(q)}$ be the corresponding vector of the canonical basis. The approximation of the column in the restricted space is found by solving the following (small) system:

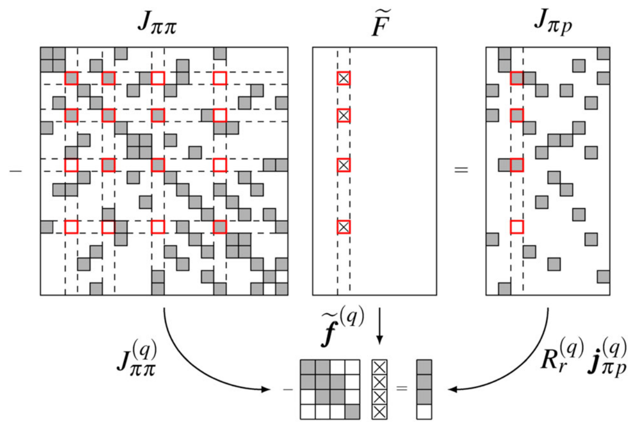
$$-J_{\pi\pi}^{(q)} \widetilde{\boldsymbol{f}}^{(q)} = R_r^{(q)} \boldsymbol{j}_{\pi p}^{(q)}, \tag{28}$$

where $J_{\pi\pi}^{(q)} = R_r^{(q)} J_{\pi\pi} \left(R_r^{(q)}\right)^T$ and $\boldsymbol{j}_{\pi p}^{(q)} = J_{\pi p} \boldsymbol{l}^{(q)}$. The inexact factor, $\widetilde{F}$, is obtained by gathering all the $\widetilde{\boldsymbol{f}}^{(q)}$ contributions after being prolonged to the original space:

$$\widetilde{F} = \sum_{q=1}^{n_E} \left(R_r^{(q)}\right)^T \widetilde{\boldsymbol{f}}^{(q)} \left(\boldsymbol{l}^{(q)}\right)^T. \tag{29}$$

The sequence of operations to compute $\widetilde{F}$ is sketched in Fig. 2. Notice that $J_{\pi p} \in \mathbb{R}^{n_f \times (n_E + n_w)}$ so F must share the same size. However, the well and flux continuity equations are mutually decoupled, hence the last n_w columns in $J_{\pi p}$ are zero and we can avoid solving the relevant homogeneous systems in Eq. 28. This explains the upper limit for q in Eq. 29.

For the approximation $\widetilde{F}$ to be effective, it is crucial to identify the locations of the most significant entries in each column and collect them to form the non-zero pattern, while preserving an adequate sparsity. To this end, we developed two techniques, denoted as *static* and *dynamic*, respectively, the former being more physics-based and the latter fully algebraic.



$J_{\pi\pi}$
$\widetilde{F}$
$J_{\pi p}$
$-$
$=$
$J_{\pi\pi}^{(q)}$
$\widetilde{\boldsymbol{f}}^{(q)}$
$R_r^{(q)} \boldsymbol{j}_{\pi p}^{(q)}$

Fig. 2 Column-wise approximation of the F decoupling factor. The red squares with a cross in the $\widetilde{F}$ matrix denote the locations of the selected nonzero entries, which form the q-th column nonzero pattern

The static approach is underlain by a graphical interpretation, for which each entry in the $\boldsymbol{j}_{\pi p}^{(q)}$ and $\widetilde{\boldsymbol{f}}^{(q)}$ vectors is related to a grid face and the non-zeros in $\boldsymbol{j}_{\pi p}^{(q)}$ correspond to the faces of element q, as shown in Fig. 3a. Moreover, the problem formalized in Eq. 28 is roughly equivalent to solving a flow problem in a portion of the domain whose shape and size are defined by the nonzero pattern. Therefore, starting from $\boldsymbol{j}_{\pi p}^{(q)}$'s sparsity pattern, we can customize a broad range of compact patches by including neighboring interfaces such as those displayed in Fig. 3. The patterns depend on the number of levels of the connection, i.e., how many elements in addition to the central one are involved in each direction (from zero in Fig. 3a to four in 3g), and the presence of the lateral faces, while isotropy in the three directions is preserved. Selecting an appropriate pattern such that the shape and size of the resulting subdomain, where the flow problem is solved, allows for capturing the propagation of the main pressure gradients is key for an accurate approximation of F. With the static technique, a prototype pattern is thus supplied beforehand and applied to the whole domain. Numerical tests in [32] showed that this strategy is effective when the grid is Cartesian and the modeler has a robust idea of the possible flow field.

On the other hand, the dynamic technique relies on a recursive algorithm, during which additional column non-zero entries are introduced at the locations where the components of the residual $\boldsymbol{r} = \boldsymbol{j}_{\pi p}^{(q)} + J_{\pi\pi} (R_r^{(q)})^T \widetilde{\boldsymbol{f}}^{(q)}$ are larger in absolute value, starting from the initial non-zero pattern of $\boldsymbol{j}_{\pi p}^{(q)}$. Despite being more expensive, the dynamic approach is more flexible and can be used as a black box. The parameters controlling the dynamic process are (i) n_{ent}, the number of new entries added to the original pattern, and (ii) n_{add}, the maximum number of entries introduced at each pattern augmentation step.

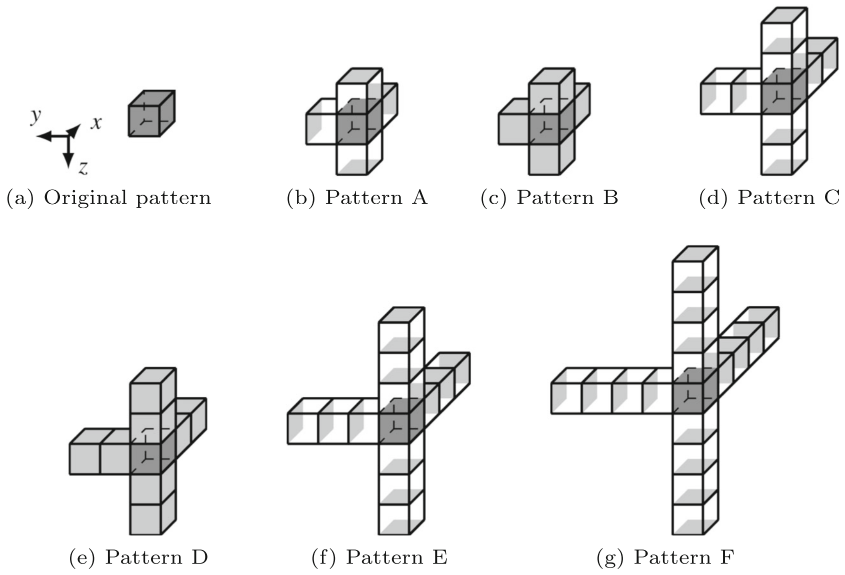


Fig. 3 Collection of patterns for the EDFA static approach. Notice that the front and right elements have been removed to improve the picture clarity

Algorithm 1 Application of the BCPR preconditioner: $[\boldsymbol{v}] = \text{apply\_BCPR}(\tau_i, \mathcal{J}, \mathcal{M}_1^{-1}, \mathcal{M}_2^{-1}, \boldsymbol{w})$.

```
1: $\boldsymbol{v} = \mathcal{M}_1^{-1}\boldsymbol{w}$    ▷ Application of the first-stage preconditioner
2: $\boldsymbol{r} = \boldsymbol{w} - \mathcal{J}\boldsymbol{v}$    ▷ Computing the residual
3: $[\delta\boldsymbol{v}] = \text{apply\_2\_stage}(\tau_i, J_{PP}, \widetilde{S}, \boldsymbol{r})$    ▷ Application of the second-stage preconditioner to $\boldsymbol{r} = [\boldsymbol{r}_\pi, \boldsymbol{r}_p, \boldsymbol{r}_s]$, i.e., $\delta\boldsymbol{v} = \mathcal{M}_2^{-1}\boldsymbol{r}$
4: $\boldsymbol{v} \leftarrow \boldsymbol{v} + \delta\boldsymbol{v}$    ▷ Correction of the first guess
```

Algorithm 2 Application of the second-stage block preconditioner: $[\delta\boldsymbol{v}] = \text{apply\_2\_stage}(\tau_i, J_{PP}, \widetilde{S}, \boldsymbol{r})$.

```
1: $\boldsymbol{t}_\pi = \text{AMG}(J_{\pi\pi}, \boldsymbol{r}_\pi)$
2: $\boldsymbol{t}_p = \boldsymbol{r}_p - J_{p\pi}\boldsymbol{t}_\pi$
3: $\delta\boldsymbol{v}_p = \text{solve\_AMG}(\tau_i, \widetilde{S}, \boldsymbol{t}_p)$
4: $\delta\boldsymbol{v} = [\boldsymbol{0}, \delta\boldsymbol{v}_p, \boldsymbol{0}]$
5: $\boldsymbol{t}_\pi = \boldsymbol{r}_\pi - J_{\pi p}\delta\boldsymbol{v}_p$
6: $\delta\boldsymbol{v}_\pi = \text{AMG}(J_{\pi\pi}, \boldsymbol{t}_\pi)$
7: $\delta\boldsymbol{v} \leftarrow \delta\boldsymbol{v} + [\delta\boldsymbol{v}_\pi, \boldsymbol{0}, \boldsymbol{0}]$
```

The action of the BCPR preconditioner to a vector $\boldsymbol{w}$ is shown in Algorithm 1, while Algorithm 2 details the application of the block preconditioner at the local stage. AMG, in particular the aggregation-based AGMG presented in [47–49], is used to approximate block $J_{\pi\pi}$ and the inexact Schur complement

$$\widetilde{S} = J_{pp} + J_{p\pi}\widetilde{F}. \tag{30}$$

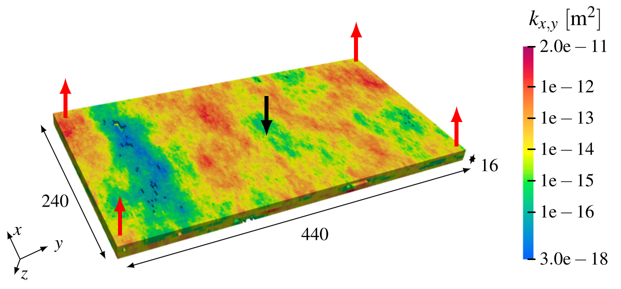
While a single V-cycle is enough for the application of $\widetilde{J}_{\pi\pi}^{-1}$, an AMG-preconditioned inner Generalized Conjugate Residual (GCR) method [50, 51] with a loose exit tolerance τ_i in the range [1.E-5,1.E-4] (achieved in less than 15 iterations) is used for the application of $\widetilde{S}^{-1}$. Function solve_AMG in Algorithm 2 performs this task.

The choice of using the AGMG function has been made out of convenience since, as opposed to many tools available in the literature, it offers a ready-to-use Matlab interface. In fact, the code implementing the two-phase flow model in compressible media, described in Section 2, along with the proposed BCPR preconditioner, has been prototyped using Matlab. In the future, when transitioning to an implementation of the BCPR preconditioner with a compiled programming language, more mainstream AMG tools, such as those available in hypre [52] or PETSc [53, 54] libraries, will be considered.

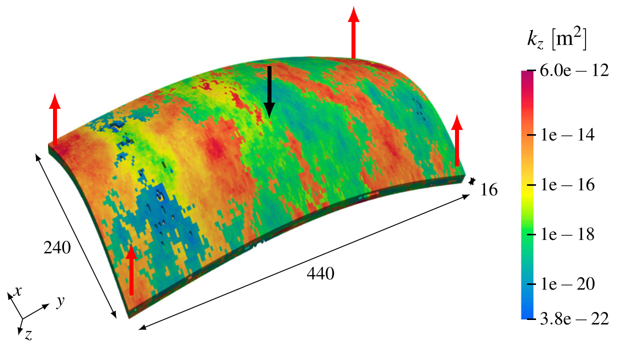
In general, the behavior of the global BCPR method depends upon several elements that can be modified or tuned to improve its effectiveness, such as the local- and global-stage preconditioners, the specific AMG tool for the application of the inverse of $J_{\pi\pi}$, as well as the approximation of the Schur complement and of its inverse. In this development stage, we focus on the computation of $\widetilde{S}$, setting the other elements as discussed above.

4 Numerical results

The performance of the BCPR preconditioner has been investigated in four test cases, built on the first four layers of the SPE10 data set [55] and obtained by varying the original shape of the domain and rock properties. The cell size is set equal to 4 × 2 × 4 m and all the elements with a porosity value smaller than 1.E-5 have been preliminarily removed from the mesh. This cleaning step generates a Swiss cheese-like grid visible, for instance, in Fig. 4, where the reader can also see the production scenario and one of the permeability distributions used in the experimental part of the work. The domain in Tests 1-2(a,b) is planar and discretized with a Cartesian mesh, while in Test 3 it has been deformed into an anticline structure with a non-Cartesian tessellation,



(a) Cartesian grid used in Tests 1-2(a,b) with the horizontal permeability field from the SPE10 data set.



(b) Non-Cartesian grid used in Test 3 with the vertical permeability field from the SPE10 data set.

Fig. 4 The two grids used in the tests with a sketch of the simulated scenario. The red and black arrows show the location of production and injection wells, respectively. The horizontal and vertical permeability distributions are superimposed on the grids

which can also be classified as *corner-point* in the reservoir simulation community. Such deformation is symmetric with respect to the horizontal symmetry axes of the grid with a maximum rotation of 19° and 32° around the x and y directions, respectively. The number of elements n_E and faces n_f is the same for both discretizations and equal to 51,741 and 171,070, respectively. The simulated scenario, i.e., the classical five-spot injection/production pattern with an injector in the middle of the reservoir and a producer at each corner, is the same for all tests. The injector operates at a constant rate equal to 20 m^3/d and the producers at a constant BHP of 490 kPa for Tests 1, 2a, and 3 and 3,200 kPa for Test 2b. The wells fully penetrate the reservoir thickness and their radius is equal to 0.1 m. A summary of the tests' setup is offered in Table 1.

The objective of Test 1 is to introduce the BCPR preconditioner and to define a baseline for its setup. To this end, we apply a homogeneous and isotropic permeability distribution ($k_{x,y,z}$ = 1.E-12 m^2), while the initial porosity is uniform throughout the domain (ϕ^0 = 2.5E-1), and gravity is neglected. Conversely, Tests 2 and 3 are numerically much more challenging. A fully heterogeneous and anisotropic permeability field with nonuniform porosity is introduced in Tests 2a and 2b, which represent a challenging bench test for the performance of the BCPR preconditioner. In Test 2b, gravity is also enabled to assess the sensitivity of our preconditioning tool to this feature. Finally, in Test 3, a heterogeneous permeability distribution, taken from the SPE10 data set, with an anisotropy ratio up to 3,300, is applied to the dome-structured domain. The objective of this test is to evaluate the performance of the BCPR preconditioner in a realistic setting and point out the differences in its setup when moving from a Cartesian to a non-Cartesian grid.

The linearized systems in Eq. 18 are solved by means of a right-preconditioned full GMRES [56]. Given the relatively low iteration number achieved during the tests, the use of full GMRES appears to be fully warranted. The iterative process is stopped whenever the 2-norm of the relative residuals of the Jacobian system falls below a user-defined threshold, τ_l, i.e., $\|\boldsymbol{r}^k\|_2/\|\boldsymbol{r}^0\|_2 < \tau_l$. For all tests, τ_l = 1.E-6. The termination criterion for the nonlinear solver relies on the evaluation of both the absolute and relative residual. In particular, given the different nature of the variables, the residual is broken down into three parts, with the iteration process ending when one of the following conditions is satisfied:

$$\max\left\{\|\boldsymbol{R}_\pi^{n,(m)}\|_2,\ \|\boldsymbol{R}_p^{n,(m)}\|_2,\ \|\boldsymbol{R}_s^{n,(m)}\|_2\right\} < \tau_{nl,a},$$

$$\max\left\{\|\boldsymbol{R}_\pi^{n,(m)}\|_2/\|\boldsymbol{R}_\pi^{n,(0)}\|_2,\ \|\boldsymbol{R}_p^{n,(m)}\|_2/\|\boldsymbol{R}_p^{n,(0)}\|_2,\right.$$
$$\left.\|\boldsymbol{R}_s^{n,(m)}\|_2/\|\boldsymbol{R}_s^{n,(0)}\|_2\right\} < \tau_{nl,r}.$$

For all tests, we set $\tau_{nl,a} = \tau_{nl,r}$ taking the value of 1.E-6 for Test 1 and 1.E-5 otherwise.

In order to provide the most comprehensive picture of the solver's computational performance, we identified a set of monitoring parameters defined as follows:

1. The number of nonlinear and linear iterations per time step, N_N and N_l, respectively;
2. The total simulation time in seconds per time step, t_t, further decomposed into t_p and t_s, i.e., the time to build the preconditioner and for GMRES to iterate to convergence, respectively;
3. The ratio R_S of the number of non-zeros in the Schur complement (Eq. 30) computed with the selected pattern for $\widetilde{F}$ with respect to the sparsest one, which corresponds to the nonzero patch of $J_{\pi p}$ (also denoted as original in the sequel), i.e., $\mathtt{nnz}(\widetilde{S})/\mathtt{nnz}(\widetilde{S}_{\text{orig}})$.

While the indicators defined in points 1. and 2. concern the computational efficiency, R_S in point 3. provides an indication as to the memory footprint of the preconditioner. With the aim at measuring both the local and global solver performance, we will report the aforementioned parameters marked by the symbols $\widehat{(\cdot)}$ and $\overline{(\cdot)}$, which refer to the cumulative per-

Table 1 Setup of the test cases

Test		1	2a	2b	3
Reservoir type		Plain	Plain	Plain	Dome
Perm. tensor prop.		Homogeneous	Heterogeneous	Heterogeneous	Heterogeneous
		Isotropic	Anisotropic	Anisotropic	Anisotropic
Horiz. perm. range	$\left[\mathrm{m}^2\right]$	1.E-12	[3.0E-18, 2.0E-11]	[3.0E-18, 2.0E-11]	[3.0E-18, 2.0E-11]
Vert. perm. range	$\left[\mathrm{m}^2\right]$	1.E-12	[3.8E-22, 6.0E-12]	[3.8E-22, 6.0E-12]	[1.0E-13, 6.0E-12]
Porosity		0.25	[2.6E-05, 5.0E-01]	[2.6E-05, 5.0E-01]	[2.6E-05, 5.0E-01]
Oil spec. gravity	$\left[\frac{\mathrm{kPa}}{\mathrm{m}}\right]$	–	–	8.00	–
Water spec. gravity	$\left[\frac{\mathrm{kPa}}{\mathrm{m}}\right]$	–	–	9.81	–

The petrophysical properties of the medium relevant to the two-phase flow model in a compressible porous matrix, common to all the tests, are as follows: rock compressibility c_r = 5.E-7 kPa^{-1}, oil dynamic viscosity μ_o = 2.3148E-11 kPa d, water dynamic viscosity μ_w = 1.1574E-11 kPa d, irreducible water saturation $S_{wr} = 0$, residual oil saturation $S_{or} = 0$, and initial water saturation $S_w^0 = 0$. The relative permeability profiles used in the tests are analytically expressed through Corey's model and exhibit a parabolic shape

formance throughout the whole simulation and the average performance in a single system, respectively. In particular, the latter parameter is computed by looking only at the first linearized systems per time step (denoted by the additional subscript 1).

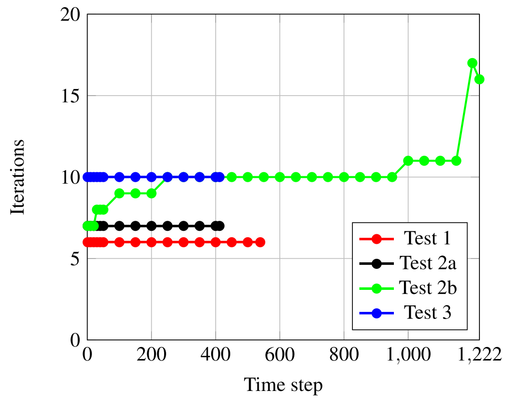
The tests are carried out on a workstation equipped with an AMD Ryzen 9 3950X 16-Core processor at 3.49 GHz and 64 GB of RAM. Due to the prototypical Matlab implementation of the code, the results should be regarded as preliminary for the CPU time. An improved absolute CPU time performance is expected with compiled programming languages, such as C++.

The applicability and effectiveness of the AMG tool to reproduce the action of the inverse of $J_{\pi\pi}$ and $\widetilde{S}$ is a crucial factor. To this aim, we investigate in advance how well the linear solver performs when dealing with the system of equations associated with $J_{\pi\pi}$ and $\widetilde{S}$, for some RHS, using the chosen AMG function as a preconditioner. The number of iterations to converge at a given tolerance (equal to 1.E-8 in this analysis) is used to measure the quality of the AMG preconditioner for these local problems. This analysis is carried out at different time steps to evaluate the possible dependency on the problem evolution. Figure 5 summarizes the outcome of this preliminary analysis, where AGMG is used as a preconditioner for the GCR solver [50, 51]. For the sake of consistency, GCR is used in place of CG even for Symmetric Positive Definite (SPD) matrices, i.e., $-J_{\pi\pi}$. Block $J_{\pi\pi} \in \mathbb{R}^{n_f \times n_f}$ arises from an elliptic contribution, so it is well-suited to AMG, and the solver converges in fewer iterations than $\widetilde{S} \in \mathbb{R}^{(n_E+n_w)\times(n_E+n_w)}$. When gravity is not considered in the model (Tests 1, 2a, and 3), $J_{\pi\pi}$ does not depend on mobility, it is constant during the simulation and is also SND. Otherwise, it is weakly non-symmetric and evolves over time (Test 2b), and such a scenario is computationally more challenging for AMG. As the simulation proceeds, the number of iterations slightly increases, nonetheless AGMG appears to behave satisfactorily well. As to Fig. 5b, we can say that AGMG serves quite satisfactorily as a preconditioner for the Schur complement as well. In particular, the combination of a non-Cartesian grid and heterogeneous anisotropic permeability produces the most challenging scenario. Ultimately, we also observe that the AMG approximation is almost insensitive to the time step size both for $J_{\pi\pi}$ and $\widetilde{S}$. For all the tests, in fact, Δt grows, reaching $\Delta t_{\max}$ within the first 40/50 time steps, and, in this range, the iteration count remains almost constant.

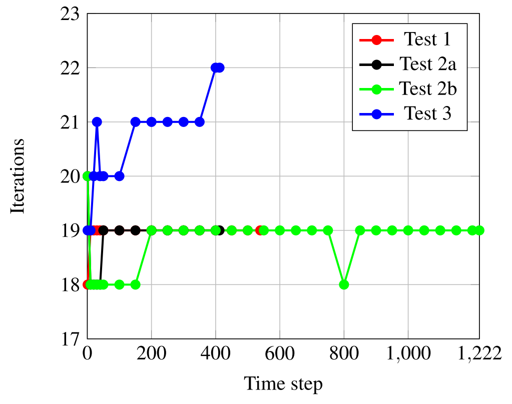
While the AMG setup for $\widetilde{S}$ needs to be performed at each nonlinear step, this task can be carried out only once at the outset of the simulation for $J_{\pi\pi}$ when gravity is neglected, since this block is constant during the simulation. On the contrary, when gravity is considered in the model, the AMG setup is required at each nonlinear iteration for both $J_{\pi\pi}$ and $\widetilde{S}$.

4.1 Test 1: Planar reservoir with homogeneous and isotropic permeability

Test 1 reproduces a textbook waterflooding application with production/injection lasting for 5,000 days ($\approx$ 13.7 years), which are covered in 539 time steps with a maximum time increment, $\Delta t_{\max}$, equal to 10 days. The Courant-Friedrichs-Lewy (CFL) number [57] is up to 6.25. Figure 6 offers some model insights in terms of water saturation and water velocity at the end of the simulation. In this application, but also in Tests 2a and 2b, we study the performance of the BCPR preconditioner when the Schur complement is built using the static approach only. In our previous study [32], in fact, the more expensive dynamic variant proved not necessary when the flow field is moderately regular and the mesh is Cartesian. Table 2 summarizes the main outcome of seven runs performed with the patterns shown in Fig. 3. Not surprisingly, as the pattern is enlarged (runs from 1 to 7), the cumulative

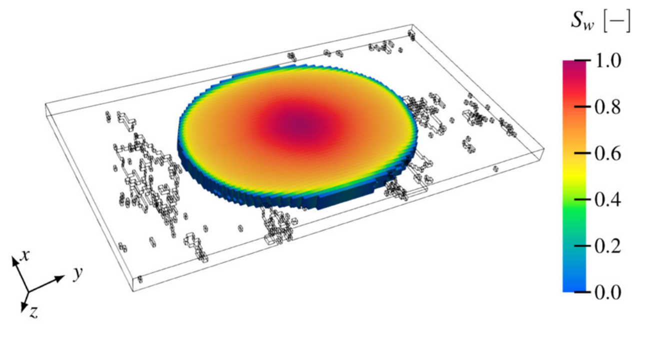


(a) Systems with $J_{\pi\pi}$ as matrix.

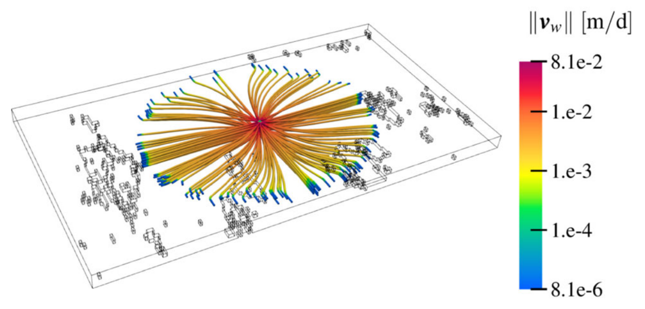


(b) Systems with $\widetilde{S}$ as matrix.

Fig. 5 Number of iterations to converge for the systems of equations associated with blocks $J_{\pi\pi}$, (a), and $\widetilde{S}$, (b), generated during the simulations in Tests 1-3. The systems are solved, to a tolerance equal to 1.E-8, by means of GCR preconditioned with AGMG. The Schur complement $\widetilde{S}$ is built using pattern A (Fig. 3b)



(a) Water saturation



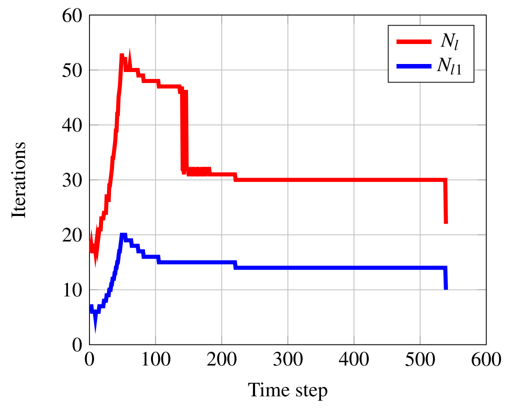
(b) Water velocity

Fig. 6 Test 1: Some model insights at the end of the simulation ($t = 5,000$ d)

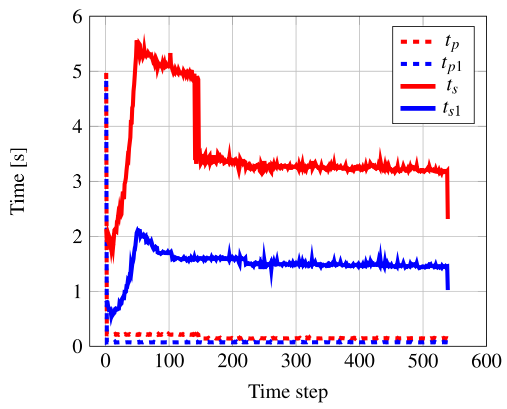
time to build the preconditioner, $\hat{t}_p$, increases, whereas the number of linear iterations, $\hat{N}_l$, decreases. The most significant reduction, around 20%, is from run 1 to 2, i.e., moving from $\widetilde{F}$ built with the original pattern to pattern A. Further patch enlargements give negligible improvements. Actually, the density of the Schur complement (see the R_S column) resulting from the pattern expansion increases the computation and application cost of the preconditioner, so that the best performance is achieved with patterns A-D. This result shows that there is no need for a significant filling of $\widetilde{F}$, with one or two additional levels of the connection, with respect to the original pattern, already enough. Focusing on every single system, convergence is achieved on average in less than 15 iterations. Moreover, we observe that patterns B and D deliver the same results as their counterparts A and C, meaning that they produce the same $\widetilde{F}$ factor. This is due to the block diagonal structure of the B^E matrices (Eq. 9) for Cartesian discretizations. Therefore, in that context, the approximation in computing $\widetilde{F}$ is only controlled by the level of the connection and not by the presence of the lateral faces.

The overall performance of the linear solver during the whole simulation is shown in Fig. 7 in terms of linear iterations and CPU time per time step. The number of nonlinear iterations is 3 until the 140th time step and 2 later on, thus explaining the sudden jump observed in the red profiles. The linear solver displays good performance throughout the full simulation, stabilizing after $\Delta t_{\max}$ is reached, with no impact on the nonlinear convergence, as confirmed in column $\hat{N}_N$ in Table 2. Figure 7b also shows, from the peak in the t_p and t_{p1} profiles, that the impact of the computation of $\widetilde{F}$ is at the outset of the simulation. However, such a cost can be effectively amortized during a full run.

The motivation behind the introduction of the BCPR algorithm is the lack of efficiency of conventional general-purpose AMG tools to approximate the whole 2×2 pressure part of the system in Eq. 18. In order to investigate this problem, the system with J_{PP} is solved using the GCR method preconditioned with AGMG for some RHS. The convergence profile is shown in Fig. 8 along with those of the diagonal blocks, $J_{\pi\pi}$ and J_{pp}, and Schur complement $\widetilde{S}$ for comparison. While convergence is fast for $J_{\pi\pi}$, J_{pp}, and $\widetilde{S}$, GCR fails to converge within 100 iterations for the system with J_{PP}. Therefore, extending the classical CPR method to our model problem (Eq. 18) is expected to deliver poor results.



(a) Linear iterations vs. time step



(b) CPU time vs. time step

Fig. 7 Test 1: Number of linear iterations (a) and CPU time (b) vs. the time step index during the full run 2 of Table 2

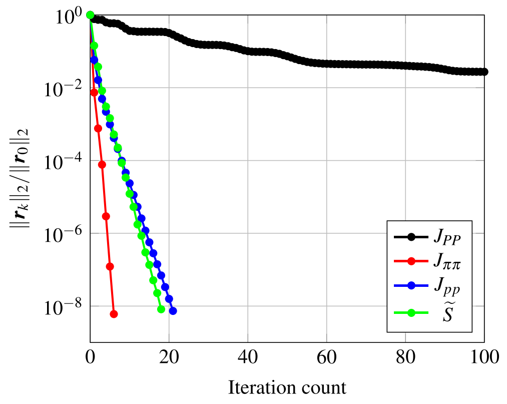


Fig. 8 Test 1: Convergence profiles of GCR preconditioned with AGMG for the global pressure problem, J_{PP}, leading terms, $J_{\pi\pi}$ and J_{pp}, and approximate Schur complement $\widetilde{S}$. The exit tolerance is 1.E-8

Substituting the ILU(0) factorization with the less expensive Jacobi preconditioner at the global stage is the second main modification, in our BCPR preconditioner, to the original CPR algorithm, since we observed a substantial degradation in the solver convergence as the size of the model is enlarged. The numerical results in Fig. 9 help us to support this choice. In this analysis, we solve the system with the Jacobian matrix $\mathcal{J}$ using GMRES preconditioned with some variants of incomplete factorization and the Jacobi preconditioner. The exit tolerance and the maximum number of iterations are set equal to 1.E-8 and 100, respectively. We do not expect GMRES to converge due to the complex block structure of the problem, rather we are interested in the solver behavior during the very first iterations since the global-stage preconditioner is applied only once in the BCPR algorithm. Computing a factorization with zero fill-in directly on the Jacobian matrix ordered as in Fig. 1b produces the worst results in terms of initial convergence rate. Applying some algebraic reordering techniques, such as Minimum degree ordering (MinDeg), Nested dissection (Dissect), and Reverse Cuthill-McKee (RCM) [58] may help to improve the initial solver behavior. A threshold-based variant of ILU, computed on the RCM-reordered Jacobian matrix, does not allow for significantly better performance. On the contrary, as the threshold τ is reduced, convergence can degrade, although the factors become very dense. On the other hand, the convergence profile with the Jacobi preconditioner (the cyan line) is overall comparable to the best outcome of the reordered ILU factorizations, while being computationally much cheaper and embarrassingly parallel.

Table 2 Test 1: Numerical performance of the static technique

#	Pat	R_S	$\hat{N}_N$	$\hat{N}_l$	$\hat{t}_p$ [s]	$\hat{t}_s$ [s]	$\hat{t}_t$ [s]	$\overline{N}_{l1}$	$\bar{t}_{p1}$ [s]	$\bar{t}_{s1}$ [s]	$\bar{t}_{t1}$ [s]
1	Orig	1	1,221	22,452	68.7	2,556.7	2,625.4	17.7	0.1	2.0	2.1
2	A	1.4	1,221	17,892	92.2	1,917.7	2,009.9	14.2	0.1	1.5	1.6
3	B	1.4	1,221	17,892	91.4	1,911.6	2,003.0	14.2	0.1	1.5	1.6
4	C	1.7	1,221	17,396	104.4	1,909.2	2,013.6	13.7	0.1	1.5	1.6
5	D	1.7	1,221	17,396	105.8	1,910.8	2,016.6	13.7	0.1	1.5	1.6
6	E	2.0	1,221	17,341	141.7	2,163.1	2,304.8	13.6	0.1	1.7	1.8
7	F	2.4	1,221	17,326	154.5	2,244.7	2,399.2	13.6	0.1	1.7	1.8

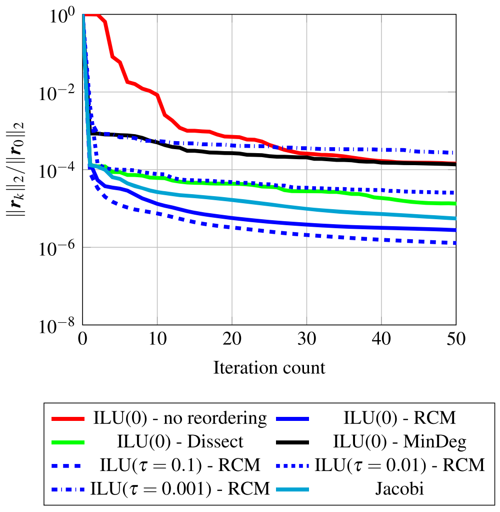


Fig. 9 Test 1: Convergence profiles of GMRES preconditioned with Jacobi preconditioner and some variants of ILU factorizations for the Jacobian matrix $\mathcal{J}$. The exit tolerance is 1.E-8

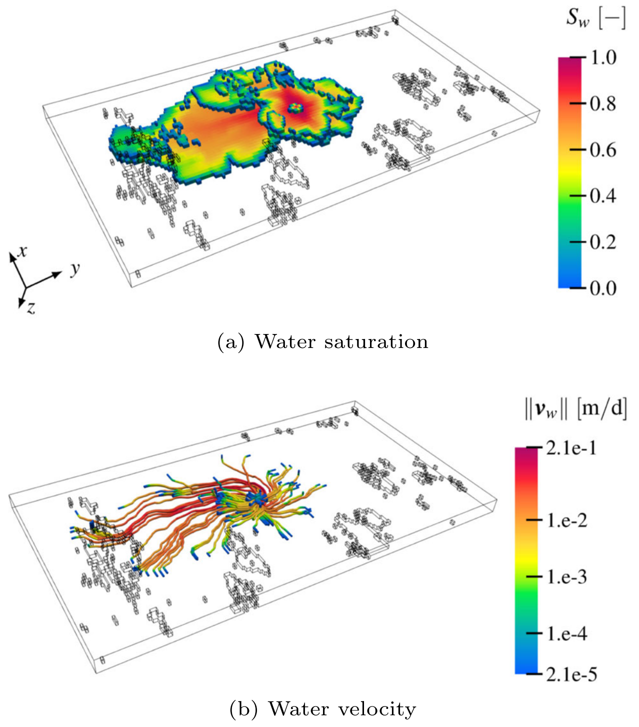


Fig. 10 Test 2a: Some model insights at the end of the simulation ($t = 1,500$ d)

4.2 Tests 2a: Planar reservoir with heterogeneous and anisotropic permeability

In this test, we consider a challenging scenario characterized by the permeability and porosity distributions of the SPE10 data set. The simulated time is 1,500 days ($\approx$ 4.1 years) of continuous production/injection spanned in 412 time steps. The maximum time increment is set equal to 4 days, and the CFL number reaches a value of 19.9.

Figure 10 illustrates the physical outcome of the model at the end of the simulation. The analysis of the BCPR performance with the five patterns without lateral faces is provided in Table 3, where we see that a failure is reported when the Schur complement is built with the original patch. This pattern, in fact, produces a rough approximation of $\widetilde{S}$, which soon loses the algebraic properties that make it suitable for AGMG. Such an issue is easily solved by expanding the element-to-face connection and, once again, pattern A (run 2) delivers the best results in terms of CPU time. The performance of the nonlinear and linear solvers with pattern A for the EDFA method is displayed in Fig. 11, where we can observe a similar behavior as in Test 1.

The computational challenge offered by the SPE10 data set allows us to benchmark, in a realistic setting, our preconditioning solution against a standard approach to approximate the Schur complement. Performing this task in an effective but inexpensive way is typically one of the most challenging steps in the design of a block preconditioner. Before resorting to more advanced approaches, Jacobi preconditioner for the inverse of the block in position (1,1) is definitely one of the first attempts, yielding the approximation $\widetilde{S} = J_{pp} - J_{p\pi}\text{diag}(J_{\pi\pi})^{-1}J_{\pi p}$. We tested this option in the BCPR framework and compared the solver performance, in terms of linear iterations and total CPU time, with that obtained with the best EDFA setting (run 2 in Table 3). The EDFA method clearly outperforms the Jacobi-based approach, as shown in Fig. 12, which produces a too-coarse approximation of the second part of the Schur complement.

4.3 Tests 2b: Planar reservoir with heterogeneous and anisotropic permeability and gravity

This application is the evolution of Test 2a with the introduction of gravity in the model to evaluate its effect on the performance of the BCPR preconditioner. The maximum time step size, equal to 1 day, is smaller than in the previous tests and has been chosen to limit the number of nonlinear iterations to 5 or 6 at most. This results in a larger number of time steps, up to 1,222, although the simulated time interval has been reduced to 1,200 days ($\approx$ 3.3 years). The CFL number reaches a value of 24.4 during the simulation. The final outcome from the model is depicted in Fig. 13.

Table 4 reports the performance of the BCPR preconditioner with the static approach for building $\widetilde{F}$. As compared to Test 2a, the introduction of gravity does not seem to affect the setup strategy of the preconditioner. Pattern A enables a significant decrease (17.63%) in the number of linear iterations with respect to the original patch (which allows for convergence in this test) and is again the optimal choice. Further expansions produce a larger cost in terms of CPU time,

Table 3 Test 2a: Numerical performance of the static technique

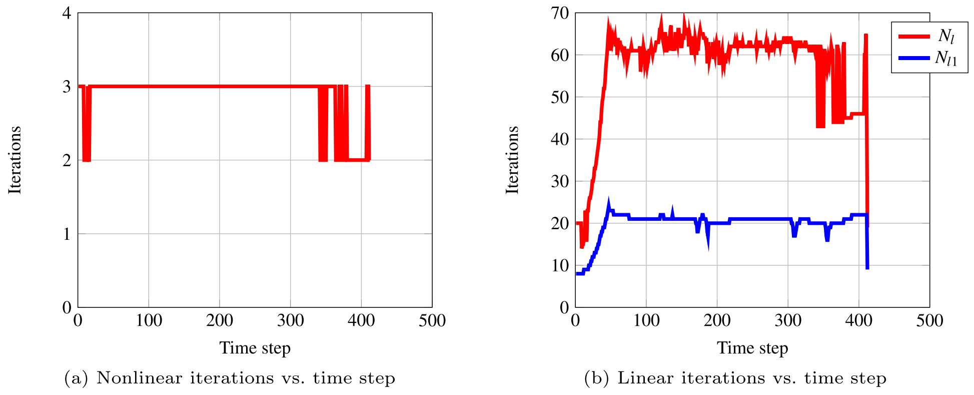
#	Pat	R_S	$\hat{N}_N$	$\hat{N}_l$	$\hat{t}_p$ [s]	$\hat{t}_s$ [s]	$\hat{t}_t$ [s]	$\overline{N}_{l1}$	$\overline{t}_{p1}$ [s]	$\overline{t}_{s1}$ [s]	$\overline{t}_{t1}$ [s]
1	Orig	1	NC	–	–	–	–	–	–	–	–
2	A	1.4	1,184	23,402	94.2	2,070.9	2,165.1	19.9	0.1	1.7	1.8
3	C	1.7	1,183	22,488	112.6	2,209.2	2,321.8	19.3	0.1	1.9	2.0
4	E	2.0	1,221	23,037	138.2	2,352.2	2,490.4	19.3	0.1	2.0	2.1
5	F	2.4	1,212	22,810	167.1	2,514.8	2,681.9	19.3	0.1	2.1	2.2

which, nevertheless, remains close to 1 s for the solution of a single system after approximately 10 iterations. Although the presence of gravity does not appear to significantly influence the setup and convergence of the linear solver, it does affect that of the nonlinear solver, as shown in Fig. 14. The number of nonlinear iterations, in fact, is larger than in Test 2a, though the maximum time step size is smaller. The other panels confirm that the performance of the BCPR preconditioner is overall stable during the simulation, being the average number of linear iterations per nonlinear step almost constant after the maximum time step size is reached.

4.4 Test 3: Dome reservoir with heterogeneous and anisotropic permeability

In this application, the simulated temporal domain spans 1,500 days (≈ 4.1 years) of continuous production/injection, subdivided in 412 time steps with $\Delta t_{\max} = 4$ days and a maximum CFL number equal to 13.2. An overview of the model outcome at the end of the simulation is provided in Fig. 15, while Tables 5 and 6 report the results of the numerical investigation, carried out using both the static and dynamic variants for the Schur complement approximation, respectively.

Using a non-Cartesian grid with heterogeneity and anisotropy makes the linearized problem computationally more challenging, as first observed in the comment about Fig. 5b and confirmed by the increase in the number of linear iterations in $\overline{N}_{l1}$ (compare Tables 5 and 6 to 2 and 3). The computation and application cost of the preconditioner is also larger than in the previous tests. In fact, blocks $J_{\pi\pi}$ and $\widetilde{S}$ with the original pattern are 3.6 and 1.9 times denser, respectively. This is due to the fact that, with a non-Cartesian grid, the elementary matrix B^E (see Eq. 9) is generally full.



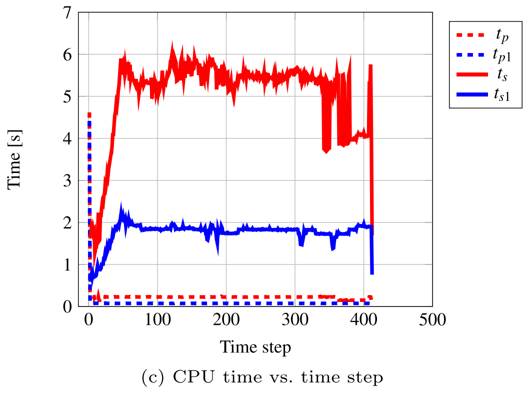
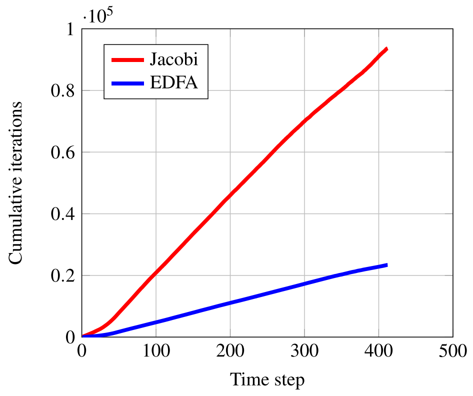
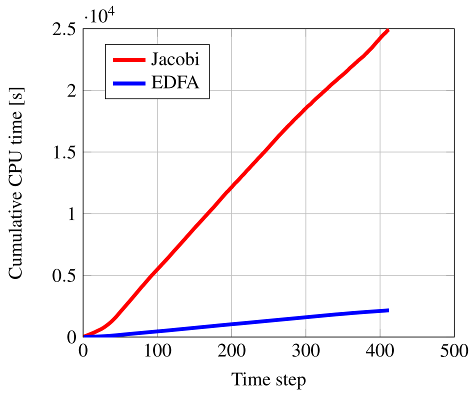


Fig. 11 Test 2a: Number of nonlinear (a) and linear (b) iterations and CPU time (c) vs. the time step index during the full run 2 of Table 3

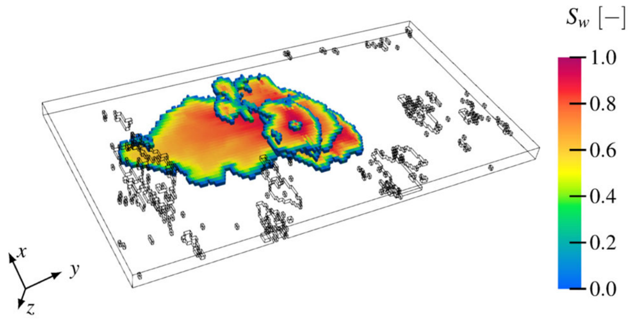


(a) Cumulative linear iterations during the simulation

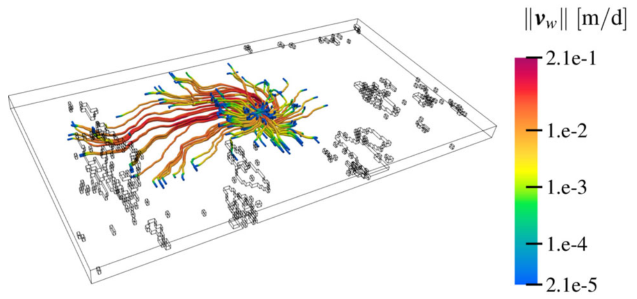


(b) Cumulative total CPU time during the simulation

Fig. 12 Test 2a: Comparing the performance of different approximations of the Schur complement, using Jacobi preconditioner for $J_{\pi\pi}$ and the best setting for the EDFA method (run 2 in Table 3)



(a) Water saturation



(b) Water velocity

Fig. 13 Test 2b: Some model insights at the end of the simulation ($t = 1,200$ d)

This has also an effect on $\widetilde{F}$, for which including the lateral faces in the element-to-face pattern produces now a different approximation (compare runs 2,3, and 4,5 in Table 5).

An appreciable reduction in the total number of linear iterations occurs in run 2 (pattern A), as in the previous tests, however, the total CPU time does not decrease correspondingly because of the R_S growth. Further pattern expansions turn out to be actually detrimental to the solver speed. Hence, the optimal setup makes use of either the original or the A pattern, with up to 6 additional non-zeros introduced in each column of $\widetilde{F}$.

Based on these results, we might argue that the optimal performance of the dynamic technique is achieved with a similar number of new column entries, n_{ent}. This is confirmed by a two-step sensitivity analysis in which we tuned n_{ent} to find its optimal value, before adjusting n_{add}, i.e., the number of entries added at each iteration. Table 6 summarizes the outcome of this investigation. In the first six runs, we fix $n_{\text{add}} = 2$. The total number of linear iterations decreases as n_{ent} grows up to 6, then the performance deteriorates; therefore, this is the optimal value sought. In the second part of the analysis, where n_{add} is tuned (runs 7-11), we observe that, as this parameter grows, $\hat{t}_p$ decreases since fewer iterations are required to compute each column of $\widetilde{F}$. The best results are given for $n_{\text{add}} = 3$ (run 8), then the number of linear iterations starts to increase slowly, reaching the maximum when $n_{\text{add}} = 6$, i.e., all the new column entries are introduced at the first iteration. This result is consistent with the sensitivity analysis performed in [32] on the same non-Cartesian discretization, where we observed that at least two iterations are needed for the decoupling factor approximation to be effective. Overall, the solver performance in run 8 improves the one recorded in Table 5 both in terms of iteration count and total CPU time.

5 Discussion and conclusions

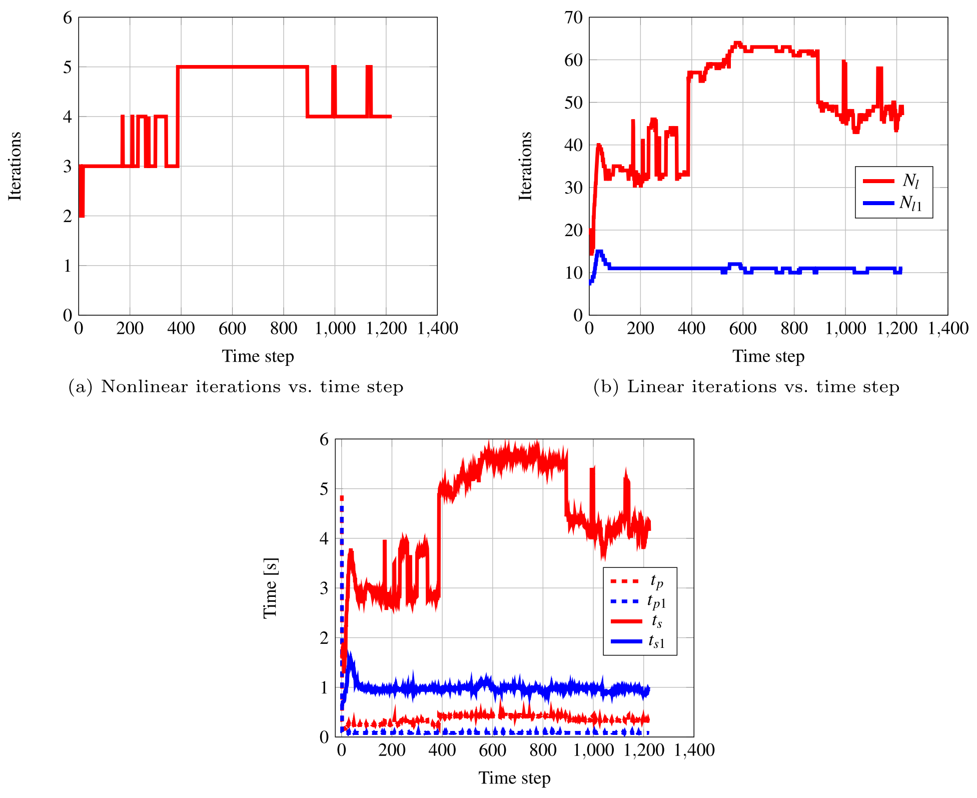
The efficient solution of the systems of linearized equations (Eq. 18), originating from an original MHFE-FV discretization of the PDEs that govern the classical two-phase flow model in compressible media, was the main objective of this work. To this end, we designed a preconditioning algorithm built on top of the classical CPR approach and adapted to the 3×3 block structure of the Jacobian matrix (Eq. 18)

Table 4 Test 2b: Numerical performance of the static technique

#	Pat	R_S	$\hat{N}_N$	$\hat{N}_l$	$\hat{t}_p$ [s]	$\hat{t}_s$ [s]	$\hat{t}_t$ [s]	$\overline{N}_{l1}$	$\bar{t}_{p1}$ [s]	$\bar{t}_{s1}$ [s]	$\bar{t}_{t1}$ [s]
1	Orig	1	5,105	73,148	341.0	7,064.5	7,405.5	13.5	0.1	1.4	1.5
2	A	1.4	5,099	60,249	433.9	5,338.3	5,772.2	10.9	0.1	1.0	1.1
3	C	1.7	5,099	56,953	504.9	5,665.6	6,170.5	10.4	0.1	1.0	1.1
4	E	2.0	5,099	57,413	612.0	5,823.7	6,435.7	10.5	0.1	1.1	1.2
5	F	2.4	5,100	57,276	759.6	5,446.0	6,205.6	10.4	0.1	1.0	1.1

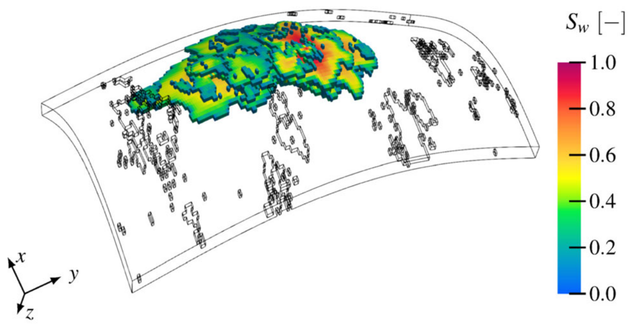
by introducing a block preconditioner for the 2 × 2 pressure subproblem J_{PP}. This allows for achieving higher solver robustness by exploiting the inner block structure instead of using a single AMG for the whole J_{PP} part. The resulting preconditioner succeeds in incorporating block preconditioning within a global CPR-like algorithm and is thus labeled Block CPR (BCPR). The new core of this tool is the block preconditioner for J_{PP}, whose development builds on our previous works [31, 32] about the Explicit Decoupling Factor Approximation (EDFA) preconditioner. This technique relies on the approximation of the Schur complement with inexact versions of the Jacobian decoupling factors, making use of appropriate restriction/prolongation operators built upon non-zero patterns created either statically or dynamically, as also proposed in [59, 60]. Two additional modifications to the native CPR algorithm have been introduced. Due to the poor performance of a global ILU preconditioner and in view of a massive parallelization of the algorithm, the Jacobi preconditioner has been used at the global stage. The second modification concerns the order of the stages, which has been inverted, following the approaches in [22, 46]. The aim is to introduce some kind of decoupling effect of pressure from saturation, since this task is not explicitly performed in a preprocessing step to preserve the original algebraic properties of J_{PP}. In essence, we inverted the classical CPR algorithmic structure and modified both preconditioners for the stages. The resulting BCPR algorithm can be cast in an AMG-like framework, where the global stage corresponds to a Jacobi pre-smoothing step, followed by a coarse-level solver on pressures carried out by the EDFA tool.

The testing phase in Section 4 showed that the BCPR preconditioner is overall robust and efficient. Its setup consists mainly of tuning the parameters controlling the behavior of the inner EDFA tool, which represents the original core of the

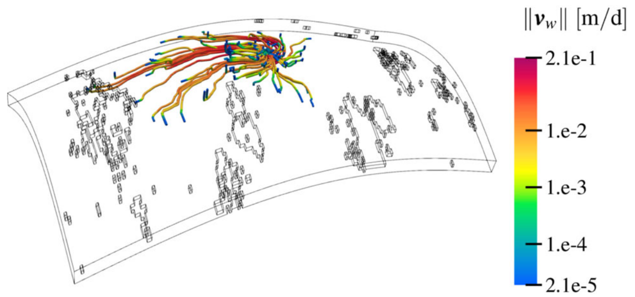


(a) Nonlinear iterations vs. time step
(b) Linear iterations vs. time step
(c) CPU time vs. time step

Fig. 14 Test 2b: Number of nonlinear (a) and linear (b) iterations and CPU time (c) vs. the time step index during the full run 2 of Table 4



(a) Water saturation



(b) Water velocity

Fig. 15 Test 3: Some model insights at the end of the simulation (t = 1, 500 d)

BCPR technique, although other elements can be modified such as the AMG function or the global-stage preconditioner. The numerical experiments on different test cases showed that only a few additional entries need to be included in the original column sparsity pattern (Fig. 3a) to obtain an accurate and sparse, thus efficient, approximation of F, hence of the Schur complement itself. These results are consistent with the previous outcome obtained for the EDFA preconditioner alone in a single-phase problem [31, 32]. Specifically, with a Cartesian grid and regardless of the rock property distribution (Test 1-2), the static technique with pattern A is preferable. The introduction of gravity into the flow model (Test 2b), while causing the nonlinear problem to be computationally more challenging, does not significantly affect the overall behavior and setup of the linear solver. Moving from a Cartesian to a non-Cartesian tessellation modifies the stencil of some blocks in the Jacobian matrix, in particular, $J_{\pi\pi}$ and $J_{p\pi}$, and results in a denser approximation of the Schur complement, while using the same pattern, and in higher computation and application cost of the BCPR preconditioner. In this setting, expanding the original pattern with the dynamic technique proved to be more efficient than the static approach, but, once more, only a few additional column entries in $\widetilde{F}$ are required for optimal performance of the BCPR preconditioner.

The BCPR preconditioner has been designed following an algebraic approach in order to develop a general framework that can be adapted to other multiphysics problems, including, for instance, thermodynamic equilibrium, and discretization schemes with hybrid variables, like the Mimetic finite difference method [27]. This work is thus the basis for future developments, mainly following two principal directions: expanding the underlying physical model and strengthening the implementation. We initially took the two-phase flow problem as a reference model, with incompressible fluids and no capillarity effects. However, fluid compressibility and capillarity can be introduced in the flow model as a first step towards an extension of the BCPR framework to multiphase black-oil models and, in a farther future, compositional simulations. Yet, adding capillarity would require some work as the structure of the Jacobian can change (see also [26]). The good performance of the proposed algorithm also needs to be confirmed for those more sophisticated flow models. On the other hand, we want to develop a more efficient version of the BCPR preconditioner, derived from the Matlab prototype, using compiled programming languages, such as C++, with the ultimate goal of including it in our in-house built QASR simulator [28]. This would also allow linking mainstream AMG libraries to the code and carrying out more significant analyses on the CPU time con-

Table 5 Test 3: Numerical performance of the static technique

#	Pat	R_S	$\hat{N}_N$	$\hat{N}_l$	$\hat{t}_p$ [s]	$\hat{t}_s$ [s]	$\hat{t}_t$ [s]	$\overline{N}_{l1}$	$\overline{t}_{p1}$ [s]	$\overline{t}_{s1}$ [s]	$\overline{t}_{t1}$ [s]
1	Orig	1	1,235	27,915	125.4	4,924.7	5,050.1	24.4	0.1	4.3	4.4
2	A	2.0	1,235	23,879	233.6	4,797.0	5,030.6	20.8	0.2	4.2	4.4
3	B	2.3	1,238	28,903	266.2	7,714.8	7,981.0	25.2	0.2	6.8	7.0
4	C	2.9	1,235	23,705	343.2	5,748.9	6,092.1	20.6	0.3	5.0	5.3
5	D	4.3	1,224	23,268	504.7	7,671.6	8,176.3	19.3	0.4	6.4	6.8
6	E	3.7	1,235	23,722	456.7	7,803.3	8,260.0	20.5	0.4	6.8	7.2
7	F	4.5	1,235	23,673	558.4	8,767.7	9,326.1	20.5	0.5	7.6	8.1

Table 6 Test 3: Numerical performance of the dynamic technique

#	n_{ent}	n_{add}	R_S	$\hat{N}_N$	$\hat{N}_l$	$\hat{t}_p$ [s]	$\hat{t}_s$ [s]	$\hat{t}_t$ [s]	$\overline{N}_{l1}$	$\bar{t}_{p1}$ [s]	$\bar{t}_{s1}$ [s]	$\bar{t}_{t1}$ [s]
1	2	2	1.4	1,235	25,420	176.7	5,191.4	5,368.1	22.1	0.1	4.6	4.7
2	4	2	1.7	1,235	24,429	233.0	5,308.3	5,541.3	21.1	0.2	4.6	4.8
3	6	2	2.0	1,235	23,151	278.4	4,409.9	4,688.3	20.2	0.2	3.9	4.1
4	8	2	2.2	1,235	23,881	319.7	6,043.2	6,362.9	20.3	0.2	5.2	5.4
5	10	2	2.5	1,235	24,682	356.8	6,869.6	7,226.4	21.1	0.3	5.9	6.2
6	12	2	2.7	1,235	25,878	390.1	7,577.3	7,967.4	21.9	0.3	6.5	6.8
7	6	1	2.0	1,235	23,299	301.1	4,493.2	4,794.3	20.3	0.2	3.9	4.1
8	6	3	2.0	1,235	22,904	272.7	4,356.7	4,629.4	20.1	0.2	3.9	4.1
9	6	4	2.0	1,235	23,179	275.2	4,471.2	4,746.4	20.3	0.2	4.0	4.2
10	6	5	2.0	1,235	23,910	271.6	4,683.7	4,955.3	20.5	0.2	4.0	4.2
11	6	6	2.0	1,235	24,404	252.1	4,903.1	5,155.2	21.1	0.2	4.3	4.5

sumption, especially on larger models. Moreover, ongoing work is being conducted to better understand the benefits of inverting the order of the stages in the BCPR algorithm for its overall effectiveness, by performing a theoretical error analysis accompanied by a thorough numerical investigation.

Funding Open Access funding provided by the Qatar National Library. This publication was supported by the National Priorities Research Program grant NPRP11S-1210-170079 from Qatar National Research Fund.

Declarations

Competing interests The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Magras, J.F., Quandalle, P., Bia, P.: High-performance reservoir simulation with parallel ATHOS. In: SPE Reserv. Simul. Symp. Houston, Texas, USA: Society of Petroleum Engineers, pp. SPE–66342–MS. (2001). Available from: https://onepetro.org/spersc/proceedings/01RSS/All-01RSS/Houston,Texas/133525
2. Hu, X., Wu, S., Wu, X.H., Xu, J., Zhang, C.S., Zhang, S., et al.: Combined preconditioning with applications in reservoir simulation. Multiscale. Model. Simul. **11**(2), 507–521 (2013). https://doi.org/10.1137/120885188
3. Esler, K., Gandham, R., Patacchini, L., Garipov, T., Samardzic, A., Panfili, P., et al.: A graphics processing unit–based, industrial grade compositional reservoir simulator. SPE J, pp. SPE–203929–PA. (2021). https://doi.org/10.2118/203929-PA
4. Wallis, JR.: Incomplete Gaussian elimination as a preconditioning for generalized conjugate gradient acceleration. In: SPE Reserv. Simul. Symp. San Francisco, California: Society of Petroleum Engineers, pp. 325–334. (1983). Available from: https://doi.org/10.2118/12265-MS
5. Wallis, J.R., Kendall, R.P., Little, T.E.: Constrained residual acceleration of conjugate residual methods. In: SPE Reserv. Simul. Symp. Dallas, Texas: Society of Petroleum Engineers, pp. SPE–13536–MS. (1985). Available from: https://doi.org/10.2118/13536-MS
6. Zhou, Y., Jiang, Y., Tchelepi, H.A.: A scalable multistage linear solver for reservoir models with multisegment wells. Comput. Geosci. **17**(2), 197–216 (2013). https://doi.org/10.1007/s10596-012-9324-0
7. Garipov, T.T., Tomin, P., Rin, R., Voskov, D.V., Tchelepi, H.A.: Unified thermo-compositional-mechanical framework for reservoir simulation. Comput. Geosci. **22**, 1039–1057 (2018). https://doi.org/10.1007/s10596-018-9737-5
8. Cremon, M.A., Castelletto, N., White, J.A.: Multi-stage preconditioners for thermal-compositional-reactive flow in porous media. J. Comput. Phys. **418**, 109607 (2020). https://doi.org/10.1016/j.jcp.2020.109607
9. Klemetsdal, Ø.S., Møyner, O., Lie, K.A.: Accelerating multiscale simulation of complex geomodels by use of dynamically adapted basis functions. Comput. Geosci. **24**(2), 459–476 (2020). https://doi.org/10.1007/s10596-019-9827-z
10. Lie, K.A.: An introduction to reservoir simulation using MATLAB/GNU Octave. Cambridge, United Kingdom: Cambridge University Press (2019). Available from: https://www.cambridge.org/core/product/identifier/9781108591416/type/book
11. Alvestad, J., Baxendale, D., Bao, K., Blatt, M., Hove, J., Lauser, A., et al.: OPM flow: Reference manual. Oslo, Norway: Equinor ASA (2022). Available from: https://opm-project.org/wp-content/uploads/2022/05/OPM_Flow_Reference_Manual_2022-04_Rev-0_Reduced.pdf
12. Rasmussen, A.F., Sandve, T.H., Bao, K., Lauser, A., Hove, J., Skaflestad, B., et al.: The open porous media flow reservoir simulator. Comput. Math. with Appl. **81**, 159–185 (2021). https://doi.org/10.1016/j.camwa.2020.05.014
13. Schlumberger: Eclipse: Technical description (2020)
14. Schlumberger: Intersect: Technical description (2020)
15. Halliburton: Nexus: Technical reference guide (2014)

16. Lacroix, S., Vassilevski, Y.V., Wheeler, M.F.: Decoupling preconditioners in the implicit parallel accurate reservoir simulator (IPARS). Numer. Linear. Algebr. with Appl. **8**(8), 537–549 (2001). https://doi.org/10.1002/nla.264
17. Singh, G., Pencheva, G., Wheeler, M.F.: An approximate Jacobian nonlinear solver for multiphase flow and transport. J. Comput. Phys. **375**, 337–351 (2018). https://doi.org/10.1016/j.jcp.2018.08.043
18. Lacroix, S., Vassilevski, Y., Wheeler, J., Wheeler, M.: 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
19. Cao, H., Tchelepi, H.A., Wallis, J.R., Yardumian, H.E.: Parallel scalable unstructured CPR-type linear solver for reservoir simulation. In: SPE Annu. Tech. Conf. Exhib. Dallas, Texas: Society of Petroleum Engineers, pp. SPE–96809–MS. (2005). Available from:https://doi.org/10.2118/96809-MS
20. 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
21. Nardean, S., Ferronato, M., Abushaikha, A.: Linear solvers for reservoir simulation problems: An overview and recent developments. Arch. Comput. Methods Eng. **29**(6), 4341–4378 (2022). https://doi.org/10.1007/s11831-022-09739-2
22. Roy, T., Jönsthövel, T.B., Lemon, C., Wathen, A.J.: A constrained pressure-temperature residual (CPTR) method for non-isothermal multiphase flow in porous media. SIAM J. Sci. Comput. **42**(4), B1014–B1040 (2020). https://doi.org/10.1137/19M1292023
23. White, J.A., Castelletto, N., Klevtsov, S., Bui, Q.M., Osei-Kuffuor, D., Tchelepi, H.A.: A two-stage preconditioner for multiphase poromechanics in reservoir simulation. Comput. Methods Appl. Mech. Eng. **357**, 112575 (2019). https://doi.org/10.1016/j.cma.2019.112575
24. T Camargo, J., White, J.A., Castelletto, N., Borja, R.I.: Preconditioners for multiphase poromechanics with strong capillarity. Int. J. Numer. Anal. Methods Geomech. **45**(9), 1141–1168 (2021). https://doi.org/10.1002/nag.3192
25. Brezzi, F., Fortin, M.: Mixed and hybrid finite element methods. Vol. 15 of Springer Series in Computational Mathematics. New York, NY: Springer-Verlag New York (1991). Available from: http://link.springer.com/10.1007/978-1-4612-3172-1
26. Abushaikha, A.S., Voskov, D.V., Tchelepi, H.A.: Fully implicit mixed-hybrid finite-element discretization for general purpose subsurface reservoir simulation. J. Comput. Phys. **346**, 514–538 (2017). https://doi.org/10.1016/j.jcp.2017.06.034
27. Abushaikha, A.S., Terekhov, K.M.: A fully implicit mimetic finite difference scheme for general purpose subsurface reservoir simulation with full tensor permeability. J. Comput. Phys. **406**, 109194 (2020). https://doi.org/10.1016/j.jcp.2019.109194
28. Li, L., Abushaikha, A.: A fully-implicit parallel framework for complex reservoir simulation with mimetic finite difference discretization and operator-based linearization. Comput. Geosci. (2021). https://doi.org/10.1007/s10596-021-10096-5
29. Kuznetsov, Y.A.: Spectrally equivalent preconditioners for mixed hybrid discretizations of diffusion equations on distorted meshes. J. Numer. Math. **11**(1), 61–74 (2003). https://doi.org/10.1163/156939503322004891
30. Maryska, J., Rozlozník, M., Tuma, M.: Schur complement systems in the mixed-hybrid finite element approximation of the potential fluid flow problem. SIAM J. Sci. Comput. **22**(2), 704–723 (2000). https://doi.org/10.1137/S1064827598339608
31. Nardean, S., Ferronato, M., Abushaikha, A.S.: A novel and efficient preconditioner for solving Lagrange multipliers-based discretization schemes for reservoir simulations. In: ECMOR XVII - 17th Eur. Conf. Math. Oil Recover. Edinburgh: European Association of Geoscientists & Engineers, pp. 1–12. (2020). Available from: https://www.earthdoc.org/content/papers/10.3997/2214-4609.202035072
32. Nardean, S., Ferronato, M., Abushaikha, A.S.: A novel block non-symmetric preconditioner for mixed-hybrid finite-element-based Darcy flow simulations. J. Comput. Phys. 110513 (2021). https://doi.org/10.1016/j.jcp.2021.110513
33. Nardean, S., Ferronato, M., Abushaikha, A.: A blended CPR/block preconditioning approach for mixed discretization schemes in reservoir modeling. In: ECMOR 2022 Eur. Conf. Math. Geol. Reserv. The Hague, The Netherlands: European Association of Geoscientists & Engineers, pp. 1–15. (2022). Available from: https://www.earthdoc.org/content/papers/10.3997/2214-4609.202244067
34. Coats, K.H.: An equation of state compositional model. SPE J. **20**(5), 363–376 (1980). https://doi.org/10.2118/8284-PA
35. Peaceman, D.W.: Interpretation of well-block pressures in numerical reservoir simulation. SPE J. **18**(3), SPE–6893–PA (1978). https://doi.org/10.2118/6893-PA
36. Chen, Z., Huan, G., Ma, Y.: Computational methods for multiphase flows in porous media. Society for Industrial and Applied Mathematics. Philadelphia, PA, USA (2006). Available from: https://doi.org/10.1137/1.9780898718942
37. Corey, A.T.: The interrelation between gas and oil relative permeabilities. Prod. Mon. **19**(1), (1954)
38. Brooks, R.H., Corey, A.T.: Hydraulic properties of porous media. Colorado State University, Fort Collins, Colorado, USA (1964)
39. Aziz, K., Settari, A.: Petroleum reservoir simulation. Applied Science Publishers, London, United Kingdom (1979)
40. Raviart, P.A., Thomas, J.M.: A mixed finite element method for 2-nd order elliptic problems. In: Galligani, I., Magenes, E. (eds.) Math. Asp. Finite Elem. Methods. Lect. Notes Math, pp. 292–315. Springer, Berlin, Heidelberg (1977). Available from: http://link.springer.com/10.1007/BFb0064470
41. Zhang, N., Abushaikha, A.S.: An implementation of mimetic finite difference method for fractured reservoirs using a fully implicit approach and discrete fracture models. J. Comput. Phys. 110665 (2021). 10.1016/j.jcp.2021.110665
42. Maryška, J., Rozložník, M., Tůma, M.: Mixed-hybrid finite element approximation of the potential fluid flow problem. J. Comput. Appl. Math. **63**(1–3), 383–392 (1995). https://doi.org/10.1016/0377-0427(95)00066-6
43. Mosé, R., Siegel, P., Ackerer, P., Chavent, G.: Application of the mixed hybrid finite element approximation in a groundwater flow model: Luxury or necessity? Water Resour. Res. **30**(11), 3001–3012 (1994). https://doi.org/10.1029/94WR01786
44. Younes, A., Ackerer, P., Delay, F.: Mixed finite elements for solving 2-D diffusion-type equations. Rev. Geophys. **48**(1), RG1004 (2010). https://doi.org/10.1029/2008RG000277
45. Younis, R.M.: Modern advances in software and solution algorithms for reservoir simulation [PhD dissertation]. Stanford University (2011). Available from:https://stacks.stanford.edu/file/druid:fb287kz3299/RMY_PHD_THESIS-augmented.pdf
46. Bui, Q.M., Elman, H.C., Moulton, J.D.: Algebraic multigrid preconditioners for multiphase flow in porous media. SIAM J. Sci. Comput. **39**(5), S662–S680 (2017). https://doi.org/10.1137/16M1082652
47. Napov, A., Notay, Y.: An algebraic multigrid method with guaranteed convergence rate. SIAM J. Sci. Comput. **34**(2), A1079–A1109 (2012). https://doi.org/10.1137/100818509
48. Notay, Y.: An aggregation-based algebraic multigrid method. Electron. Trans. Numer. Anal. **37**, 123–146 (2010)
49. Notay, Y.: Aggregation-based algebraic multigrid for convection-diffusion equations. SIAM J. Sci. Comput. **34**(4), A2288–A2316 (2012). https://doi.org/10.1137/110835347

50. Eisenstat, S.C., Elman, H.C., Schultz, M.H.: Variational iterative methods for nonsymmetric systems of linear equations. SIAM J. Numer. Anal. **20**(2), 345–357 (1983). https://doi.org/10.1137/0720023
51. Jiránek, P., Rozložník, M., Gutknecht, M.H.: How to make simpler GMRES and GCR more stable. SIAM J. Matrix Anal. Appl. **30**(4), 1483–1499 (2009). https://doi.org/10.1137/070707373
52. Falgout, R.D., Yang, U.M.: HYPRE: A library of high performance preconditioners. In: Sloot, P.M.A., Hoekstra, A.G., Tan, C.J.K., Dongarra, J.J. (eds.) Comput. Sci. — ICCS 2002, pp. 632–641. Springer, Berlin, Heidelberg (2002). Available from: http://link.springer.com/10.1007/3-540-47789-6_66
53. Balay, S., Abhyankar, S., Adams, M.F., Benson, S., Brown, J., Brune, P., et al.: PETSc/TAO users manual - ANL-21/39 - Revision 3.17. Argonne National Laboratory (2022)
54. Balay, S., Gropp, W.D., McInnes, L.C., Smith, B.F.: Efficient management of parallelism in object-oriented numerical software libraries. In: Mod. Softw. tools Sci. Comput. Boston, MA: Birkhäuser Boston, pp. 163–202. (1997). Available from: http://link.springer.com/10.1007/978-1-4612-1986-6_8
55. Christie, M.A., Blunt, M.J.: Tenth SPE comparative solution project: A comparison of upscaling techniques. In: SPE Reserv. Simul. Symp. Houston, Texas: Society of Petroleum Engineers, pp. 308–317. (2001). Available from: https://doi.org/10.2118/66599-MS
56. Saad, Y., Schultz, M.H.: GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. **7**(3), 856–869 (1986). https://doi.org/10.1137/0907058
57. Coats, K.H.: IMPES stability: Selection of stable timesteps. SPE J. **8**(02), 181–187 (2003). https://doi.org/10.2118/84924-PA
58. Saad, Y.: Iterative methods for sparse linear systems. Philadelphia, USA: Society for Industrial and Applied Mathematics (2003). Available from: http://epubs.siam.org/doi/book/10.1137/1.9780898718003
59. Ferronato, M., Franceschini, A., Janna, C., Castelletto, N., Tchelepi, H.A.: A general preconditioning framework for coupled multiphysics problems with application to contact- and poromechanics. J Comput Phys. **398**, 108887 (2019). https://doi.org/10.1016/j.jcp.2019.108887
60. Franceschini, A., Castelletto, N., Ferronato, M.: Approximate inverse-based block preconditioners in poroelasticity. Comput. Geosci. **25**(2), 701–714 (2021). https://doi.org/10.1007/s10596-020-09981-2