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

Efficient Reordered Nonlinear Gauss–Seidel Solvers With Higher Order For Black-Oil Models

Øystein S. Klemetsdal · Atgeirr F. Rasmussen · Olav Møyner · Knut-Andreas Lie

Received: date / Accepted: date

Abstract The fully implicit method is the most commonly used approach to solve black-oil problems in reservoir simulation. The method requires repeated linearization of large nonlinear systems and produces ill-conditioned linear systems. We present a strategy to reduce computational time that relies on two key ideas: (*i*) a sequential formulation that decouples flow and transport into separate subproblems, and (*ii*) a highly efficient Gauss–Seidel solver for the transport problems. This solver uses intercell fluxes to reorder the grid cells according to their upstream neighbors, and groups cells that are mutually dependent because of counter-current flow into local clusters. The cells and local clusters can then be solved in sequence, starting from the inflow and moving gradually downstream, since each new cell or local cluster will only depend on upstream neighbors that have already been computed. Altogether, this gives optimal localization and control of the nonlinear solution process.

This method has been successfully applied to real-field problems using the standard first-order finite volume discretization. Here, we extend the idea to first-order dG methods on fully unstructured grids. We also demonstrate proof of concept for the reordering idea by applying it to the full simulation model of the Norne oil field, using a prototype variant of the open-source OPM Flow simulator.

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

A.F. Rasmussen
SINTEF Digital, Norway
E-mail: atgeirr.rasmussen@sintef.no

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

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

1 Introduction

Reservoir simulation requires the solution of large systems of nonlinear partial differential equations to compute fluid pressure, phase saturations, and component concentrations/mass fractions. In industry-grade simulations, it is most common to use implicit temporal discretization to overcome severe CFL restrictions arising because of grids with high aspect ratios, orders of magnitude variations in petrophysical properties, and large variations in fluid velocities from stagnant and slow-moving flow regions to high-flow regions around wells. For cases with weak coupling between the fluid pressure and the transport of conserved quantities, one can observe significant reduction in computational costs by splitting the flow model into a pressure equation and a set of transport equations and solve them sequentially [32, 34]. To this end, it is common to use specialized solvers since the two subproblems typically have very different mathematical character: pressure equations are often close to elliptic, whereas transport equations have a strong hyperbolic character; see e.g., Bell et al. [2] and Lie [18].

The pressure equation is usually discretized by a first-order, two-point flux-approximation scheme, giving a discrete problem that can be efficiently solved by a multigrid [33, 11] or multiscale method [22]. For the transport equations, a number of methods aim to

accelerate computations by utilizing inherent locality and co-current flow properties of hyperbolic equations. Examples include streamline simulation [9, 4], *a priori* estimation of nonzero update regions [30], and use of interface-localized trust regions to determine safe saturation updates [23, 15].

Reordering methods rely on the important observation that the transport of fluid phases is always unidirectional along streamlines in the absence of gravity and capillary forces. Consequently, it is possible to order the grid cells so that the discretization matrix for the transport equations becomes lower triangular and can be solved very efficiently in a cell-by-cell manner. A branch of these methods builds upon the Cascade method [1] and reorders grid cells based on the fluid potential [17, 29]. Another branch uses topological traversal of the intercell flux graph [25, 20, 21]. The latter has the advantage that it is easy to implement using methods from standard graph theory. When gravity and capillary forces are present, the flow field will have regions of counter-current flow, which show up as cycles of mutually connected cells in the discretized flow equations. If the cells in these cycles are grouped into supernodes and solved for simultaneously, we can still use the same reordering idea.

Numerical smearing and grid-orientation effects are well-known problems in reservoir simulation. These effects are particularly evident for displacement fronts with little or no self-sharpening effects, as are often found in multicomponent models describing various mechanisms for enhanced oil recovery, e.g., polymer injection and miscible flooding. We can mitigate these effects by increasing the grid resolution and/or the formal order of the spatial discretization. Both approaches increase the computational cost by increasing the number of unknowns and/or the nonlinearity of the discretized equations, and it is therefore even more important to have a nonlinear solver with high efficacy. Most high-resolution methods—i.e., methods that deliver high formal order on smooth parts of the solution and stable propagation of discontinuities—rely on some kind of spatial reconstruction using cell-based averaged quantities from a ring of immediate cell neighbors. The resulting stencils involve cells both in the upstream and downstream direction. This means that the resulting discretization graph does not reflect the same unidirectional properties as the underlying flow equations for co-current flow.

Alternatively, one can use discontinuous Galerkin methods (dG), which were first extended to general systems of hyperbolic conservation laws by Cockburn and Shu [7, 8]. When combined with a single-point upstream mobility scheme for flux evaluation, these methods preserve the causality of the continuous flow equations. That is, the corresponding stencil is restricted to cells in the upstream direction in regions of co-current flow. After application of flux-based reordering, the discrete nonlinear flow equations are permuted to block-triangular form, with small blocks representing individual cells from regions of co-current flow and larger blocks representing regions of counter-current flow. This gives a natural localization: You start at the inflow locations and solve the nonlinear equations block-by-block toward the location of outflow. For blocks consisting of multiple cells, you can either solve for all mutually dependent unknowns simultaneously, or you can use an effective Gauss–Seidel solver that decomposes the problem to an iteration over a sequence of single-cell problems [21]. The advantage of the Gauss–Seidel approach is that it relies entirely on single-cell nonlinear solvers, which are simpler to optimize.

In this work, we combine intercell flux reordering methods with higher-order dG methods. This idea has previously been studied by Natvig et al. [26] and Eikemo et al. [10] for passive advection problems and by Natvig and Lie [25] for incompressible problems in two-phase, three-phase, and two-phase–three-component flow without gravity and capillary forces. Later, the first-order variant of the method was extended to include gravity [20] and to polymer flooding [21] with compressibility and gravity effects. Herein, we study the widely used family of black-oil equations and present a method that can handle all relevant flow effects including hysteresis and capillary forces as seen in the simulation model of the Norne oil field [31]. We also discuss how to formulate high-order dG discretization for general polyhedral grids, introduce a simple order-reduction method to prevent creating spurious oscillations, and present a new method based on blocks of cells, which is more parallel and cache efficient than solving for a single cell at a time.

2 Governing equations

The black-oil model describes conservation of three pseudo-components (water, oil, and gas), which at reservoir conditions can distribute in three phases (aqueous, oleic, and gaseous):

$$
\begin{aligned}
&\partial_t\left(\phi b_w S_w\right) + \nabla\cdot\left(b_w\mathbf{v}_w\right) - b_w q_w = 0,\\
&\partial_t\left(\phi\left[b_o S_o + b_g r_v S_g\right]\right)\\
&\quad + \nabla\cdot\left(b_o\mathbf{v}_o + b_g r_v\mathbf{v}_g\right) - \left(b_o q_o + b_g r_v q_g\right) = 0,\\
&\partial_t\left(\phi\left[b_g S_g + b_o r_s S_o\right]\right)\\
&\quad + \nabla\cdot\left(b_g\mathbf{v}_g + b_o r_s\mathbf{v}_o\right) - \left(b_g q_g + b_o r_s q_o\right) = 0.
\end{aligned}
\tag{1}
$$

Here, S_α denotes saturation, $\mathbf{v}_\alpha$ the macroscopic Darcy velocity, and q_α sources and sinks of phase α. Shrinkage

factors b_α model pressure-dependent densities ρ_α, and the gas-oil and oil-gas ratios r_s and r_v model the volume of gas dissolved in oil and oil vaporized in gas, respectively, both at standard conditions. The phase velocity $\mathbf{v}_\alpha$ is given by the multiphase extension of Darcy's law:

$$\mathbf{v}_\alpha = -\lambda_\alpha \mathbf{K} \left(\nabla p_\alpha - \rho_\alpha g \nabla z \right),$$

where $\lambda_\alpha = k_{r\alpha}/\mu_\alpha$ is the mobility of phase α; relative permeability $k_{r\alpha}$ models the reduced mobility of one phase in the presence of another, whereas μ_α as usual denotes the viscosity of phase α. The model is closed by assuming that the fluid phases fill up the entire pore space, $S_w + S_o + S_g = 1$, and that the phase pressures are related through capillary pressures:

$$p_o = p_w + P_{cow}(S_w, S_o), \quad p_g = p_o + P_{cgo}(S_o, S_g).$$

3 Sequential splitting

We use backward Euler to discretize in time. On semi-discrete form, the equation describing conservation of mass for water (1) at time $n+1$ then reads

$$\begin{aligned} \mathcal{R}_w = &\frac{1}{\Delta t} \left([\phi b_w S_w]^{n+1} - [\phi b_w S_w]^n \right) \\ &+ \nabla \cdot (b_w \mathbf{v}_w)^{n+1} - (b_w q_w)^{n+1} = 0, \end{aligned} \tag{2}$$

where we have introduced the time-step length $\Delta t = t^{n+1} - t^n$, and superscripts n and $n+1$ refer to the time step. Semi-discrete forms for the oil ($\mathcal{R}_o$) and gas ($\mathcal{R}_g$) equations are analogous. We multiply each equation by the following factors

$$\begin{aligned} \omega_w &= \frac{1}{b_w^{n+1}}, \\ \omega_o &= \frac{1}{1 - r_s^{n+1} r_v^{n+1}} \left(\frac{1}{b_o^{n+1}} - \frac{r_s^{n+1}}{b_g^{n+1}} \right), \\ \omega_g &= \frac{1}{1 - r_s^{n+1} r_v^{n+1}} \left(\frac{1}{b_g^{n+1}} - \frac{r_v^{n+1}}{b_o^{n+1}} \right), \end{aligned}$$

and sum to eliminate all terms involving saturations at time step $n+1$. This gives us a pressure equation

$$\mathcal{R}_p = \omega_w \mathcal{R}_w + \omega_o \mathcal{R}_o + \omega_g \mathcal{R}_g. \tag{3}$$

To obtain a fully discrete formulation, we introduce a grid covering our computational domain with N non-overlapping polyhedral cells $\{\Omega_i\}_{i=1}^N$. We denote the common interface of cell i and j by Γ_{ij} and the volumetric flux from cell i to cell j by

$$v_{\alpha,ij} = \int_{\Gamma_{ij}} \mathbf{v}_\alpha \cdot \mathbf{n}_{ij} \, \mathrm{d}\sigma.$$

Here, $\mathbf{n}_{ij}$ is the unit normal from cell i to cell j. Notice that conservation of mass requires that $v_{\alpha,ij} = -v_{\alpha,ji}$.

In the following, we assume that we have a pressure solver that can solve the fully discrete version of (3) to yield a total velocity field $\mathbf{v} = \mathbf{v}_w + \mathbf{v}_o + \mathbf{v}_g$, given as a set of constant fluxes $v_{ij} = v_{w,ij} + v_{o,ij} + v_{g,ij}$ over each cell interface. In the numerical experiments, unless otherwise noted, we use the standard black-oil pressure solver from the open-source MATLAB Reservoir Simulation Toolbox [18].

In a sequential solution procedure, we solve (3) with fixed saturations to obtain pressures and the total Darcy velocity $\mathbf{v}$. Given the total flux, we compute new phase fluxes using a fractional flow formulation. These are defined from the following formula in the semi-continuous case (neglecting capillary effects)

$$\mathbf{v}_\alpha = f_\alpha \left(\mathbf{v} + \mathbf{K} \sum_{\beta \neq \alpha} \lambda_\beta \left(\rho_\alpha - \rho_\beta \right) g \nabla z \right), \tag{4}$$

where we have introduced the fractional flow function

$$f_\alpha(S_w, S_o, S_g) = \frac{\lambda_\alpha(S_\alpha)}{\lambda_w(S_w) + \lambda_o(S_o) + \lambda_g(S_g)}.$$

Capillary pressure differences are accounted for with a term similar to the gravity term in (4). Using this, we solve the transport equations (1) with fixed pressure to obtain new saturations (or solution ratios). This introduces a splitting error proportional to the time step Δt that can be significant for cases with strong coupling between pressure and saturation. One can also make the solution converge towards the fully implicit solution by adding outer iterations, as described by e.g, Jenny et al. [12].

4 Discontinuous Galerkin discretization

For simplicity, we only describe the weak formulation for the water equation without capillary and gravity effects, even though gravity effects are included in our solver. Capillary effects are also included, but currently only supported in our first-order transport solver. Moreover, we drop the phase subscript w and the time superscript $n+1$. Equation (2) now reads

$$\mathcal{R}_w = \tfrac{1}{\Delta t} \left([\phi b S] - [\phi b S]^n \right) + \nabla \cdot (b f \mathbf{v}) - (b q) = 0, \tag{5}$$

where we recall that the fractional flow function f depends on all phase saturations. We multiply by a test function ψ in a function space V of arbitrarily smooth functions and integrate (by parts) over cell Ω_i:

$$\begin{aligned} \frac{1}{\Delta t} \int_{\Omega_i} \Big((\phi b S) - (\phi b S)^n \Big) \psi \, \mathrm{d}V - \int_{\Omega_i} b f \mathbf{v} \cdot \nabla \psi \, \mathrm{d}V \\ + \int_{\partial \Omega_i} [b f \psi] \, \mathbf{v} \cdot \mathbf{n} \, \mathrm{d}\sigma - \int_{\Omega_i} (b q) \psi \, \mathrm{d}V = 0. \end{aligned}$$

Here, the square brackets in the surface integral signify that the integrand is possibly discontinuous across the cell boundary. Note that we can express the surface integral as

$$\int_{\partial\Omega_i} [bf\psi]\,\mathbf{v}\cdot\mathbf{n}\,\mathrm{d}\sigma = \sum_{j\in\mathcal{N}(i)}\int_{\Gamma_{ij}} [bf\psi]\,\mathbf{v}\cdot\mathbf{n}_{ij}\,\mathrm{d}\sigma,$$

where $\mathcal{N}(i)$ denotes the set of all cells sharing a common interface with cell i. We define the upwind operator

$$u[y,\mathbf{v}_\alpha\cdot\mathbf{n}_{ij};x] = \begin{cases} y_{ij}\mathbf{v}_\alpha\cdot\mathbf{n}_{ij}, & \text{if} \quad \mathbf{v}_\alpha\cdot\mathbf{n}_{ij} > 0,\\ y_{ji}\mathbf{v}_\alpha\cdot\mathbf{n}_{ij}, & \text{if} \quad \mathbf{v}_\alpha\cdot\mathbf{n}_{ij} \le 0,\end{cases}$$
$$\text{where } y_{ij} = \lim_{x'\to x} y(x') \text{ for } x'\in\Omega_i.$$

Subscripts ij and ji denote the limiting interface value of $y(x')$ for $x\in\Gamma_{ij}$ as x' approaches x from inside cell i or cell j, respectively. If the coordinate is not important, we simply write $u[y,\mathbf{v}_\alpha\cdot\mathbf{n}_{ij}]$. Using this, we write

$$\int_{\partial\Omega_i} [bf\psi]\,\mathbf{v}\cdot\mathbf{n}\,\mathrm{d}\sigma = \sum_{j\in N(i)}\int_{\Gamma_{ij}} u\,[b, f\mathbf{v}\cdot\mathbf{n}_{ij}]\,\psi\,\mathrm{d}\sigma.$$

Since the fractional flow function is always non-negative, $\mathbf{v}$ and $\mathbf{v}_\alpha = f_\alpha\mathbf{v}$ will point in the same direction in the absence of gravity and capillary effects, and the upwind definition is explicit. However, referring back to (4), we see that the upwind definition is generally implicit in the presence of gravity/capillary effects, since quantities in the parenthesis depend on the mobilities λ_α. An equivalent explicit upwind definition has been derived by Brenier and Jaffré [5] for finite-volume methods. Herein, we employ this definition *at each cubature point* involved in evaluating the surface integrals.

With this notation, we define the following weak form of the residual equation (5) in cell i

$$\begin{aligned}&\mathcal{R}_{w,i}(S_w,S_o,S_g,\psi)\\ &= \tfrac{1}{\Delta t}\mathcal{A}_{w,i}(S_w,\psi) + \mathcal{F}_{w,i}(S_w,S_o,S_g,\psi) - \mathcal{Q}_{w,i}(\psi),\end{aligned} \tag{6}$$

where the accumulation, flux and source/sink functionals are defined as

$$\mathcal{A}_{w,i}(S_w,\psi) = \int_{\Omega_i}\Big((\phi b_w S_w) - (\phi b_w S_w)^n\Big)\psi\,\mathrm{d}V,$$

$$\begin{aligned}&\mathcal{F}_{w,i}(S_w,S_g,S_o,\psi)\\ &\quad= \sum_{j\in N(i)}\int_{\Gamma_{ij}} u\,[b_w, f_w(S_w,S_o,S_g)\mathbf{v}\cdot\mathbf{n}_{ij}]\,\psi\,\mathrm{d}\sigma\\ &\qquad - \int_{\Omega_i} b_w f_w(S_w,S_o,S_g)\mathbf{v}\cdot\nabla\psi\,\mathrm{d}V,\end{aligned}$$

$$\mathcal{Q}_{w,i}(\psi) = \int_{\Omega_i} b_w q_w\psi\,\mathrm{d}V.$$

Weak form residuals for the oil ($\mathcal{R}_{o,i}$) and gas ($\mathcal{R}_{g,i}$) can be derived in a similar fashion. To obtain a discrete weak formulation, we replace the function space V with a finite-dimensional subspace V_h consisting of functions that are smooth on each cell Ω_i, but possibly discontinuous across cell interfaces. We replace the saturations and the test function ψ with approximations $S_{\alpha,h}\in V_h$ and $\psi_h\in V_h$, and arrive at the following discrete weak formulation:

Find $S_{w,h}$,$S_{o,h}$, $S_{g,h}\in V_h$ such that
$$\mathcal{R}_{\alpha,i}(S_{w,h},S_{o,h},S_{g,h},\psi_h) = 0 \text{ for all } \psi_h\in V_h.$$

If we introduce the basis $\{\psi_k\}_{k=1}^{n_{\mathrm{dof}}}$ for V_h, we may express the saturations as $S_{\alpha,h} = \sum_{k=1}^{n_{\mathrm{dof}}} S_{\alpha,k}\psi_k$, where $S_{\alpha,k}\in\mathbb{R}$ is referred to as the kth degree of freedom of S_α. The discrete weak formulation then takes the form

Find $(S_{\alpha,1},\ldots,S_{\alpha,n_{\mathrm{dof}}})\in\mathbb{R}^{n_{\mathrm{dof}}}$, $\alpha = w,o,g,$ such that
$$\mathcal{R}_{\alpha,i}\left(\sum_{k=1}^{n_{\mathrm{dof}}} S_{w,k}\psi_k, \sum_{k=1}^{n_{\mathrm{dof}}} S_{o,k}\psi_k, \sum_{k=1}^{n_{\mathrm{dof}}} S_{g,k}\psi_k, \psi_\ell\right) = 0$$
for all $\ell = 1,\ldots,n_{\mathrm{dof}}$.

Note that we typically use the closure relation $S_w + S_o + S_g = 1$ to eliminate one variable and one equation from the above.

There are several important factors that need careful consideration to implement the above discretization. In the following, we briefly discuss the most important of these. A more detailed analysis is subject of further research.

4.1 Basis functions

In finite-element methods, it is common to use *orthogonal* basis functions $\{\psi_k\}_{k=1}^{n_{\mathrm{dof}}}$, i.e., $\int_{\Omega_i}\psi_k\psi_\ell = \delta_{k,\ell}$. The main reason for this choice is that it typically results in a significantly less dense discretization matrix. Defining orthogonal basis functions for general polyhedral grids is not a trivial problem. For the types of problems we are interested in, however, the weak form residual (6) will generally depend nonlinearly on the saturations (and thus the basis functions) due to the fractional flow function f_α in the flux terms $\mathcal{F}_{\alpha,i}$. Effectively, orthogonal basis functions will generally not lead to less dense discretization matrices. Herein, we will therefore simply use tensor products of Legendre polynomials as our basis functions. These form a convenient basis for the space of polynomials and are orthogonal for cuboid grid cells. The first two Legendre basis functions are $\ell_0(x) = 1$ and $\ell_1(x) = x$, and higher-order basis functions are defined in our dG implementation by exploiting Bonnet's recursion formula,

$$(k+1)\ell_{k+1}(x) - (2k+1)x\ell_k(x) + k\ell_{k-1}(x) = 0.$$

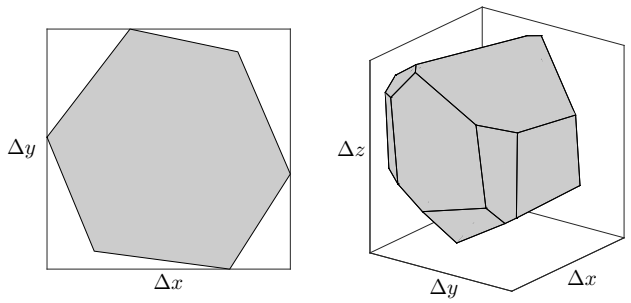

Fig. 1 Dimensions of bounding boxes used to define the dG basis functions for a polygon in 2D and a polyhedron in 3D.

Basis function j of polynomial order $r+s+t$ for cell i is then defined as

$$\psi_j(\mathbf{x}) = \begin{cases} \ell_r\left(\frac{x-x_i}{\Delta x_i/2}\right)\ell_s\left(\frac{y-y_i}{\Delta y_i/2}\right)\ell_t\left(\frac{z-z_i}{\Delta z_i/2}\right), & \mathbf{x}\in\Omega_i, \\ 0, & \mathbf{x}\notin\Omega_i, \end{cases}$$

where

$$\mathbf{x}=(x,y,z), \quad \mathbf{x}_i=(x_i,y_i,z_i), \quad \Delta\mathbf{x}=(\Delta x_i,\Delta y_i,\Delta z_i)$$

denote the spatial coordinate, cell centroid and the dimensions of the smallest cuboid aligned with the coordinate axes that completely contains cell i, as shown in Figure 1. For a dG method of degree k, the basis consists of all tensor products on this form such that $0\le r+s+t\le k$, with $r,s,t\ge 0$. In d space dimensions, this gives a total number of basis functions

$$n_{\text{dof}} = \binom{k+d}{d} = \frac{(k+d)!}{d!k!}.$$

We use the notation dG(k) to refer to a dG method of degree k. The error of a method of degree k will for smooth solutions decay with order $k+1$, so that a dG(k) method is of formal order $k+1$.

4.2 Cubature rules

The integrals in (6) must be numerically evaluated. Efficient cubature rules for general polyhedral grid cells are crucial to obtain an efficient implementation. A straightforward approach to numerically evaluate integrals over polyhedral cells is to partition the cell into non-overlapping simplices and apply a known cubature rule for these. This uses significantly more cubature points than strictly needed for the cubature to be correct and is thus highly inefficient from a computational point of view. Instead, we apply an approach based on moment-fitting, see e.g., [24]. This approach defines the quadrature rule for a cell Ω (omitting subscript i) as the solution to a small linear system

$$\begin{bmatrix} \psi_1(\mathbf{x}_1) & \dots & \psi_1(\mathbf{x}_{n_{\text{dof}}}) \\ \vdots & \ddots & \vdots \\ \psi_{n_{\text{dof}}}(\mathbf{x}_1) & \dots & \psi_{n_{\text{dof}}}(\mathbf{x}_{n_{\text{dof}}}) \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_{n_{\text{dof}}} \end{bmatrix} = \frac{1}{|\Omega|}\begin{bmatrix} \int_\Omega \psi_1 \,\mathrm{d}V \\ \vdots \\ \int_\Omega \psi_{n_{\text{dof}}}\,\mathrm{d}V \end{bmatrix}, \quad \text{or} \quad \Psi\mathbf{w}=\mathbf{b}.$$

The integrals (or moments) on the right-hand side can be evaluated using a sub-optimal cubature rule. If we pick the cubature points $\mathbf{x}_0,\dots\mathbf{x}_{n_{\text{dof}}}$ so that the matrix Ψ is invertible, we can solve for the quadrature weights $w_1,\dots w_{n_{\text{dof}}}$ to obtain a cubature rule with n_{dof} points for each cell. Note that it is possible to construct a quadrature rule of precision k with less than n_{dof} points by eliminating points with marginal significance. This computation can be done once for each grid cell in a preprocessing step.

In 3D, the surface integrals in the second term of $\mathcal{F}$ in (6) are evaluated in the same way as areal integrals in 2D, using the moment-fitting approach described above. In 2D, the surface integrals are simple line integrals, for which a standard Gauss cubature rule is employed.

4.3 Velocity interpolation

As stated above, we assume that the solution to the pressure equation yields a total velocity as a set of fluxes that are constant on each interface. We see from (6) that for linear and higher-order basis functions, we need information about the velocity $\mathbf{v}$ also inside the cell, meaning that we must interpolate from the interface fluxes. Herein, we apply a simple scheme inspired by the mimetic finite difference method [19]. This gives a constant velocity inside the cell, consistent with the volumetric interface fluxes v_{ij}:

$$\mathbf{v}_i = \frac{1}{|\Omega_i|}\sum_{j\in N(i)} v_{ij}(\mathbf{x}_{ij}-\mathbf{x}_i).$$

Here, $\mathbf{x}_i$ and $\mathbf{x}_{ij}$ refer to the centroid of cell i and interface ij, respectively. The velocity on an interface is assumed to be constant. More sophisticated interpolation schemes like the extended corner-velocity interpolation scheme (ECVI) [13] have earlier been investigated for dG methods for flow diagnostics [28], but will for simplicity not be applied herein.

4.4 Order reduction

Higher-order methods usually require a slope limiter, or some other means to prevent the creation of over- and under-shoots or other types of nonphysical oscillations near spatial discontinuities in the solution. Natvig et al. [26] previously showed that order reduction followed by local and dynamic grid refinement could be used as an alternative strategy to avoid oscillations and give high-resolution in a robust and cost-efficient manner. Herein, we use a simplified version: Whenever the jump across an intercell interface exceeds a prescribed tolerance, we reduce the local approximation space to dG(0). The same is done when the solution exceeds its domain of definition by a small factor ε.

5 Localized nonlinear solvers

This section discusses how we can utilize certain unidirectional flow properties to formulate efficient nonlinear solution strategies that localize the Newton procedure to avoid spending unnecessary iterations in regions of the reservoir where many iterations are not needed.

5.1 Reordering based on intercell fluxes

An important physical property of the transport (1) is that the flow is unidirectional along streamlines if no gravity or capillary effects are present. Each dG basis function is restricted to a single cell only, and hence the only intercell interactions in the weak form $\mathcal{R}_{w,i}$ in (6) are due to the upwind operator. We split the neighboring cells $\mathcal{N}(i)$ of i into upstream cells $\mathcal{U}(i)$ and downstream cells $\mathcal{D}(i)$ based on the total flux $\mathbf{v}\cdot\mathbf{n}_{ij}$:

$$\begin{aligned}\sum_{j\in\mathcal{N}(i)}\int_{\Gamma_{ij}} u\left[b, f\mathbf{v}\cdot\mathbf{n}_{ij}\right]\psi\,\mathrm{d}\sigma \\ = \sum_{j\in\mathcal{U}(i)}\int_{\Gamma_{ij}}(bf)_j\,\mathbf{v}\cdot\mathbf{n}_{ij}\psi\,\mathrm{d}\sigma \qquad \text{(Upstream)} \\ + \sum_{j\in\mathcal{D}(i)}\int_{\Gamma_{ij}}(bf)_i\,\mathbf{v}\cdot\mathbf{n}_{ij}\psi\,\mathrm{d}\sigma, \quad \text{(Downstream)}\end{aligned}$$

where $(\cdot)_i$ indicates that quantities inside the parentheses should be evaluated from cell i. Recall that all other unknowns in the weak-form residual $\mathcal{R}_{w,i}$ have support limited to cell i, and hence the inherent unidirectional property of the continuous equations (1) is preserved by the discretization. In practice, this means that if we know the solution in all upstream cells $\mathcal{U}(i)$, the only unknowns in $\mathcal{R}_{w,i}$ are the ones associated with cell i. That is, if we now perform a topological sort of the directed, acyclic graph (DAG) induced by the intercell fluxes and use this to reorder the cells, we can solve the transport equations cell-by-cell by traversing the sorted graph. The algebraic interpretation of this is the following: If we linearize the nonlinear transport equations for all cells simultaneously, and permute the system according to the topological order, we obtain a lower-triangular matrix. Note, however, that we never assemble the discretization matrix for the full system in the reordering solution procedure. Instead we solve the nonlinear transport equations $\mathcal{R}_{\alpha,i}=0$ cell-by-cell. This way, we avoid expensive linearizations of large systems of nonlinear equations.

Figure 2 illustrates the reordering principle applied to a dG(0) scheme for a quarter-five spot test case formulated on a 2D Voronoi grid. In the original grid, the cells are ordered almost, but not entirely, by the x-coordinate of their centroids, giving an irregular sparsity pattern with entries both above and below the diagonal. After reordering, the cells are ordered such that if $j\in\mathcal{N}(i)$, then $j\in\mathcal{U}(i)$ for $j<i$ and $j\in\mathcal{D}(i)$ for $j>i$, where we recall that $\mathcal{N}(i)$ denotes neighbors and $\mathcal{U}(i)$ and $\mathcal{D}(i)$ are cells that lie upstream/downstream of cell i.

Figure 3 similarly illustrates the difference between a dG(0) and a dG(1) discretization on a Cartesian grid. The two matrices have exactly the same flux graph, and thus obtain the same cell ordering. However, whereas dG(0) gives a nonlinear scalar problem in each cell, dG(1) gives a 3×3 nonlinear system.

When gravity and capillary effects are included, the flow will no longer be unidirectional along streamlines. As a result, the intercell flux graph may contain cycles, so that the reordering method cannot be applied directly. However, we may construct a DAG from the intercell flux graph by grouping cells that are part of the same cycle into a single supernode. By topologically sorting this DAG, we end up with a block-triangular system in which each block consists of degrees of freedom that are mutually coupled and must be solved for simultaneously. Note that as explained above, we evaluate the upstream direction for each quadrature point at the interface. For a higher-order method, the upstream direction may thus vary along a single interface when gravity/capillary effects are included, particularly when we use high polynomial order and/or cells with large vertical extent. This means that each quadrature point on interfaces with changing upstream direction must be treated as an edge in the original intercell flux graph, and such interfaces will automatically introduce a local cycle and require the neighboring cells to be grouped into a supernode. In general, the reordering procedure will result in a DAG with n_c nodes, where n_c equals the

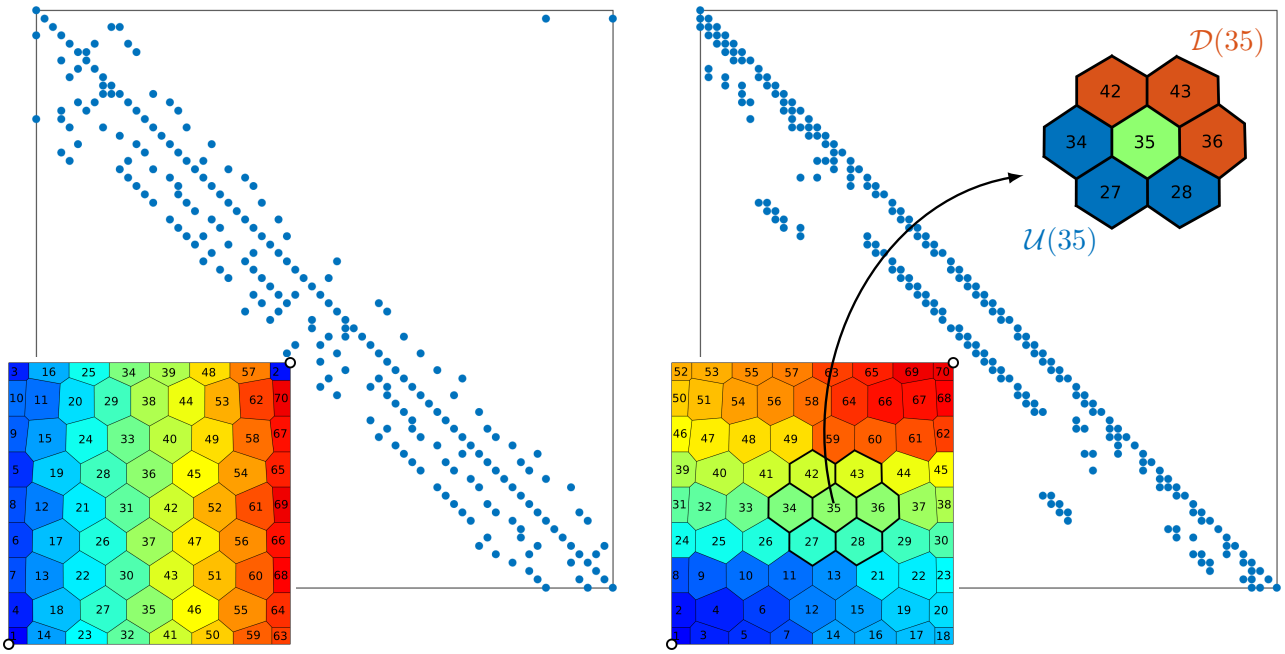

Fig. 2 Sparsity pattern for a dG(0) method formulated on a Voronoi grid with original (left) and reordered (right) numbering of cells. The inset shows grid cell 35 and its upstream $\mathcal{U}(35)$ and downstream $\mathcal{D}(35)$ neighbors.

number of connected components of the original inter-cell flux graph. Topologically sorting this graph gives us an ordering

$$\mathcal{P} = (\mathcal{P}_1, \ldots, \mathcal{P}_{n_c}), \quad \mathcal{P}_k = (i_1, \ldots, i_{n_k}),$$

where each $\mathcal{P}_k$ is a set of cell indices, and n_k is the number of cells in connected component k. The number of connected components n_c may vary from one in the worst case, where all cells are connected in a single cycle, to the total number of cells N in the best case, where the problem can be solved cell by cell without the need of any supernodes. Given two connected components $\mathcal{P}_k$ and $\mathcal{P}_\ell$, we will then have that if $i > j$ for a cell $i \in \mathcal{P}_k$ and a cell $j \in \mathcal{P}_\ell$ then $i > j$ for all cells $i \in \mathcal{P}_k$, $j \in \mathcal{P}_\ell$. Figure 4 illustrates different scenarios of coupling, from an almost completely reordered system, to a system where all cells belong to a single cycle.

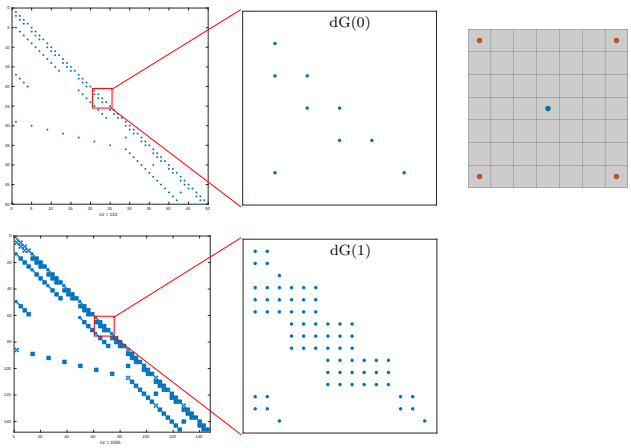

Fig. 3 Sparsity pattern after reordering for dG(0)/dG(1) for a five-spot problem on a rectangular grid.

5.2 Block-wise processing

For test cases with only viscous forces, the reordered system can be solved cell-by-cell in a single pass. This is optimal in the sense that it minimizes the number of cell-wise nonlinear solves over the simulation. However, it is not necessarily optimal in terms of runtime; most modern workstations contain several CPU cores, as well as any number of SIMD-type registers. In this setting, feeding sufficient computations to the CPU is often a large bottleneck and a barrier to rapid execution time. For this reason, our solver has an optional block-wise algorithm that solves a prescribed number of cells simultaneously. If the largest ordering index of the previous solve was i, this algorithm then solves for indices $i + 1$ to $i + n_b$, where n_b is the block size. This gives a larger system, and individual cells can follow a sub-optimal solution path when solved by a multi-cell Newton–Raphson solver. On the other hand, the startup cost is incurred only once per block of cells and not for every cell, and the block solve utilizes cache better and leaves less computational resources idle. Hence,

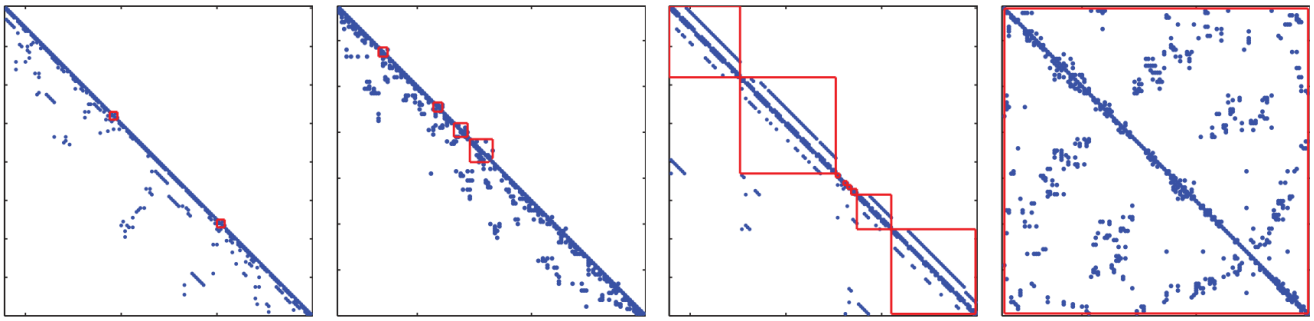

Fig. 4 Examples of different degrees of coupling, where the cycles can be seen as blocks on the diagonal of the dG(0) discretization matrix. In the first sparsity pattern, the system is almost completely reorderable with only two small cycles; in the second, a small fraction of the cells belongs to four small cycles; the third patterns shows an example where almost all cells belongs to eight larger cycles; finally the last pattern shows a case where all cells belong to one huge cycle. All cells in the same cycle must be solved simultaneously. Figure from [21].

simultaneous solve of n_b cells will typically be significantly faster than n_b single-cell solvers.

5.3 Nonlinear Gauss–Seidel solvers

As explained above, gravity will result in a block-triangular system in which each block consists of degrees of freedom that are mutually coupled and must be solved for simultaneously. The size of each block depends on whether the block represents degrees of freedom that are part of a co-current or counter-current flow region. Blocks in co-current regions represent the discrete system posed inside a single cell and are treated as described above. Blocks in counter-current regions consist of multiple cells that are coupled in cycles. In principle, each of these cycles of cells should be solved by a localized Newton iteration. Another possibility is to first solve the water equations cell-by-cell in the order of descending water phase potential, and then solve the oil equations in the order of descending oil phase potential [17]. This method requires that the oil component equation only depends on water and oil saturation, which is the case for black-oil models. Herein, we use a nonlinear Gauss–Seidel strategy instead that simply loops through all the cells of a cycle and solves them one at a time, using the values in all neighboring cells as if they were all known upstream values. This will typically not give the correct solution after a single iteration through the loop, but a large number of numerical experiments indicate that the process converges after only a few repetitions. Capillary forces will in the worst case couple all cells in one big cycle, as seen in the rightmost sparsity pattern in Figure 4. However, the coupling is oftentimes so weak that a Gauss–Seidel type iteration is still very effective.

6 Solution procedure

The entire procedure to evolve the solution one time step can now be summarized as follows: (*i*) Solve the pressure equation (3) to obtain intercell fluxes v_{ij} and oil phase pressure p_o; (*ii*) construct a DAG from the intercell flux graph by grouping cells that are part of the same cycle into a single supernode; (*iii*) starting from the first node in the sorted graph, solve the transport equations (6) cell-by-cell or cycle-by-cycle in topological order. The procedure is summarized in Figure 5. Notice the important property that if the residual of a cell or cycle is converged at the beginning of the time step, and none of the upstream neighbors were updated in this step, we do not need to call the nonlinear Gauss–Seidel solver and can proceed to the next component. Effectively, we thus avoid solving for a large number of zeros, and can instead focus our computational resources on parts of the domain where updates are nonzero.

7 Numerical examples

We have implemented our reordering approach in two different open-source simulator platforms. The MATLAB Reservoir Simulation Toolbox (MRST) [18] is a general toolbox aimed at rapid development of proof-of-concept simulators and workflow tools. We have used MRST to develop a combined dG–reordering method that allows for cell-based refinement. The solver uses automatic differentiation to linearize the localized systems and is as such applicable to a wide variety of multiphase flow models. At the time of writing, the implementation in MRST has only been tested for two-phase models with varying degrees of compressibility.

OPM Flow [27] is developed with the goal of providing industry-grade simulations and generally offers single-core computational performance that is compa-

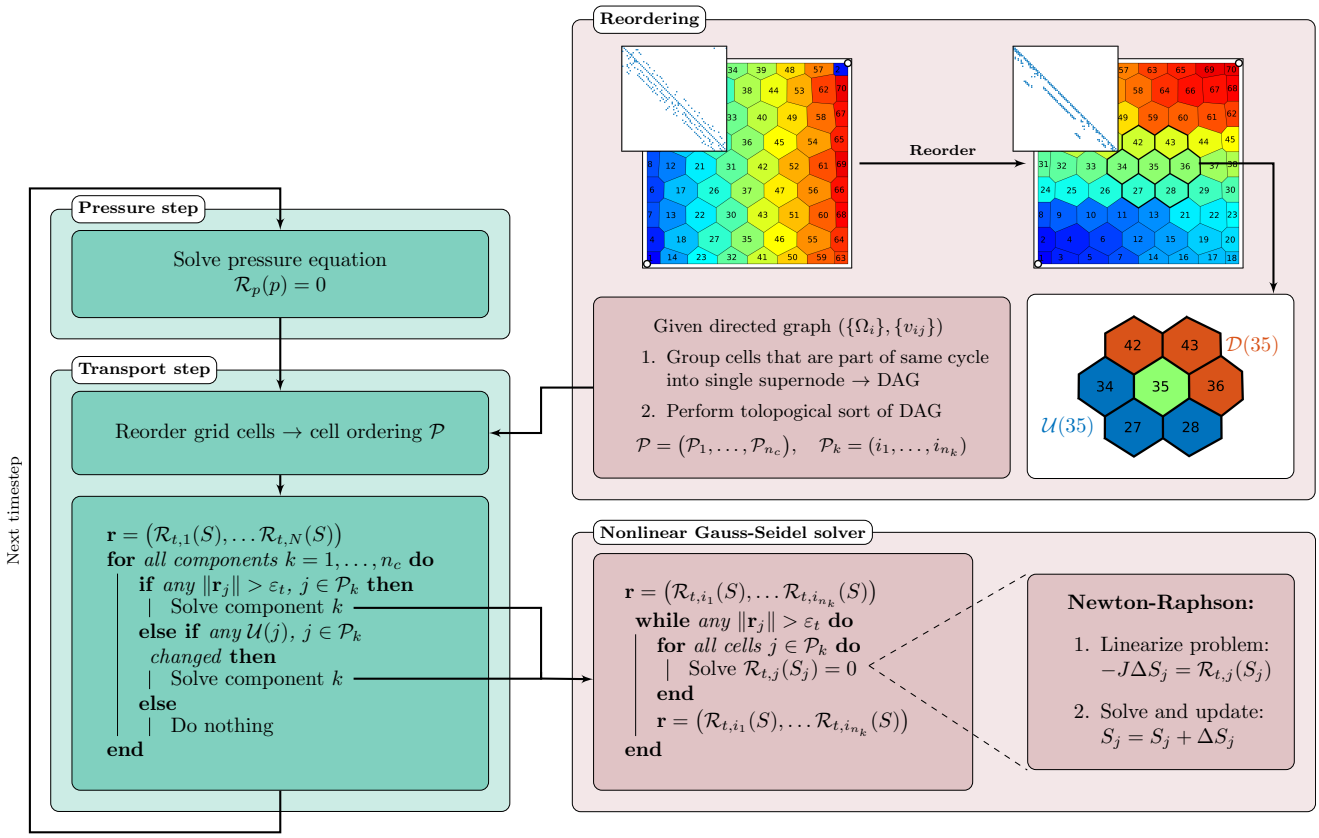

Fig. 5 Schematic overview of the solution procedure. The pressure equation is solved with a suitable solver, resulting in intercell fluxes v_{ij}, which we may interpret as a directed graph with grid cells as nodes, and intercell fluxes as edge weights. By grouping all cells that are part of the same cycle into a (super)node, we obtain a directed, acyclic graph (DAG). A topological sort of this graph gives us a node (or component) ordering $\mathcal{P}$. By traversing the nodes in this order, we can solve the transport problem $\mathcal{R}_t(S) = 0$ node by node, since the cells in each node in the DAG only depend on each other and cells from upstream (and already resolved) nodes.

rable with contemporary commercial simulators. We have used this as a platform to implement dG(0) with reordering for a general three-phase black-oil model with compressibility, dissolution and vaporization, gravity and capillary forces, and hysteresis. Our implementation uses a simple, global nonlinear Gauss–Seidel method that traverses all cells repeatedly until the residual of the transport equations are below a prescribed tolerance in all cells.

7.1 The Norne oil field

The Norne oil field in the Norwegian Sea is one of the few real assets for which simulation models and other data have been made openly available [31]. Over the 3,312 days of production history, 36 wells are active. We study a water-alternating-gas (WAG) injection scenario, in which the wells are controlled by reservoir-volume rates to match the historical rates to obtain a reasonable pressure development. The purpose of the example is to demonstrate that the sequential splitting method is capable of capturing the correct behavior of an industry-grade simulation run in history-matching mode.

We ran the case with two simulators from OPM: the fully implicit `Flow` simulator and the experimental `Flow-reorder` simulator, which uses sequential splitting and reordering of the transport equations. We note that these simulators both support a slightly more complex fluid model than that described in the introduction, incorporating hysteresis effects for relative permeability. Both simulators are available from OPM in source and binary form [27].

We highlight the behavior of two different wells to compare the fully implicit to the sequential/reordering solution. Well “D-3AH” is a typical producer and shows good match between the fully implicit and reordering solutions for both bottom-hole pressure and rates; see the upper row in Figure 6. Well “B-2H” is also a producer, which we have chosen to highlight because it shows the worst match between the two solutions. As can be seen in the lower row of Figure 6, gas and oil

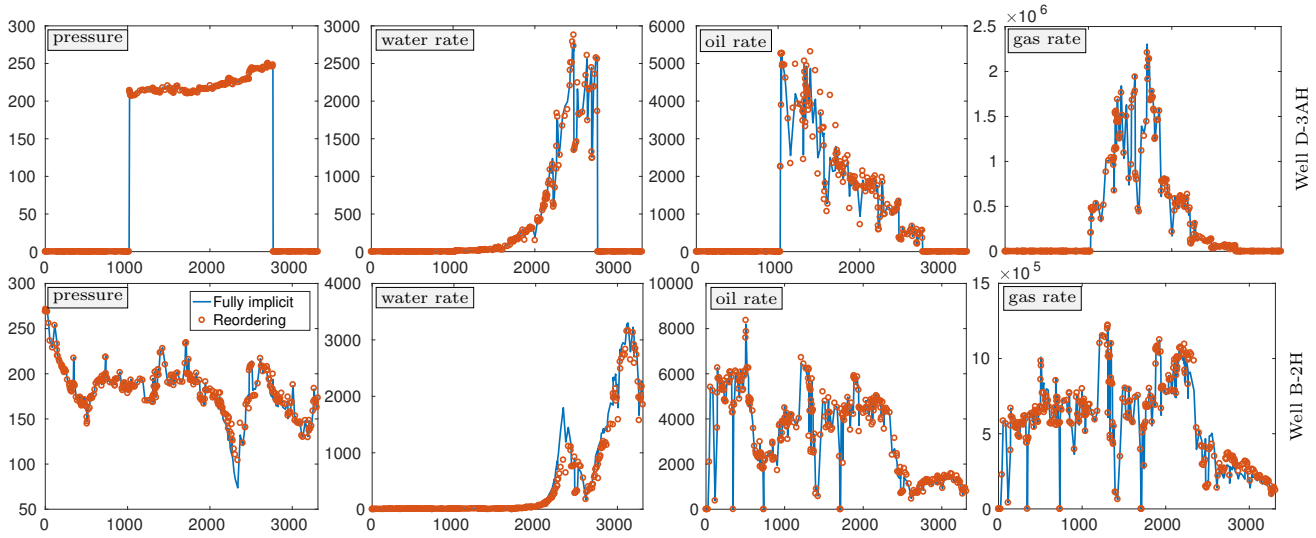

Fig. 6 Simulation responses for two of the many wells in the Norne field model. We plot the bottom-hole-pressure and production rates for water, oil and gas at surface conditions.

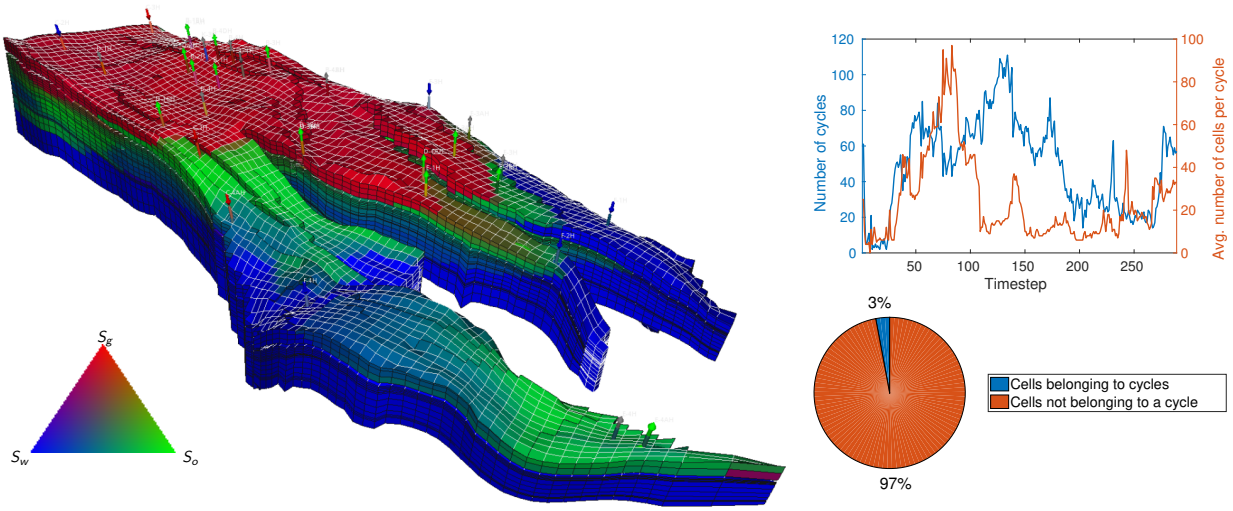

Fig. 7 Initial fluid saturations and wells (left); aggregate statistics for the number of cycles (right) for the Norne field model in Example 7.1.

production rates match well, but water-production rate and bottom-hole pressure show significant deviation some time after water breakthrough. There are two main factors that contribute to the difference: First of all, the sequential splitting in `Flow-reorder` does not use outer iterations. The overall solution algorithm is thus not guaranteed to reduce the overall residual at the end of each time step to the same tolerance as the fully implicit method in `Flow`. Moreover, convergence is measured somewhat differently in the global Gauss–Seidel and the Newton–Raphson methods.

In our opinion, this simulation is proof of concept for the reordering idea in a production-grade code running a full simulation of a real field in history mode. This is a first step, and there are several ways in which this implementation can be improved. The important observation is that we made no simplifications in the description of geology, fluid behavior, and well and production facilities.

Figure 7 shows the fluids in place and reports the occurrence of cycles throughout the simulation. Altogether, the model has approximately 44,000 active cells. The number of cycles and the average number of cells in each cycle vary largely throughout the simulation because of significant changes in well controls and fluid composition. In any time step, the number of cells belonging to cycles only represents a fraction of the total cells. This shows the inherent locality and the speedup

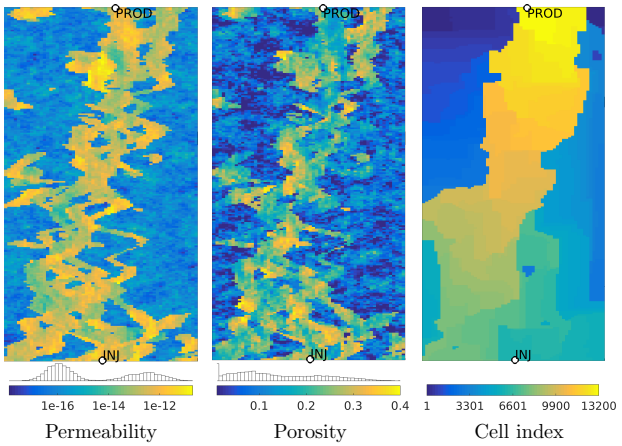

Fig. 8 Petrophysical properties for Layer 50 from the SPE10 Benchmark. The right plot shows the cell index after reordering.

potential, since most of the cells can be solved independently.

7.2 Subset of the SPE10 Model 2

Next, we consider a 2D simulation model with petrophysical properties sampled from Layer 50 from the SPE10 Model 2 Benchmark [6]. This layer is part of a fluvial formation and consists of an intertwined pattern of high-permeable sand channels on a background of low-permeable mudstone. We inject 0.2 pore volumes of water at a constant rate in one end of a channel, and produce at a fixed bottom-hole pressure of 275 bars at the opposite end of the same channel. The reservoir is initially filled with a mixture of water and oil, and the rock and fluids are weakly compressible. Water and oil have quadratic relative permeabilities with residual saturations of 0.2, and viscosities of 2.85 and 3.0 cP, respectively. Permeability and porosity are depicted in Figure 8. The figure also shows the cell index after reordering. As expected, the cell number increases from the injector and along the high-flow channels to the producer. Interestingly, the model contains cycles of cells that are *not* connected to the injector–producer pair by nonzero fluxes. This is an effect of fluid compressibility, and the reordering algorithm assigns lower indices to cells which essentially should remain inactive throughout the transport time step.

We simulate using dG(0) and dG(1) with reordering implemented in MRST. Figure 9 shows the resulting water saturation at selected time steps, whereas Figure 10 shows the iterations used per active cell in the corresponding time steps. By active cells, we mean cells in which the water saturation was updated during the time step either because the residual at the beginning of the time step was above the nonlinear tolerance, or because one or more of the cell's upstream neighbors were updated during the time step. Notice that iterations are performed only in a small fraction of the cells in each time step and that the updates are chiefly located to the fluvial channel. We also see that the higher-order solution profile is much less diffuse. Effectively, the water channel fills faster, so that dG(1) predicts earlier water breakthrough than dG(0).

For comparison, Figure 11 also reports the (average) number of nonlinear Newton–Raphson iterations used to solve each time step. For the reordered solver, we report the average number of nonlinear iterations used per cell, defined over all cells, and the maximum number of iterations observed in the cell with slowest nonlinear convergence. Because large fractions of the cells remain inactive in most of the time steps, the average number of cellwise nonlinear iterations is far below one in all time steps. This confirms previous observations by Natvig and Lie [25]. Interestingly, we see that the maximum number of nonlinear iterations for the reordered solver in most time steps is *less than* the number of Newton–Raphson solves used when solving for all cells simultaneously for the same time step. This indicates that the iteration path followed by individual cells in a Newton–Raphson solve of all cells simultaneously is not necessarily optimal.

7.3 3D Voronoi grid

In this example, we study the effect of using higher-order dG methods and reordering to simulate a case posed on a 3D, fully unstructured PEBI/Voronoi grid generated using the `upr` module in MRST [3, 14]. The domain is comprised of a rectangular box of dimensions $300 \times 100 \times 100$ m^3 with approximately 20, 10, and 10 cells in each of the axial directions, respectively. Porosity and permeability are sampled from the top layers of Model 2 from SPE10 [6]. The injector is placed in a cell at the center of the left boundary. We inject water at constant bottom-hole pressure of 600 bars. A producer operated at a fixed bottom-hole pressure of 50 bars is placed at the center of the right boundary. Figure 12 shows petrophysical properties and well positions.

We simulate the case using the dG(0) and dG(1) solvers from MRST with and without reordering. For the reordering algorithm, we use the block version with a block size n_b of 100 cells. Figure 13 reports saturation profiles at three selected time steps. Compared with dG(1), the dG(0) saturation profiles are more diffusive and fill more of the reservoir cross-section. As a result, dG(1) predicts a higher water-cut in the producer, as reported in Figure 15.

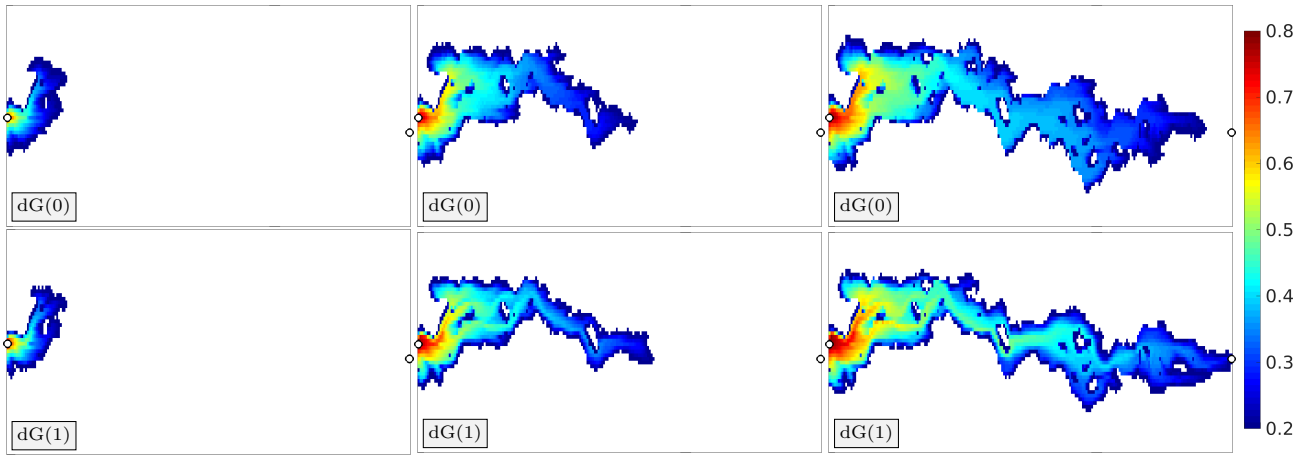

Fig. 9 Evolution of the water saturation for three different times in the channelized reservoir in Example 7.2. The upper row is for dG(0) and the lower row for dG(1).

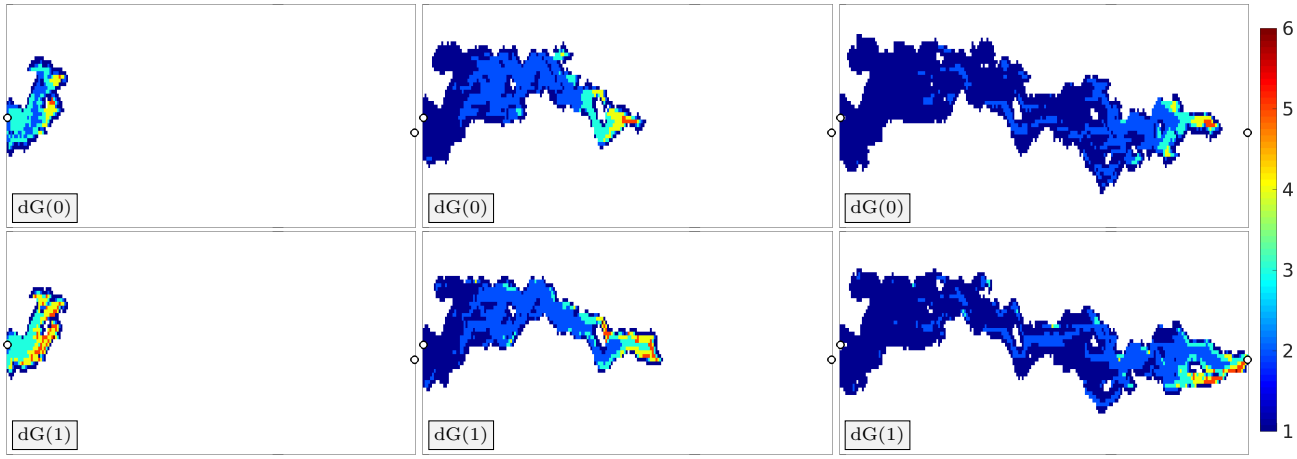

Fig. 10 Number of nonlinear iterations per cell at three instances in time corresponding to the solution profiles shown in Figure 9. Zero iterations were performed for cells in white. The upper row is for dG(0) and the lower row for dG(1).

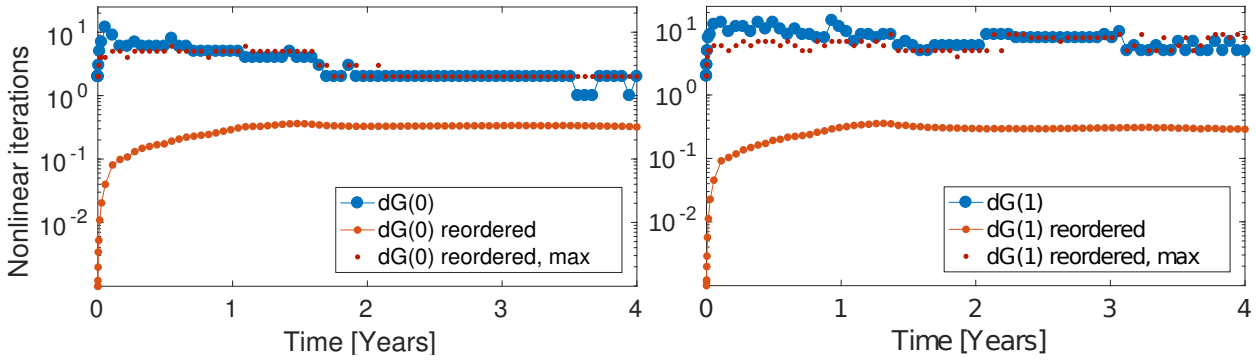

Fig. 11 Comparison of average number of nonlinear iterations per cell as function of time for Example 7.2. Note that the y-axis is logarithmic.

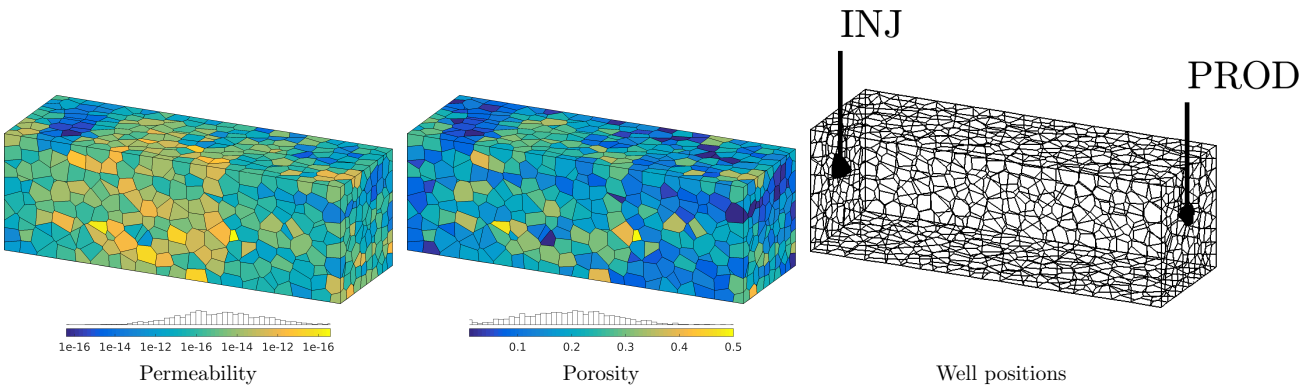

Fig. 12 Petrophysical properties and well setup for the Voronoi-example.

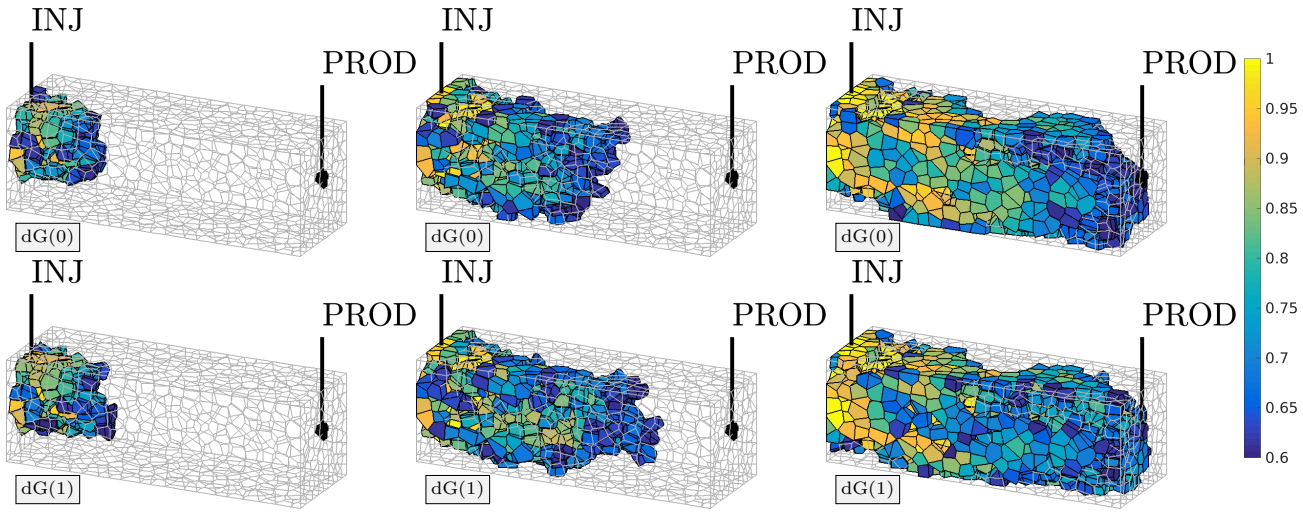

Fig. 13 Thresholded water saturation profiles for the two dG solutions at three selected time steps for Example 7.3.

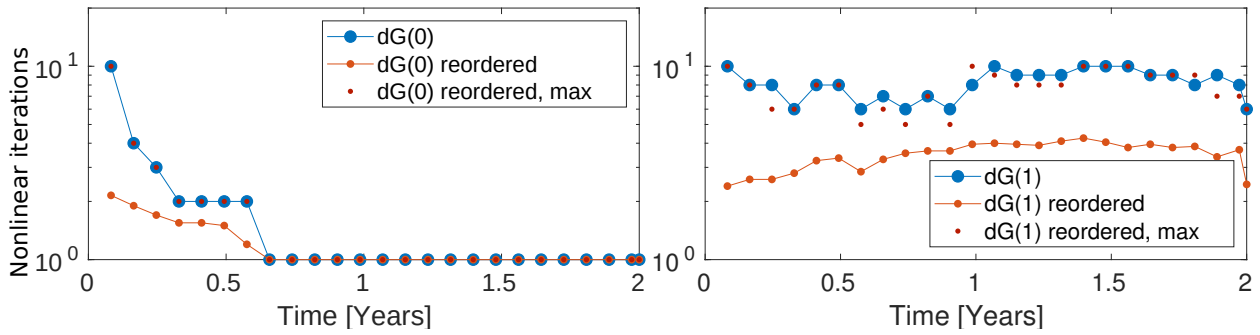

Fig. 14 Average number of cellwise nonlinear iterations per time step used for each of the solvers in Example 7.3, with dG(0) in the left plot and dG(1) in the right plot.

The solutions computed with the ordered and the original Newton–Raphson solvers are identical to plotting resolution, which verifies the correctness of the localization and the new block algorithm. Finally, we look at the number of iterations used by the different solvers, shown in Figure 14. We observe that even with a block size of $n_b = 100$ cells, the average number of nonlinear iterations per cell is significantly smaller than the number of iterations used when solving for all cells simultaneously.

8 Concluding remarks

The transport substep in a sequentially implicit solution procedure for black-oil models can be significantly accelerated by ordering the grid cells in the simula-

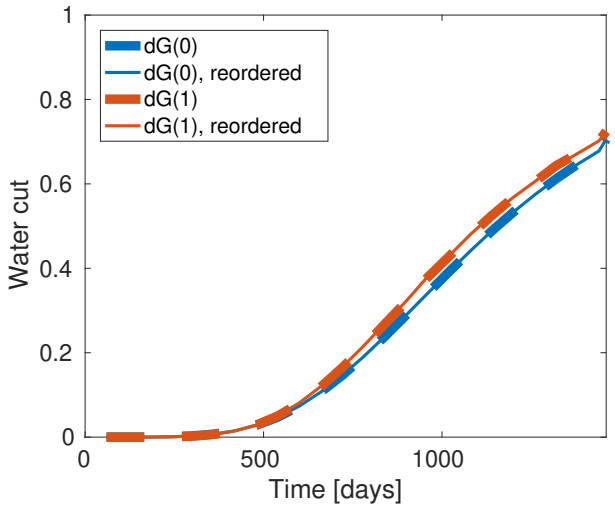

Fig. 15 Water cut in the production well in Example 7.3.

tion model according to total velocity computed in a proceeding pressure step, so that the residual transport equations can be solved cell-by-cell from inflow to outflow. This not only localizes the nonlinear Newton solver, so that the computational effort can be focused on cells where the transported quantities change significantly during each time step, but also contributes to improve the general robustness of the Newton solver. As a verification of this concept in an industry-grade setting, we have used a sequential simulator from the open-source OPM initiative to demonstrate that a reordered, global nonlinear Gauss–Seidel strategy computes an acceptable solution for the full simulation model of the Norne oil field. Additional verifications have also been obtained using similar industry-grade simulators from MRST.

The ordering approach is also applicable to transport solvers based on discontinuous Galerkin discretizations in combination with single-point upstream mobility weighting. Such discretizations are detailed herein for general 3D cases with (almost) no assumptions on the polyhedral cell geometries and grid topology. By reordering cells optimally, one can ensure that the added computational cost of solving for more degrees of freedom to obtain higher spatial order remains local to each cell. Our results obtained using MRST show that a second-order dG(1) method has significantly improved accuracy compared with the standard first-order volume method used in industry at the expense of only a moderate increase in the number of nonlinear iterations.

Herein, we have only presented simulations of two-phase oil-water and three-phase black-oil systems, but the same principles are applicable to a wide variety of different black-oil type and compositional models used to described secondary and tertiary recovery processes. As an example, Klemetsdal et al. [16] discuss several cases with significantly more complex flow physics, including compositional flow and gas injection with strong gravity effects, as well as use of dynamically adaptive grid coarsening.

9 Acknowledgements

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

References

1. J. R. Appleyard and I. M. Cheshire. The cascade method for accelerated convergence in implicit simulators. In *European Petroleum Conference, 25-28 October, London, United Kingdom*. Society of Petroleum Engineers, October 1982. doi: 10.2118/12804-MS.
2. J. B. Bell, J. A. Trangenstein, and G. R. Shubin. Conservation laws of mixed type describing three-phase flow in porous media. *SIAM Journ. Appl. Math.*, 46(6):1000–1017, 1986. doi: 10.1137/0146059.
3. R. L. Berge, Ø. S. Klemetsdal, and K.-A. Lie. Unstructured voronoi grids conforming to lower dimensional objects. *Comput. Geosci.*, 23(1):169–188, 2019. doi: 10.1007/s10596-018-9790-0.
4. F. Bratvedt, T. Gimse, and C. Tegnander. Streamline computations for porous media flow including gravity. *Transp. Porous Media*, 25(1):63–78, 1996. doi: 10.1007/BF00141262.
5. Y. Brenier and J. Jaffré. Upstream differencing for multiphase flow in reservoir simulation. *SIAM J. Numer. Anal.*, 28(3):685–696, 1991. doi: 10.1137/0728036.
6. M. A. Christie and M. J. Blunt. Tenth SPE comparative solution project: A comparison of upscaling techniques. *SPE Reservoir Eval. Eng.*, 4:308–317, 2001. doi: 10.2118/72469-PA.
7. B. Cockburn and C.-W. Shu. TVB runge-kutta local projection discontinuous galerkin finite element method for conservation laws ii: General. *Math. Comp.*, 52(186):411–435, 1989. doi: 10.1090/S0025-5718-1989-0983311-4.
8. B. Cockburn and C.-W. Shu. The runge-kutta local projection p^1- discontinuous-galerkin finite element method for scalar conservation laws. *ESAIM–Math. Model. Num.*, 25(3):337–361, 1991.
9. A. Datta-Gupta and M. J. King. *Streamline simulation: theory and practice*, volume 11 of *SPE Textbook Series*. Society of Petroleum Engineers, 2007.
10. B. Eikemo, K.-A. Lie, H. K. Dahle, and G. T. Eigestad. Discontinuous Galerkin methods for transport in advective transport in single-continuum models of fractured media. *Adv. Water Resour.*, 32(4):493–506, 2009. doi: 10.1016/j.advwatres.2008.12.010.
11. S. Gries, K. Stüben, G. L. Brown, D. Chen, and D. A. Collins. Preconditioning for efficiently applying algebraic

multigrid in fully implicit reservoir simulations. *SPE J.*, 19(04):726–736, 2014. doi: 10.2118/163608-PA.
12. P. Jenny, S. H. Lee, and H. A. Tchelepi. Adaptive fully implicit multi-scale finite-volume method for multi-phase flow and transport in heterogeneous porous media. *J. Comput. Phys.*, 217(2):627–641, 2006. doi: 10.1016/j.jcp.2006.01.028.
13. R. A. Klausen, A. F. Rasmussen, and A. Stephansen. Velocity interpolation and streamline tracing on irregular geometries. *Comput. Geosci.*, 16:261–276, 2012. doi: 10.1007/s10596-011-9256-0.
14. Ø. S. Klemetsdal, R. L. Berge, K.-A. Lie, H. M. Nilsen, and O. Møyner. Unstructured gridding and consistent discretizations for reservoirs with faults and complex wells. In *SPE Reservoir Simulation Conference*. Society of Petroleum Engineers, 2017.
15. Ø. S. Klemetsdal, O. Møyner, and K.-A. Lie. Robust nonlinear newton solver with adaptive interface-localized trust regions. *SPE J.*, 2019. doi: 10.2118/195682-PA.
16. Ø. S. Klemetsdal, O. Møyner, and K.-A. Lie. Implicit high-resolution compositional simulation with optimal ordering of unknowns and adaptive spatial refinement. In *SPE Reservoir Simulation Conference, 10-11 April, Galveston, Texas, USA*. Society of Petroleum Engineers, 2019. doi: 10.2118/193934-MS.
17. F. Kwok and H. Tchelepi. Potential-based reduced newton algorithm for nonlinear multiphase flow in porous media. *J. Comput. Phys.*, 227(1):706 – 727, 2007. doi: 10.1016/j.jcp.2007.08.012.
18. K.-A. Lie. *An Introduction to Reservoir Simulation Using MATLAB/GNU Octave: User guide for the Matlab Reservoir Simulation Toolbox (MRST)*. Cambridge University Press, 2016.
19. K.-A. Lie, S. Krogstad, I. S. Ligaarden, J. R. Natvig, H. M. Nilsen, and B. Skaflestad. Open source MATLAB implementation of consistent discretisations on complex grids. *Comput. Geosci.*, 16:297–322, 2012. ISSN 1420-0597. doi: 10.1007/s10596-011-9244-4.
20. K. A. Lie, J. R. Natvig, and H. M. Nilsen. Discussion of dynamics and operator splitting techniques for two-phase flow with gravity. *Int. J. Numer. Anal. Mod.*, 9 (3):684–700, 2012.
21. K.-A. Lie, H. M. Nilsen, A. F. Rasmussen, and X. Raynaud. Fast simulation of polymer injection in heavy-oil reservoirs on the basis of topological sorting and sequential splitting. *SPE J.*, 19(06):0991–1004, 2014. doi: 10.2118/163599-PA.
22. K.-A. Lie, O. Møyner, J. R. Natvig, A. Kozlova, K. Bratvedt, S. Watanabe, and Z. Li. Successful application of multiscale methods in a real reservoir simulator environment. *Comput. Geosci.*, 21(5):981–998, Dec 2017. doi: 10.1007/s10596-017-9627-2.
23. O. Møyner. Nonlinear solver for three-phase transport problems based on approximate trust regions. *Comput. Geosci.*, 21(5-6):999–1021, 2017. doi: 10.1007/s10596-017-9660-1.
24. B. Müller, F. Kummer, and M. Oberlack. Highly accurate surface and volume integration on implicit domains by means of moment-fitting. *Int. J. Numer. Meth. Eng.*, 96 (8):512–528, 2013. doi: 10.1002/nme.4569.
25. J. R. Natvig and K.-A. Lie. 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. doi: 10.1016/j.jcp.2008.08.024.
26. J. R. Natvig, K.-A. Lie, B. Eikemo, and I. Berre. An efficient discontinuous Galerkin method for advective transport in porous media. *Adv. Water Resour.*, 30(12):2424–2438, 2007. doi: 10.1016/j.advwatres.2007.05.015.
27. OPM. The open porous media (OPM) initative. `https://opm-project.org/`, 2019.
28. A. F. Rasmussen and K.-A. Lie. Discretization of flow diagnostics on stratigraphic and unstructured grids. In *16th European Conference on the Mathematics of Oil Recovery, Catalania, Sicily, Italy*. European Association of Geoscientits and Engineers, 2014.
29. M. Shahvali and H. A. Tchelepi. Efficient coupling for non-linear multiphase flow with strong gravity. In *SPE Reservoir Simulation Symposium, 18-20 February, The Woodlands, Texas, USA*. Society of Petroleum Engineers, February 2013. doi: 10.2118/163659-MS.
30. S. M. Sheth and R. M. Younis. Localized solvers for general full-resolution implicit reservoir simulation. In *SPE Reservoir Simulation Conference*. Society of Petroleum Engineers, 2017. doi: 10.2118/182691-MS.
31. The Open Porous Media (OPM) Initative. The Norne dataset. `https://github.com/OPM/opm-data`, 2015.
32. J. A. Trangenstein and J. B. Bell. Mathematical structure of the black-oil model for petroleum reservoir simulation. *SIAM J. Appl. Math.*, 49(3):749–783, 1989. doi: 10.1137/0149044.
33. U. Trottenberg, C. W. Oosterlee, and A. Schuller. *Multigrid*. Academic press, 2000.
34. J. Watts. A compositional formulation of the pressure and saturation equations. *SPE Reservoir Eng.*, 1(03): 243–252, 1986. doi: 10.2118/12244-PA.