Final version is available at Springer: http://dx.doi.org/10.1007/s10596-017-9660-1

Nonlinear Solver for Three-phase Transport Problems Based on Approximate Trust Regions

Olav Møyner

Received: date / Accepted: date

Abstract

Implicit transport solvers used in reservoir simulation can take longer time steps than explicit solvers, but for long time steps the commonly used Newton-Raphson's method will often fail to converge. The convergence issues may manifest themselves as oscillating residuals even though the implicit discretization itself is stable. This behavior occurs because the fractional flow-type flux functions often change between convex and concave during long time steps, resulting in multiple contraction regions for the Newton-Raphson solver. The common strategy to overcome this is to set limits on the saturation changes during the nonlinear iteration, but such a limit has to be determined on a case by case basis, excess iterations may be required and practical convergence is not guaranteed for a given problem. Previous work on this problem by multiple authors has resulted in solvers based on trust regions, where unconditional convergence can be obtained for incompressible two-phase flow provided a priori analytical knowledge of the flux function exists. The goal of our work is to extend this methodology to a solver where inflection points demarking the different contraction regions does not need to be explicitly known. Instead, these values are estimated during the solution process, giving improved convergence by a local computation for each interface in the simulation model. By systematically reducing updates over regions known to produce convergence issues, it is possible to greatly reduce the computational expense, making the same formulation suitable for an arbitrary number of components. We present a series of numerical results, including arbitrary time-step lengths for two and three-phase gravity segregation, as well as three-dimensional gas and water injection problems with wells and a mixture of both viscous and gravity-dominated flow regimes. The test cases are a systematic validation on a wide variety of both analytical and tabulated relative permeability curves.

1 Introduction

Simulation of flow in porous media is an essential tool in a wide variety of fields, ranging from the management of petroleum assets and enhanced oil recovery to CO_2 sequestration and geothermal energy. The advent of modern computers has enabled practitioners to model a large number of complex physical effects in highly detailed models with grids that finely resolve the simulation domain. The resulting model equations constitute large systems of highly nonlinear conservation laws, which, when discretized, often suffer from slow convergence and severe limitations on the time steps.

A common approach has been to exploit knowledge of the underlying physics to create better solvers for porous media flow. The first step in some approaches is to split a set of coupled conservation equations governed by Darcy's law into an elliptic or parabolic pressure equation along with a set of hyperbolic or parabolic transport equations. This may be done to solve each equation separately, keeping certain quantities fixed [39, 33], or to create tailored linear solvers such as the constrained pressure residual (CPR) two-step preconditioner [37]. By treating the different equations according to their distinct nature, greatly improved performance can be achieved.

The pressure equation is generally considered to be computationally expensive due to the (semi-)global nature of the pressure field, but the *convergence rates* are usually quadratic in the absence of phase transitions. Advanced solvers for elliptic equations have thus focused on efficient ways of systematically reducing the residual [14, 34, 9] or producing approximate pressures that retain certain key properties, such as the overall trend of the pressure and a conservative velocity field [12, 16, 24].

The transport problems, owing to the hyperbolic nature of the underlying equations, are often considered to be less computationally expensive to solve, as temporal changes in the solutions for these equations are spatially localized near fronts and source terms. However, convergence rates can be poor even for seemingly simple problems and as such the transport problem is a major source of time-step restrictions and reduction during simulation. Traditionally, explicit methods have seen widespread use for the transport equations, but as modern reservoir models have a large number of cells with greatly varying flow throughput and strong nonlinear coupling, the restrictions posed by the Courant–Friedrichs–Lewy (CFL) condition have led to the popularity of implicit methods. The two main choices are the fully implicit and sequential-implicit methods, which may be unconditionally stable, but in practice the convergence can fail for longer time steps for even relatively well behaved problems. Different heuristic techniques are therefore used to select time steps and limit solution updates to reasonable values, but these strategies will often fail in practice or be too conservative for practical purposes.

The localized nature of the equations has resulted in a wide variety of approaches where equation locality and other characteristics are exploited. Some examples include solvers based on potential ordering, which use topological traversal of the flux graph to localize and reduce the size of the nonlinear systems [25, 17, 23], approaches based on operator splitting to distinguish between flow due to viscous forces and gravity [7, 8, 22], localization techniques for implicit schemes [40, 28] and streamline methods [2, 32] as well as alternate discretizations that make the residual functions smoother and thus better suited for Newton's method [19, 18, 10].

Convergence difficulties for the transport problem were attributed to the nature of the flux functions by Jenny et al. [13], who argue that unconditional convergence for two-phase viscous incompressible flow can be achieved by limiting the saturation updates based on inflection points in the flux function. The same principle was applied to compositional problems with phase transitions in Voskov et al. [35], Voskov and Tchelepi [36]. Wang and Tchelepi [38] further developed what was termed a trust-region solver by considering saturation-dependent potential changes due to gravity and capillary forces. Further works by Li and Tchelepi [20, 21] demonstrated that the numerical flux function differs from the continuous form of the functions previously analyzed, which improved the convergence rates for problems with changes in upwind directions. In this work, we develop a solver that considers the changes in the flux function along the Newton solution path in order to apply the trust-region methodology to three-phase problems, as well as non-smooth relative permeability curves. In addition, we present an approach for localizing the updates so that problems with multiple fronts and different flow regions can be solved simultaneously without being limited by each other.

2 Model equations

2.1 Conservation equations

To study the behavior of the proposed nonlinear solver, we consider the conservation equations for m different phases, where we assume that both the fluid phases and the rock surrounding the pores are incompressible. The continuity equation for a given phase α is then

$$\phi \frac{\partial S_\alpha}{\partial t} + \nabla \cdot \vec{v}_\alpha = q_\alpha, \quad \alpha \in \{1, ..., m\}, \tag{1}$$

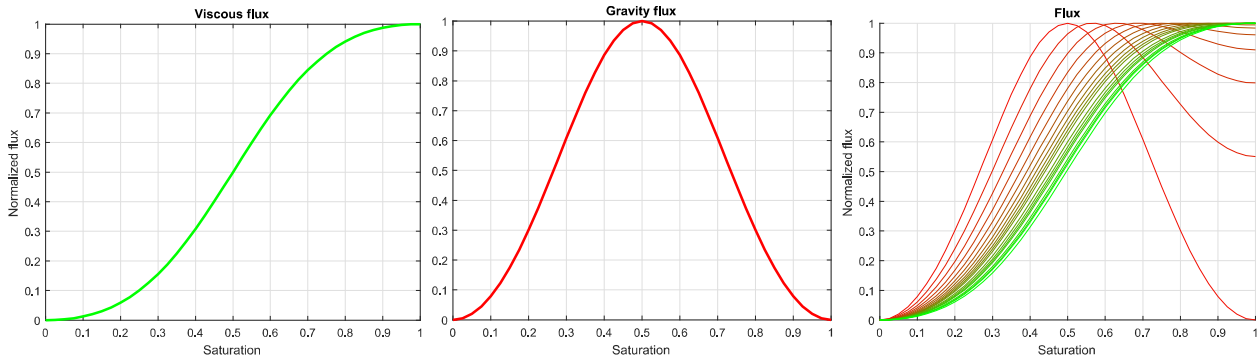
where S_α is the phase saturation, ϕ the porosity and v_α is the Darcy velocity,

$$\vec{v}_\alpha = -\mathbf{K} \frac{k_{r\alpha}}{\mu_\alpha} (\nabla p - \rho_\alpha \nabla z \cdot \vec{g}) = -\mathbf{K} \lambda_\alpha (\nabla p - \rho_\alpha \nabla z \cdot \vec{g}). \tag{2}$$

The form of the Darcy velocity given here implies no capillary pressure and as such it is a balance between the viscous forces and the buoyancy term due to gravity.

2.2 Sequential formulation

For simplicity, we will study a sequential scheme based on a splitting of the equations into a pressure and a transport part. This formulation is convenient in part because it isolates the local changes in the transported variables from the near-global changes in pressures, and thereby makes it possible to analyze sources of



(a) Viscous flux (b) Gravity flux (c) Combined flux

Figure 1: The flux function is a balance between viscous and gravity forces, shown in green and red respectively. Based on the magnitude of the total velocity, the resulting flux function can take on the shape of either gravity or viscous flux, or something in between.

nonlinearity separately. We form a linear pressure equation from (1) by taking the sum over all phases and use the closure relation $\sum_\alpha S_\alpha = 1$ to obtain

$$\nabla \cdot \vec{v}_T = q_T, \quad \vec{v}_T = \sum_{\beta=1}^{m} \vec{v}_\beta, \quad q_T = \sum_{\beta=1}^{m} q_\beta. \tag{3}$$

This elliptic equation is solved for the pressure to obtain total velocities $\vec{v}_T$ and source terms q_T. By keeping the total velocity term fixed during transport, we obtain the standard fractional-flow formulation commonly seen in literature,

$$\vec{v}_\alpha = f_\alpha(\vec{v}_T + \mathbf{K}\sum_{\beta=1}^{m} \lambda_\beta(\rho_\alpha - \rho_\beta)g\nabla z). \tag{4}$$

The total velocity $\vec{v}_T$ is only present in the first term of (4), and from this we see that the behavior of the flux function itself depends on the magnitude of $\vec{v}_T$ relative to the gravity term. When $\vec{v}_T$ goes towards zero, the flow is dominated by gravitational forces. Figure 1 demonstrates the transition between the different flow regimes for a flux function with quadratic relative permeability. The fractional flow for a given phase is the ratio of the mobility of the phase to the total mobility which both change during the transport step,

$$f_\alpha(S_1, ..., S_m) = \frac{\lambda_\alpha(S_\alpha)}{\sum_{\beta=1}^{m} \lambda_\beta(S_\beta)}, \qquad \vec{v}_T = \sum_\beta \vec{v}_\beta. \tag{5}$$

If we now discretize (1) using a standard first-order finite-volume scheme (omitting the subscript denoting a specific phase) and thus transform the velocity into a discrete flux, we obtain an implicit residual equation for each cell i that depends on the values in the cell itself and the set of neighboring cells $N(i)$,

$$R_i = \frac{\Phi}{\Delta t}(S_i^{n+1} - S_i^n) + \sum_{j \in N(i)} F_{ij}(S_i^{n+1}, S_j^{n+1}) - q_i = 0 \quad \forall \quad i \in \{1, ..., n_c\}. \tag{6}$$

The discrete fluxes use the standard two-point transmissibility approximation to the flux at the interface between a cell pair ij,

$$F_{ij}(S_i, S_j) = f_\alpha \left(V_{ij} + T_{ij} \sum_{\beta=1}^{m} \lambda_\beta(\rho_\alpha - \rho_\beta)g\Delta z_{ij} \right). \tag{7}$$

The total velocity V_{ij} is defined at the interfaces by solving the pressure equation. The phase mobilities used in the fractional-flow and gravity fluxes are upwinded based on the sorting of phase fluxes as described by Brenier and Jaffré [3]. Δz_{ij} is the depth difference between cells i and j.

2.3 Nonlinearity of the flux function

Solving the transport equations usually amounts to solving (6) for all cells and $m-1$ phases. We define residual equations for each phase and require that these equations are solved to some tolerance ϵ,

$$\sum_{i=1}^{n_c} |R_i| \leq \epsilon_t. \tag{8}$$

To solve the nonlinear equations that appear by the implicit discretization of (6), the common choice is to employ Newton's method to find the root. Newton's method applied to classical test problems gives quadratic rate of convergence, but in practice the reduction in residuals can be slow or stagnate completely as the solver jumps between successive iterates in a loop. It matters little that the implicit discretization is unconditionally stable, if the solver is unable to find the solution.

2.4 Motivation for trust region solvers

It has been observed that the difficulty of this nonlinear problem primarily depends on the nature of the flux functions $F(S_i, S_j)$ [13]. If (6) is to be solved by a (quasi-)Newton solver, we are dependent on the problem being either convex or concave to guarantee convergence. However, if we consider the residual for a single cell with multiple faces, we easily see that if $\sum_j F_{ij}$ is to be convex (concave) everywhere, all the individual functions should likely also be convex (concave)[1]. Unfortunately, most flux functions encountered in practice in Darcy flow are *not* uniquely convex or concave. As a consequence, the Newton solver will only converge for problems where $S^{n+1} \approx S^n$ or when the path taken by the solver from S^n to S^{n+1} does not cross any inflection points or discontinuities.

To motivate the development of our nonlinear solver, we will examine the convergence criterion for the Newton solver for successive iterates. Assume that we have a scalar problem $R(x) = 0$, where the residual function is twice differentiable and has a unique solution. Let $G(x)$: $\mathbb{R} \to \mathbb{R}$ be one step of Newton's method applied to $R(x)$,

$$G(x) = x - \frac{R(x)}{R'(x)}. \tag{9}$$

In order for Newton's method to converge unconditionally, we require that for two arbitrary guesses x, y we have the strict inequality

$$\|G(x) - G(y)\| < \|x - y\|. \tag{10}$$

This is simply stating that Newton's method should be contractive for all points in our solution space so that applying Newton's method brings the solution closer to convergence. We apply the mean value theorem to the Newton operator to obtain that for some point ξ we must have,

$$\|G(x) - G(y)\| = \|G'(\xi)(x - y)\| \leq \|G'(\xi)\|\|x - y\|, \quad \xi \in [x, y]. \tag{11}$$

Inserting our definition (9) and rearranging, we obtain the convergence ratio,

$$\frac{|R''(\xi)R(\xi)|}{|R'(\xi)^2|} < 1 \quad \forall\, \xi \in [x, y]. \tag{12}$$

If we set $x = S^n$ and $y = S^{n+1}$, we see that Newton's method is not guaranteed to converge if R' is zero for some S between S^n and S^{n+1} as (12) becomes unbounded. If we now consider our residual equation (6), we can consider the second-order derivative for each term separately. The accumulation term is linear with respect to the cell saturation, and thus has constant first-order derivatives for $\Delta t \neq 0$, while the second derivatives always vanish,

$$\frac{\partial}{\partial S_j^{n+1}} \left(\frac{\Phi}{\Delta t}(S_i^{n+1} - S_i^n) \right) = \frac{\Phi}{\Delta t}\delta_{ij}, \qquad \frac{\partial^2}{\partial (S_j^{n+1})^2} \left(\frac{\Phi}{\Delta t}(S_i^{n+1} - S_i^n) \right) = 0 \,\forall\, j.$$

From this, we see that (12) vanishes for the accumulation term by itself regardless of time-step length. The remainder of (6) corresponding to the flux term is then the main source of nonlinearity as the derivative of the residual function is proportional to the sum of the second derivatives of the flux functions. Clearly, we must treat discontinuities and inflection points in (7) with great care if we are to obtain convergence.

[1] Recall that a sum of functions will be convex (concave) if the individual functions are convex (concave).

2.5 Relative permeability

As we have seen, the flux functions are the main source of difficult nonlinearities for Newton's method, and the discrete flux functions themselves are highly dependent on the mobility of the different phases. The mobility depends on the relative permeability curves, which characterize the ability of a fluid phase to flow in the presence of another phase in a specific type of porous media. The choice of relative permeability curves is an essential tool when modeling flow in porous media and can be used to embed analytical knowledge, experimental data, adjustments from history matching, and the effect of upscaling. With such a wide variety of sources, the relative permeabilities can vary greatly between phases and different simulation models. A robust nonlinear solver must be able to handle general relative permeability functions, regardless of their origins.

2.5.1 Three-phase relative permeability

For a three-phase system, we define relative permeability for the water and gas phases as functions $k_{rw}(S_w)$ and $k_{ro}(S_o)$ of the corresponding phase saturations. For the oil-phase, we define relative permeability for two-phase states with oil and water, $k_{row}(S_o)$, and oil and gas, $k_{rog}(S_o)$, respectively. There exist a number of different possible choices for evaluating the three-phase relative permeability when all phases are present, see for example the works by Stone [29, 30] and Baker [1]. The choice can impact the convergence rates of the underlying scheme, see e.g. discussion in Lee and Efendiev [18]. For this work, we follow the default setting of a commercial reservoir simulator [27], where the oil relative permeability is weighted according to the gas and water saturations,

$$k_{ro}(S_w, S_o, S_g) = k_{row}(S_o)w_w(S_w, S_g) + k_{rog}(S_o)w_g(S_w, S_g) \tag{13}$$

$$w_w(S_w, S_g) = \frac{S_w - S_{wc}}{S_g + S_w - S_{wc}}, \qquad w_g(S_w, S_g) = 1 - w_w(S_w, S_g). \tag{14}$$

This choice ensures positive values and continuous derivatives provided that k_{rog}, k_{row} fulfill the same criterion and that $S_{wc} < S_w$ where S_{wc} is the connate water saturation. In principle, however, the proposed solver can use any parametrization on the form $k_{ro}(S_w, S_o, S_g)$, as the exact expression of the relative permeability takes no part in the algorithm. The solver acts directly on the simulator flux function and will determine inflection points on the compound expression for the flux, not relative mobility or buoyancy related terms directly.

2.5.2 Relative permeability systems used in numerical examples

The first class of relative permeabilities we consider in this paper consists of analytical relative permeabilities given by a closed-form expression. We consider simple relative permeabilities given by Corey functions for a pair of wetting and non-wetting fluids,

$$k_{rw}(S_w) = S_w^n, \quad k_{rg}(S_g) = S_g^n, \quad k_{row}(S_o) = k_{rog}(S_o) = S_o^n \tag{15}$$

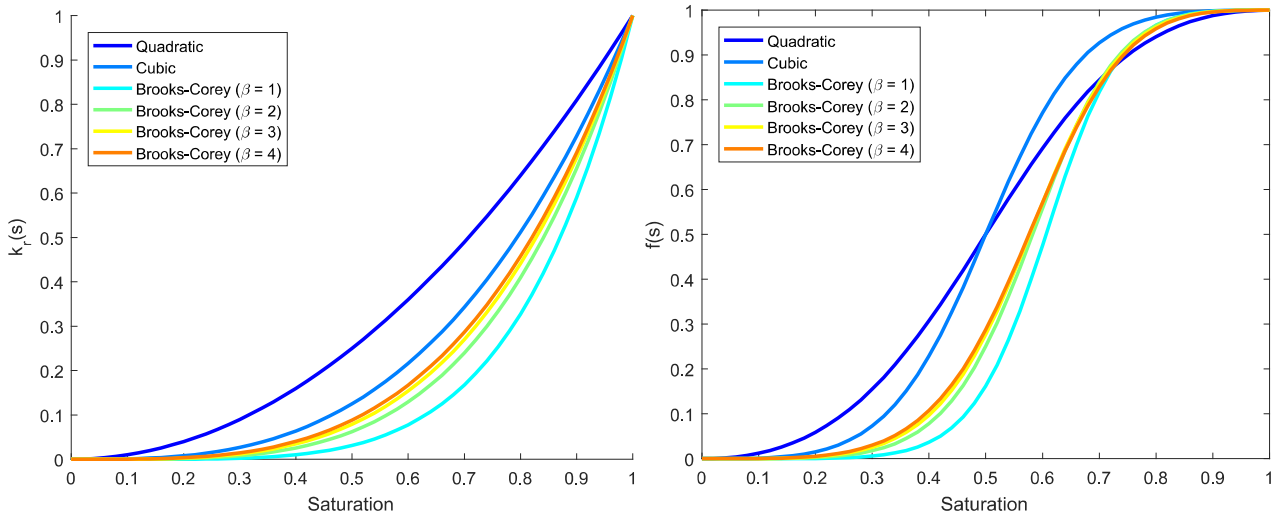
where $n \in \{2, 3\}$ in this paper.

We will also consider the form given by Brooks and Corey [4],

$$k_{rw}(S_w) = S_w^{3+2/\beta}, \quad k_{row}(S_o) = k_{rog}(S_o) = (S_o)^2 \left(1 - (1 - S_o)^{1+2/\beta}\right), \tag{16}$$

with $\beta \in \{1, 2, 3, 4\}$. Note that we have neglected endpoint scaling and residual saturation for these functions to be consistent with the test case presented by Li and Tchelepi [20]. The six analytical relative permeability functions under consideration are shown in Figure 2 along with the corresponding fractional-flow functions.

The second class of relative permeabilities to be considered consist of piecewise linear functions. While analytical expressions may be easier to work with from a mathematical standpoint, most, if not all, commercial simulators use piecewise linear functions internally. This format can approximate both analytical functions and input functions based on experimental or upscaled data. While convenient for practitioners, piecewise linear functions are more challenging than analytical functions for the purposes of designing advanced nonlinear solvers. In particular, the analysis required to determine inflection points is not well suited

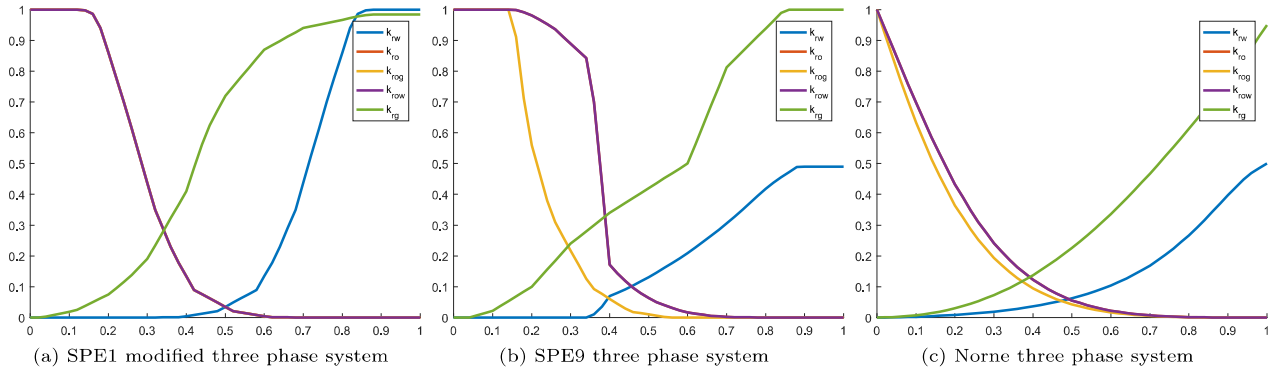


(a) Relative permeabilities (b) Fractional flow for two phase flow

Figure 2: The six different analytical relative permeability curves (left) with their corresponding fractional flow curves (right).

when the second-order derivatives are always zero along the line segments or ill-defined at the breakpoints between segments.

We consider three different relative permeability curves given as piecewise linear functions. The first is a modified version of the relative permeability system presented by Odeh [26]. A synthetic test problem for gas injection, the water considered was immobile for all saturations values. Here, we assign the water phase the same curve as the oil relative permeability to ensure that all three phases flow. The tables are otherwise unmodified and the resulting functions can be seen plotted in Figure 3a. The second set of functions, shown in Figure 3b are taken from the Ninth SPE Comparative Solution Project [15] which, in turn, were taken from a real field study. The ninth and final relative permeability model is taken from the simulation model of the Norne field, which has been released to the public [11, 31]. In the Norne model, the relative permeabilities vary with rock type and herein we have chosen to use that of the first rock type without endpoint scaling, as shown in Figure 3c. We emphasize that the relative permeability curves are used as-is. As the goal of the paper is to develop a black-box approach, no information about breakpoints or analytical nature of the curves is passed onto the nonlinear solvers themselves.



(a) SPE1 modified three phase system (b) SPE9 three phase system (c) Norne three phase system

Figure 3: The three different piecewise linear relative permeability functions used for the numerical examples.

2.6 Convergence criterion

To ensure reproducibility of our results, we will be explicit about the convergence criterion used for the solver. If we rescale the residual equation for a single cell (6) by the ratio between the time-step length and the cell pore volume we obtain,

$$\bar{R}_i = (S_i^{n+1} - S_i^n) + \frac{\Delta t}{\Phi_i}\left(\sum_j F_{ij}(S_i, S_j) - q_i\right) = 0 \quad \forall \quad i \in \{1, ..., n_c\}. \tag{17}$$

This scaling represents a saturation error, which means that $|\bar{R}\|$ is scaled between zero and one for all cells. From this, we can establish a convergence criterion based on both the maximum and average values,

$$e_{max} = \|\bar{\mathbf{R}}\|_\infty \leq 10^{-3}, \qquad e_{avg} = \Delta t \frac{\|\mathbf{R}\|_1}{\|\Phi\|_1} \leq 10^{-7}, \tag{18}$$

which is the same convergence criterion employed in a commercial simulator [27].

3 Trust-region solvers

3.1 The interface problem

We start by considering a single interface. Recall that the convergence criterion for Newton's method implied that the difficulty of the global problem is a result of the individual flux functions for each interface. As the total velocity is fixed from the pressure update in the sequential implicit scheme under consideration, we need only consider the dependence on transported values. Assume that we have computed the Newton update for the saturation vector in the standard manner, i.e.,

$$\Delta \mathbf{S} = -\mathbf{J}^{-1}(\mathbf{S})\mathbf{R}(\mathbf{S}), \tag{19}$$

where $\mathbf{J}$ is the Jacobian matrix of the residual function $\mathbf{R}$. We assume that the update is limited so that the potential new saturation can be written as,

$$S_i^{n+1} = S_i^n + \Delta S_i^n, \qquad S_i^n \in [0, 1], \quad S_i^{n+1} \in [0, 1]. \tag{20}$$

The Newton update for all cells has both a direction and a magnitude. We seek to modify the magnitude with a factor $\omega \in [0, 1]$ so that the line from $F(\mathbf{S}^n) \to F(\mathbf{S}^{n+1}) = F(\mathbf{S}^n + \omega\Delta\mathbf{S})$ does not cross beyond any inflection lines or discontinuities. Since we only intend to modify the magnitude of the update, we can define a one-dimensional space of points along the Newton path, which is the direction in the solution space where the next Newton update is headed,

$$\mathbf{S}_\omega = \mathbf{S}^n + \omega\Delta\mathbf{S}^n, \qquad \omega \in [0, 1]. \tag{21}$$

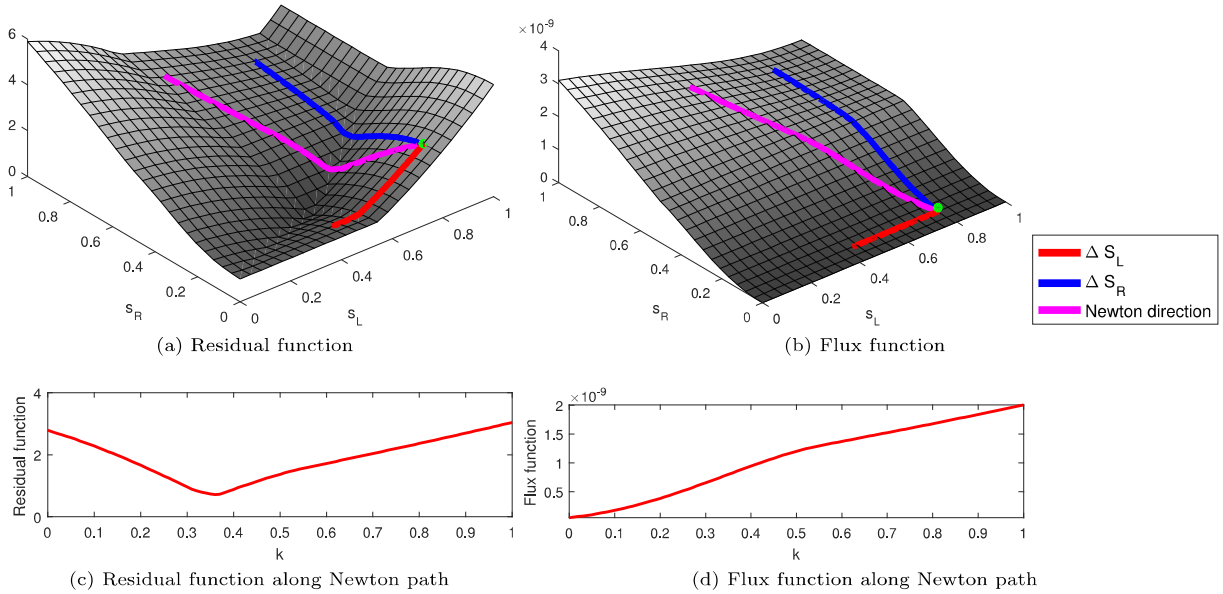
We introduce the directional derivative of F_{ij} for a given interface,

$$\nabla_d F_{ij} = \nabla F_{ij} \cdot \mathbf{d} = \frac{1}{\|\Delta\mathbf{S}^{ij}\|} \sum_{k=1}^{2n_v} \frac{\partial F_{ij}}{\partial \mathbf{S}_k^{ij}} \Delta S_k^{ij} \qquad \mathbf{d} = \frac{\Delta\mathbf{S}^{ij}}{\|\Delta\mathbf{S}^{ij}\|}, \tag{22}$$

where we have defined $\Delta\mathbf{S}^{ij}$ as the vector of updates to the n_v cell solution variables on either side of the interface,

$$\Delta\mathbf{S}^{ij} = \left[\Delta S_{1i}, ..., \Delta S_{n_v i}, \Delta S_{1j}, ..., \Delta S_{n_v j}\right]^T. \tag{23}$$

The gradient of F could in principle depend on n_v solution variables on each side of the interface. By projecting the update onto on the directional derivative, we ensure that the problem is one-dimensional for general problems. Figure 4 illustrates how the Newton update defines a one-dimensional subspace of the residual and flux functions for a problem with two phases, two cells, and a single interface. Hereafter, we shall, for simplicity, refer to the directional derivative evaluated at a point ω along the Newton direction



(a) Residual function (b) Flux function

(c) Residual function along Newton path (d) Flux function along Newton path

Figure 4: The convergence surface for a two-cell problem with gravity (left) and the flux function for the interface between the cells (right). The saturations in the cells, denoted S_L and S_R, define the two dimensional solution space. The Newton update is shown in pink, the initial guess shown as a green dot, and is made up of an update to the saturations in the left cell (shown in red) and the right cell (shown in blue). By defining a one-dimensional space parametrized tangential to the Newton path, we can easily view the flux and convergence functions as they change during the update (bottom).

using the shorthand $\nabla_d F_{ij}(\mathbf{S}_\omega) = F'(\omega)$ and regard the Newton direction as fixed for a given step. Any use of $F'(\omega)$ refers to a specific interface and so we neglect the ij subscript.

Inflection points for a given interface are determined by the second-order derivative of the flux function along the Newton path in solution space, with similar notation as for the the first-order directional derivative as $\nabla_d^2 F_{ij}(\mathbf{S}_\omega) = F''(\omega)$. This second-order directional derivative requires the computation of the full Hessian matrix for the flux function. Computing the full Hessian is expensive and ill-defined when the underlying data is non-smooth, for instance when the relative permeability functions are given as piecewise linear functions as noted earlier. In the following, we assume that we have some approximation $F''_*(S_\omega) \approx F''(S_\omega)$ that is continuous and easy to evaluate, the details of which are given in the following sections.

3.2 The flux-search algorithm

Once the problem has been parametrized by a single value ω, we can design an algorithm that determines optimal relaxation factors. Define a sequence of updates along the line, $\omega_0, \omega_1, ..., \omega_{n_s}$. We assume that $\omega_0 = 0$ and $\omega_{n_s} = 1$. We require that the sample count $n_s \geq 2$ so that there is at least one intermediate point between the current state and the tentative new Newton update. We now have a series of intervals and should attempt to determine the final relaxation factor ω corresponding to a safe update. By keeping the convergence criterion for Newton's method in mind, we observe that a safe update would be one that does not extend the new state far beyond any discontinuities or inflection points in the flux function. Ideally, an update should end just on the other side of the such points in order for the next update to be valid.

To obtain a safe update, we apply a search strategy along the update path. Initially, we only have two endpoints: ω_0 was used for evaluation of the Jacobian, and ω_{n_s} which will be used in the next Jacobian if no relaxation is needed. Recall that we defined the list of saturation updates in the different cells for a given interface ij as $\Delta\mathbf{S}^{ij}$ in (23). The norm of the update for a given interface is $\|\Delta\mathbf{S}_{ij}\|_\infty$. Note that an interface can be either a connection between two cells, or the connection between the wellbore and the cell the well is

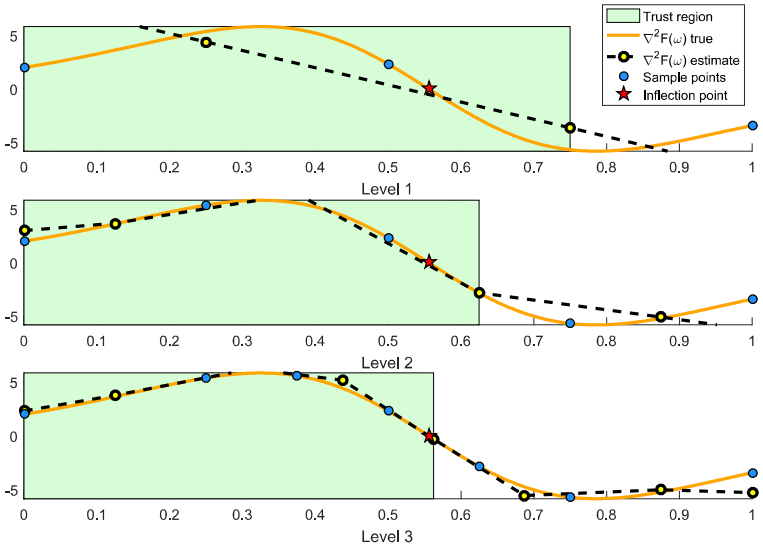


Figure 5: Three levels of sampling for an S-shaped flux function where the inflection point is the limiting factor for the Newton update (shown as a red star).

perforated in. We divide the update into n_s initial subintervals,

$$n_s = \max\left(\|\Delta \mathbf{S}^{ij}\|_\infty / \delta_S, 2\right). \tag{24}$$

We have used δ_S to ensure that more intervals are added when the update is very large and may span many discontinuities and inflection lines. All examples considered herein use $\delta_S = 0.2$, giving two initial subintervals for most interfaces and time steps and up to five intervals for the largest updates. We then define the vector of points in the unit interval to be sampled,

$$\mathbf{w} = [0, h_w, 2h_w, ..., 1]^T, \qquad h_w = \frac{1}{n_s - 1}. \tag{25}$$

Once the initial (sub)intervals have been found, we can apply the `fluxsearch` routine for each phase as detailed in Algorithm 1. For detailed descriptions of the subroutines used in the algorithm, we refer to the Appendix. The algorithm is a modified bisection method, which finds the relaxation factor based on a small number of function evaluations. We note that it would be possible to choose the evaluated points in order to minimize the error in the interpolated flux function based on error estimates for spline fitting. In this work, however, we use the same set of evaluations to determine both inflection points and the discrete upwind changes for which there is no known tailored sampling strategy. By considering different indicators where the flux function has either inflection points or kinks separately, it is possible to find the optimal step length for each interface. For immiscible flow, the two indicators used for each phase are the changes in upwind direction for either phase and changes in sign for the second-order derivative of the flux function.

For changes in upwind direction, we define the indicator $G = [u_w, u_o, u_g]$, which for a given interface and saturation update assigns unique values for each combination of upwind directions for the different phases. For instance, the indicator for a interface where water and oil is co-current and the gas is counter-current would be $G = [1, 1, 0]$, which will change to $G = [1, 0, 0]$ if the oil-phase becomes counter-current. This means that we only allow for a single change in phase upwind for any given interface during a nonlinear iteration. Discrete indicators are easily evaluated and, although the examples here are limited to upwind changes due to gravity, this approach has a natural extension to problems with phase changes along the Newton path. When we consider the second-order derivative of the flux function, we use an approximation based on interpolation. Since we have both F and F' evaluated at all the sample points, we could have used a high-order approach such as Hermitian polynomials or cubic splines for F. However, since F'' is not always well defined for piecewise linear relative permeabilities, and because the flux functions can have large regions with zero values, we will instead use a monotone interpolation scheme for F'. The interpolation scheme is a monotone cubic spline

due to Fritsch and Carlson [6], where we interpolate F' directly and take the derivatives. Monotone splines have the practical property that they avoid oscillations and overshoots near discontinuities and handle flat regions without any problems, while still being in C^1. The Fritsch-Carlson interpolation scheme is remarkably robust, in that it can treat both piecewise linear and differentiable functions with second-order accuracy in monotone regions.

The algorithm is visualized for three levels in Figure 5, applied to a periodic function with an inflection point at $x = 0.55$. In each step, the true value of F'' is plotted in orange, with the estimated F''_* shown as a dashed line. As the sign-change in F'' is estimated to lie between 0.25 and 0.75, the second level adds additional sample points in the vicinity, improving the estimate of F''. Finally, the third level gives a final trust region shown in green, which is very close to the true inflection point, plotted as a red star. More levels can be added to more accurately determine the trust region, but for all the examples considered herein, we use a maximum of three levels. Adding a third level does not significantly impact the iteration counts for the numerical examples based on gravity segregation. As the objective of this paper is to discuss unconditional convergence, we err on the side of caution and add an additional level.

Algorithm 1 The `fluxsearch` algorithm determines the suggested relaxation factor for a given phase flux function. The subroutines for the different relaxation factors are explained in the Appendix.

procedure FLUXSEARCH($\omega, \mathbf{F}, \mathbf{F}', \mathbf{G}, \mathbf{S}, \Delta\mathbf{S}, \mathbf{w}_{next}, l$)
  $\omega^G, \mathbf{w}^G_{next} \leftarrow$ `computeGravityRelaxation`$(\mathbf{G}, \mathbf{w})$
  $\omega^V, \mathbf{w}^V_{next} \leftarrow$ `computeViscousRelaxation`$(\mathbf{F}, \mathbf{F}', \mathbf{w})$
  if $l = l_{\max}$ **then** ▷ We are at the max level
    $\omega_f \leftarrow \min(\omega^G, \omega^V)$
  else
    $\mathbf{F}, \mathbf{F}', \mathbf{G} \leftarrow$ `evaluatePhaseFlux`$(\mathbf{S}, \Delta\mathbf{S}, \mathbf{w}_{next})$ ▷ Update with additional fluxes
    if $\omega^G < \omega^V$ **then**
      $\omega_f \leftarrow$ `fluxsearch`$(\omega, \mathbf{F}, \mathbf{F}', \mathbf{G}, \mathbf{S}, \Delta\mathbf{S}, \mathbf{w}^G_{next}, l+1)$ ▷ Limited by upwind indicator
    else
      $\omega_f \leftarrow$ `fluxsearch`$(\omega, \mathbf{F}, \mathbf{F}', \mathbf{G}, \mathbf{S}, \Delta\mathbf{S}, \mathbf{w}^V_{next}, l+1)$ ▷ Limited by viscous indicator
    end if
  end if
  return ω_f
end procedure

3.3 Local and global chopping

Once we have obtained relaxation factors ω_{ij} for each interface ij, we can update the solution variables with the appropriate increments. If we want the update to follow the Newton-path and not modify the direction, we should limit the update by the smallest relaxation factor present,

$$\mathbf{S}^{n+1} = \mathbf{S}^n + \min_{ij}(\omega_{ij})\Delta\mathbf{S}^n. \tag{26}$$

This is the global chopping used in the existing literature on trust-region solvers and is the only approach guaranteed to not modify the direction of the update. Using a single relaxation for all cells in the domain may lead to unconditional convergence, but it may also be overly conservative for larger models where multiple fronts are present. The transport equation (1) is a hyperbolic partial differential equation, a class of equations that are characterized by finite speed of information propagation. For example, a problem with multiple injector wells will have multiple fronts propagating from the different wells, and these fronts will not interact until a late stage in the simulation. Likewise, the updates in the Newton solver itself may converge faster in certain regions than in others. The unidirectional propagation of Newton updates along flow paths has previously been used to create nonlinear solvers that use topological sorting to decouple the global Newton update into a sequence of local updates [25]. To exploit this locality in the saturation updates, we devise a strategy that computes the relaxation factor locally so that the update is on the form

$$\mathbf{S}^{n+1} = \mathbf{S}^n + \mathbf{w}^T_{opt}\Delta\mathbf{S}^n, \tag{27}$$

where $\mathbf{w}_{opt}$ is some vector of cellwise relaxation factors. Assume that we have computed relaxation factors for all interfaces in a given Newton update. A natural starting point is to define a connectivity matrix $C_{\Delta s}$ which connects cells that are strongly dependent on each other,

$$(C_{\Delta s})_{ij} = \begin{cases} 1, & \text{if } |\mathbf{J}_{ij}^k \Delta \mathbf{S}_j^k| \geq \epsilon |\mathbf{J}_{ii} \Delta \mathbf{S}_i^k| \text{ for any primary variable } S_k, \\ 0, & \text{otherwise.} \end{cases} \tag{28}$$

The connection matrix uses the scaled Jacobians (17) of the $m-1$ equations that were solved to obtain the saturation updates. The connection from cell i to j is strong if the saturation update in cell j has a relative impact on the saturation error in cell i which is larger than some threshold ϵ. A natural choice for ϵ here is to set it equal to e_{max}, so that we neglect any changes having impacts that are smaller in magnitude than the convergence criterion.

For purely viscous problems discretized with a linear two-point scheme for the flux, the graph of (28) is a directed graph with no cycles, as the flux between two cells only depends on the upstream cell because of the upwinding of mobility. If the interface has flux due to gravity, the connection will form a cycle between the cells, as the gravity flux is proportional to the density differences between the two cells. The process for finding the relaxation factor is as follows:

1. Find all (if any) cycles in (28) and combine the nodes in each cycle into a single node.
2. Assign a tentative relaxation factor to each cell. Cells are assigned the minimum relaxation value from all interfaces connected to that cell having nonzero flux. Cells belonging to a cycle are then assigned the minimum value for all cells in that cycle.
3. Perform a topological sort of the modified connection matrix where cycles have been removed.
4. Traverse the graph so that each cell is assigned the minimum relaxation of itself and all cells that are upstream to it.

After the traversal, all cells have a relaxation factor that is bounded by any cells that are upstream by the strong connections.

3.4 Computational cost

The determination of the relaxation factors require a number of additional function evaluations. In order to assess this cost, we will both report the actual number of residual evaluations and consider the theoretical complexity of the algorithms. In this section, it is natural to compare with the line-search, which also uses additional function evaluations to improve convergence rates.

The global line-search with backtracking evaluates the full residual at $n_{ls}-1$ points where n_{ls} is the number of iterations required to find a global reduction in the residual (the final evaluated residual corresponds to the next nonlinear iteration, we can subtract the cost). If we write the cost in terms of individual cell evaluations of the residual, the total cost is $n_c(n_{ls}-1)$.

The flux-search is active for the n_a interfaces which correspond to saturation updates larger than the convergence criterion. For each of these interfaces, we require at least one additional residual evaluation at the midpoint between the current and next states. If a interface does not appear to have upwind changes or inflection points, the algorithm terminates and no additional cost is required. For faces that require estimation of the flux function, $2l_{max}$ additional points are added. The number of function evaluations are shown in Table 1.

We note that the main difference between a flux-search and a line-search is that the flux-search is local to each interface. In terms of evaluations, the flux-search does not require a full evaluation of the residual as no accumulation terms, well terms or divergence is required, but it does require the derivatives with respect to the primary variables. In the following numerical examples, we assume a flux evaluation on a single interface to be equal in cost to evaluating the residual locally in two cells.

4 Numerical results

We validate the approach on a wide variety of numerical tests. Example 1 demonstrates the applicability of the solver to a two-phase gravity-segregation problem from the literature, before extending the same problem

Table 1: Comparison of different solver strategies in terms of cell-wise function evaluations.

Solver	Residual evaluations	Additional costs
Flux-search	$2n_a(1+2\gamma l_{max})$	$n_a \times l_{max}$ spline updates
Flux-search (local)	$2n_a(1+2\gamma l_{max})$	+ Topological sort
Line-search	$n_c(n_{ls}-1)$	

to three phases in Example 2. A sloped injection problem in Example 3 demonstrates the behavior of the solver on a problem with both viscous and buoyancy forces. Example 4 benchmarks the solvers on a problem with multiple wells and strongly anisotropic permeability. Finally, Example 5 is a time-step study performed on a field model from the Norwegian Sea, which contains a large number of vertical and horizontal wells, faults, eroded cells, and anisotropic permeability.

The five solvers that will be used throughout the examples are the standard, unmodified Newton solver; the modified Newton, which corresponds to a maximal saturation update ($\Delta S_{max} = 0.1$); the Newton solver with a linesearch and maximal saturation update; the flux-search solver based on global chopping; and the flux-search solver with local chopping based on topological sorting.

4.1 Example 1: A single-cell, three-phase problem

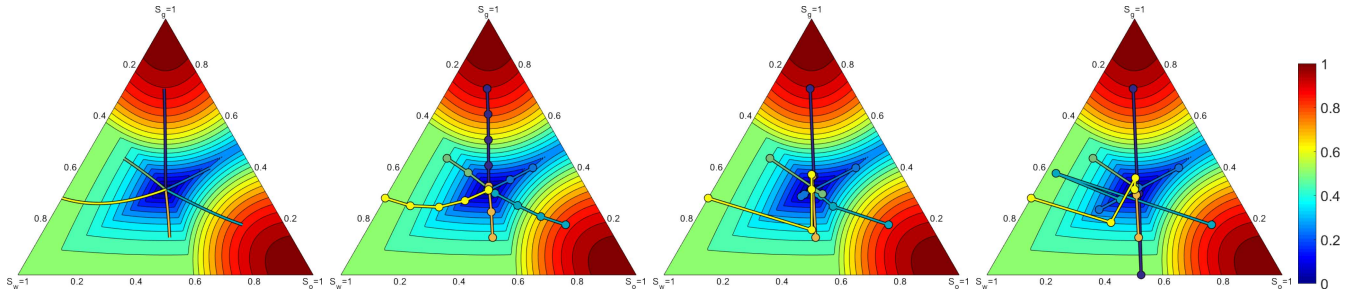
We will demonstrate the different solvers on a very simple test problem. We consider a a single cell with an injection source term and a single outflow boundary face. The injected fluid composition contains an equal volumetric mixture of oil, gas and water. The phases considered have equal viscosity and the densities are 1000, 500 and 250 kg/m^3 for water, oil and gas respectively. The three phases have quadratic relative permeabilities. The saturation distribution on the other side of the boundary condition interface is equal to the injected composition. Since Newton without saturation chopping does not converge due to the inflection lines in the flux functions, we will limit the discussion to the line search, flux search and Newton with saturation chopping. A solution with $\Delta S_{max} = 0.001$ is also included as a reference that closely approximates the true Newton path.

We first consider the problem without gravity present. For a sufficiently long time-step, the saturations in the cell should be equal to the injection composition. Figure 6 plots the normalized convergence surface for this problem together with the path taken by different nonlinear solvers for six initial guesses to the true solution at $S_w = S_o = S_g = 1/3$. We observe that the solver with $\Delta S_{max} = 0.1$ approximates the Newton path with fixed step lengths. The magnitude of the update is limited for all but the last step, giving a steady convergence rate. The flux search solver takes steps of varying step lengths, keeping the updates as large as possible without crossing inflection lines for the gas and oil flux functions. Finally, the line search with backtracking is generally similar to the flux search as it also takes variable step lengths. However, it may take steps far beyond the true solution, as only a sufficient decrease in residual is required for a step to be accepted.

The same problem is repeated with gravity in Figure 7. Note that the modified surface contains three different flow regimes: At $S_w = 1$, all phases are co-current, while on the opposite end of the diagram only water is co-current. We observe that the flux search algorithm correctly identifies these regions, giving large updates inside each region where applicable. The $\Delta S_{max} = 0.1$ solver again follows the Newton path with steady updates. The line search takes large updates initially as the required reduction in residual is easy to achieve, cutting the updates only when fairly close to the solution. Although line search converges unconditionally, this can lead to slow convergence when the initial guess has a large residual value due to a long time-step. For this reason, we combine the line search with a chopping value of $\Delta S_{max} = 0.1$ in the remaining examples.

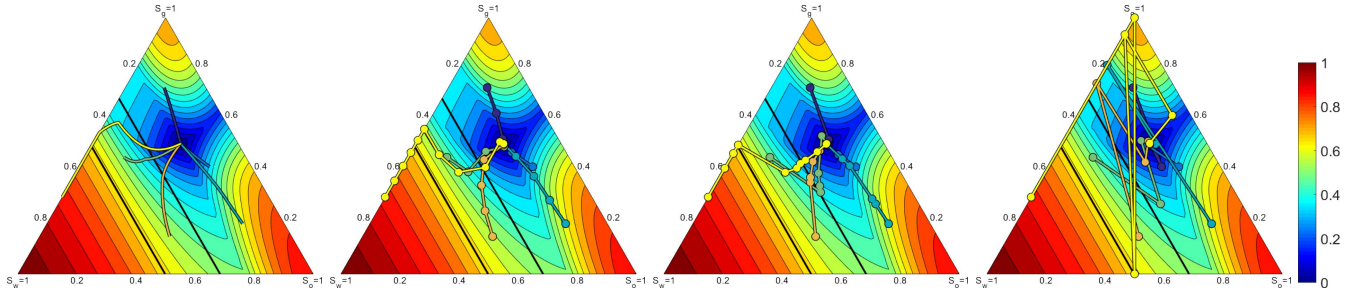
4.2 Example 2: Two-phase gravity segregation

For our second numerical example, we consider the problem of gravity segregation in a one-dimensional column. The starting point is the two-phase case posed in Li and Tchelepi [20]. We will perform a systematic test with all the nine different relative permeability curves introduced above to not only test the solver, but also estimate the nonlinearity introduced by the various fluid systems used in later examples. The standard



(a) $\Delta S_{max} = 0.001$ (b) $\Delta S_{max} = 0.1$ (c) Flux-search (d) Linesearch

Figure 6: Four different updating strategies plotted on the normalized residual surface for a three-phase single-cell problem without gravity. The lack of symmetry for the convergence surface is due to the solution of the transport equations for oil and gas only.



(a) $\Delta S_{max} = 0.001$ (b) $\Delta S_{max} = 0.1$ (c) Flux-search (d) Line-search

Figure 7: Four different updating strategies plotted on the normalized residual surface for a three-phase single-cell problem with gravity. Note that black lines correspond to changes in upwind direction for the interface.

Newton solver is compared with the global flux-search solver as well as the modified Newton solver. We do not include the local flux-search solver, as a single column with gravity will always form a strong cycle and result in a global chopping. The grid and problem can be seen in Figure 8 along with the saturations at equilibrium in Figure 9.

We define a column of height H with 80 cells. The model has homogeneous permeability and porosity, with initial water saturation of one in the top 40 cells and zero in the bottom 40. The oil phase is set to have half the density of the water phase, and both phases are assumed to be incompressible. This initial fluid distribution is unstable because of the density difference, and the phases will switch places due to buoyancy forces. We normalize the time steps by the time it takes to reach complete segregation when the relative permeabilities are linear functions of saturation $k_r(S) = S$. A time step of $\Delta t = 1$ then corresponds to the point in time when both phases have switched places in the linear case and is referred to as the characteristic time. The problem is visualized in the left part of Figure 8. The viscosities are equal for both phases.

We simulate the problem for a wide range of time steps, ranging from the very short, which are far from equilibrium after time integration ($\Delta t = 0.01$), to the very long ($\Delta t = 10000$). The results are reported in Figure 10. If we first consider the regular Newton method, we observe that it only converges consistently for small to moderate time steps. By introducing maximum saturation changes per nonlinear iteration, however, we see that the convergence is greatly improved and the method eventually converges for all time steps and relative permeability combines except for the SPE 9 fluid model. The flux-search solver converges for all time-step and relative permeability pairs. The convergence is similar to the modified Newton for short time steps, but for the truly long time steps $\Delta t \geq 50$ we see that the number of iterations is greatly reduced. The line search is comparable to the trust region solver for steps up to about 1, after which a few time-steps use more than the prescribed 500 iterations to converge. Generally, the flux-search solver has a smoother response

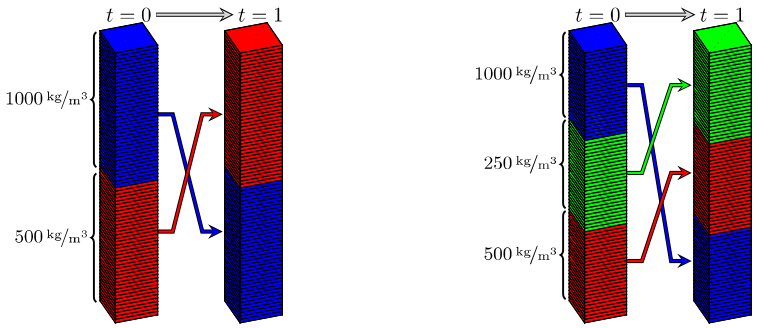
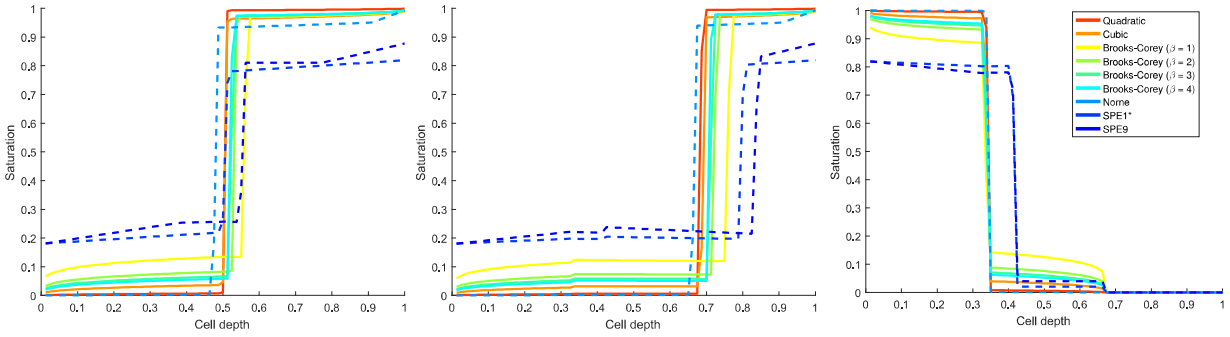


Figure 8: The two-phase and three-phase gravity segregation scenarios demonstrated. Fluids with different densities are arranged layered by density so that the system is unstable.



(a) Final water saturation, two-phase (b) Final water saturation, three-phase (c) Final gas saturation, three-phase

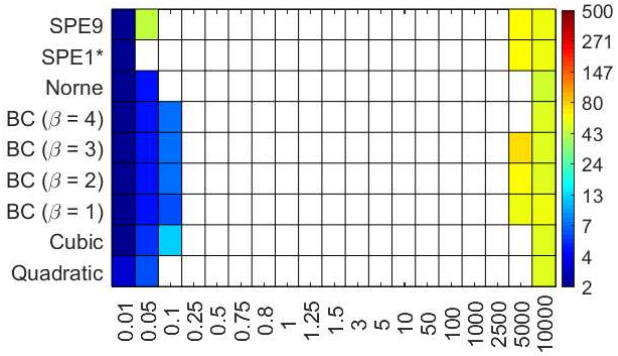
Figure 9: Saturation distributions for the two and three-phase gravity segregation problems at normalized time $t = 10^4$ for the different relative permeability curves considered. Whole lines correspond to analytical relative permeability functions, while dashed lines are given on piecewise linear form.

surface than the line-search. This is due to the systematic chopping at changes in upwind direction which will be the same number of cuts for all relative permeability curves, as all cells will switch from their initial state. For all the solvers capable of taking long time steps, the problems with analytic relative permeability curves generally converge more rapidly than with piecewise linear curves, likely due to their continuous derivatives with respect to S.

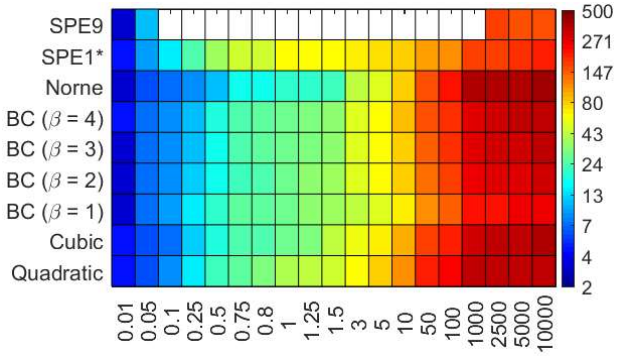
Li and Tchelepi [20] reported iteration counts for the same six analytical relative permeability curves, and their maximum observed iteration count was 123. Our solver converges in a maximum of 84 iterations, which is close to the number of cells in the problem. Even for the highly challenging SPE 9 fluid model, our new solver has no problems converging, aside from a somewhat higher number of iterations compared to the other problems with piecewise linear curves. From this, we can conclude that the approximate trust regions produced by the flux-search algorithm accurately capture inflection points and kinks in the flux function, giving unconditional convergence for this relatively simple test problem. In terms of computational cost, the number of residual evaluations correlates strongly with the number of iterations. Any differences between the local evaluations for the flux-search solvers and the global line-search approach are not significant compared to the effect of additional iterations.

4.3 Example 3: Three-phase gravity segregation

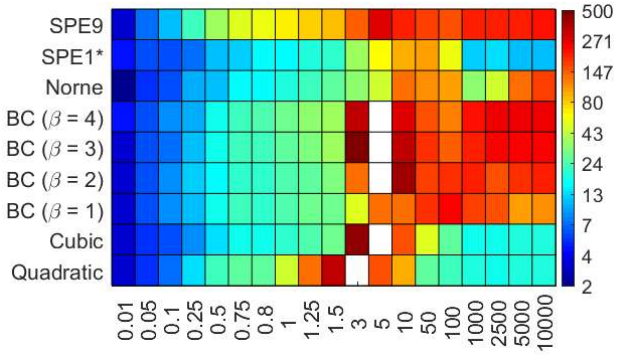
The natural next step is to consider a three-phase gravity segregation problem. We introduce an additional gas phase, which is lighter than the oil phase by a factor 2. We now have three phases, with densities 1000 kg/m^3, 500 kg/m^3, and 250 kg/m^3, respectively. The initial model is layered so that the different phases will move to another layer as shown in the right part of Figure 8. Aside from the additional phase present, the problem is analogous to the two-phase version.



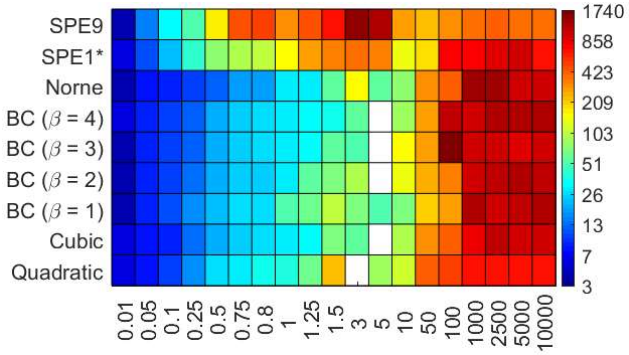
(a) **Iterations:** Newton's method



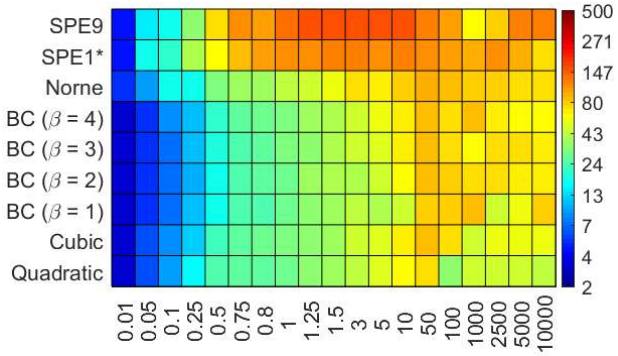
(b) **Iterations:** Newton's method ($\Delta S_{\max} = 0.1$)



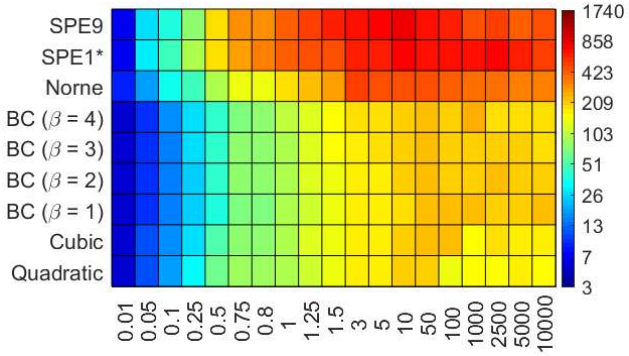
(c) **Iterations:** Line search



(d) **Residual evaluations:** Line search



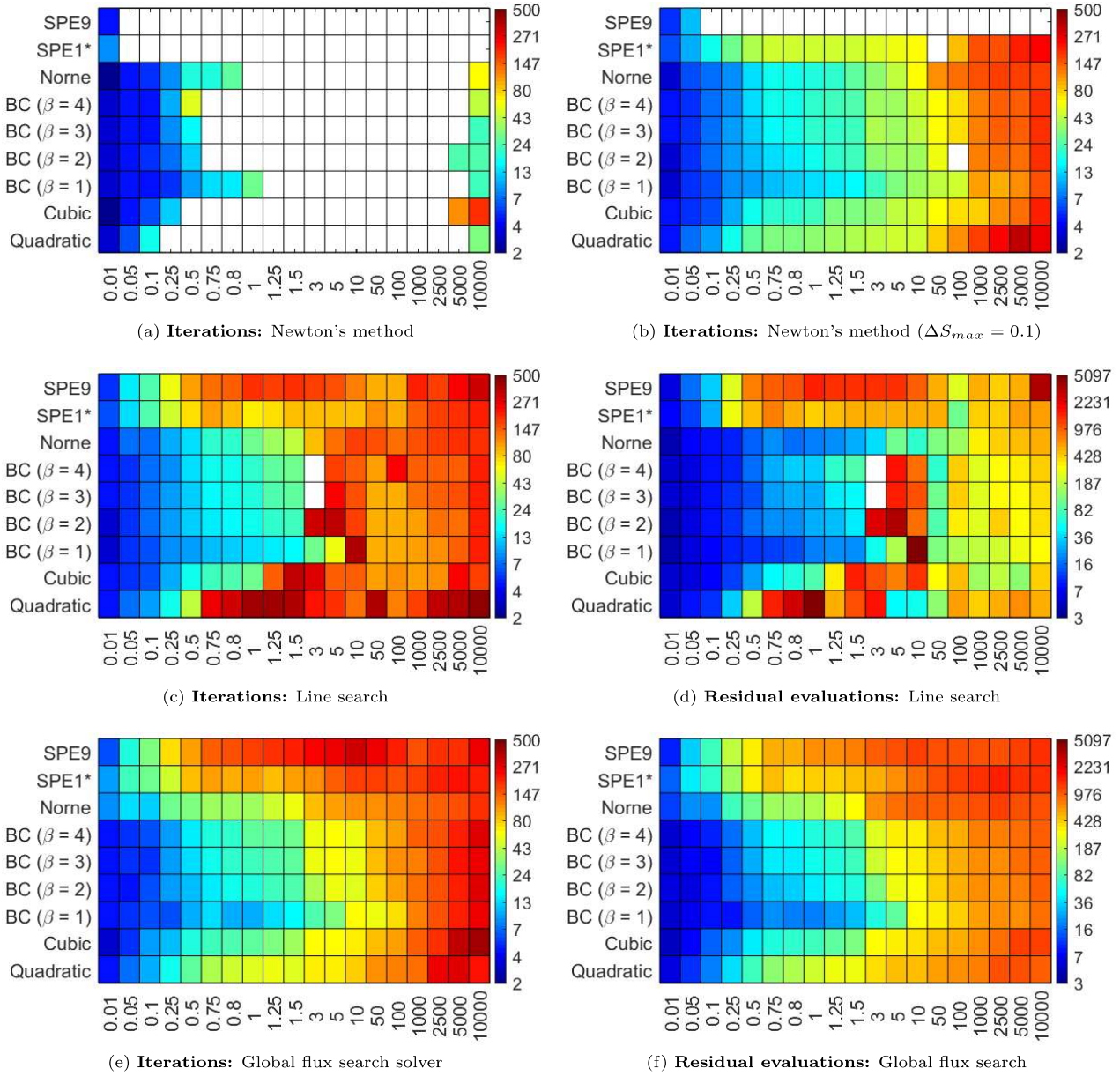
(e) **Iterations:** Global flux search



(f) **Residual evaluations:** Global flux search

Figure 10: Iterations until convergence for the two-phase gravity segregation problem. Note that the colormap is the same for all figures, and white squares represent time-step and relative permeability combinations for which the solver did not converge within 500 iterations.

The convergence plots shown in Figure 11 clearly show that the problem is significantly more challenging than the two-phase version. As all phases are present in a large number of cells during the simulation, problems that were trivial in the two-phase case now require a larger number of iterations. For instance, quadratic relative permeabilities were not a problem for any solver in the two-phase case, but in the three-phase version this problem has the slowest convergence of any of the analytical functions. This is especially evident for the line-search, which uses almost 500 iterations near the characteristic time of the system. Qualitatively, the relative convergence rates seem comparable to the two-phase case as the piecewise linear functions are still significantly harder than the analytical curves and standard Newton struggles with the longer time steps. The flux-search solver outperforms the modified Newton's method for long time steps and gives unconditional convergence. For shorter time steps, the convergence rates are comparable, but we



(a) **Iterations:** Newton's method

(b) **Iterations:** Newton's method ($\Delta S_{max} = 0.1$)

(c) **Iterations:** Line search

(d) **Residual evaluations:** Line search

(e) **Iterations:** Global flux search solver

(f) **Residual evaluations:** Global flux search

Figure 11: Iteration results for the three-phase gravity segregation problem for a wide variety of different time steps and relative permeability curves.

observe that the trust-region solver uses slightly more iterations to converge for the analytical models. This likely occurs because the chopping strategy can be overly conservative, especially when multiple inflection lines and upwind changes can be present in three-phase flow.

4.4 Example 4: Sloped injection problem

In the previous example, we demonstrated the ability to take long time steps for gravity segregation. While challenging, such problems are missing part of the nonlinearity inherent in transport problems. Notably, the total velocity $\vec{v}_T$ was zero for the gravity column, and the only contribution to the flux (4) is due to gravity. Most simulation problems of interest contain a mixture of both viscous and gravity fluxes, and in this example we therefore consider a sloped injection problem where both forces are present. The next example has been

designed to create a highly challenging problem with competing forces.

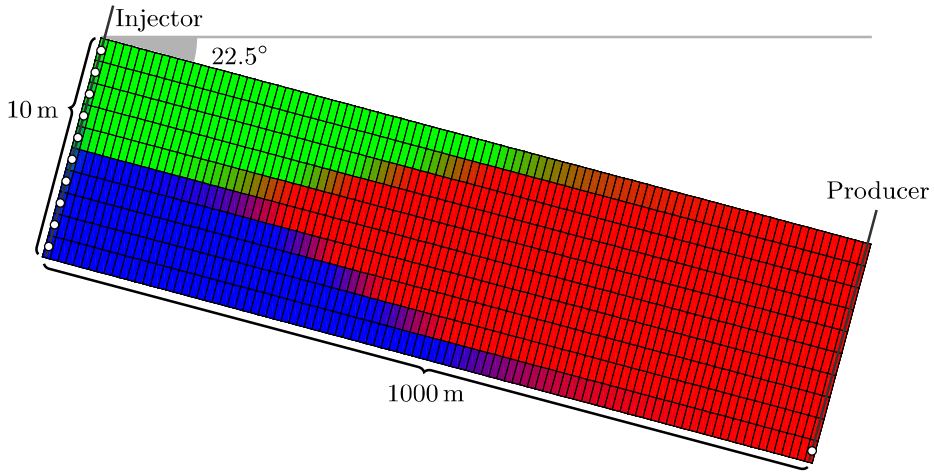


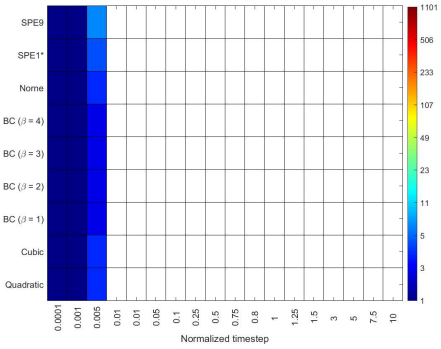
Figure 12: Sloped injection problem

The domain of the problem, the wells, and the injected fluids after 0.4 pore volumes injected (PVI) in the simulation for linear relative permeability curves are shown in Figure 12. The domain is a rectangular grid with size $1000\,\mathrm{m} \times 10\,\mathrm{m}$ uniformly discretized into 1000×10 fine cells. This is a very thin model where the force of gravity is strong due to the small vertical scale, making the injection scenario intentionally challenging for any transport solver. We consider the same density ratios as in the three-phase gravity example and inject a mixture of equal volumes of gas and water at the left edge of the domain using a standard well model. Fluids are produced by a well with a single perforation in the lower right corner at the opposite end of the domain, and the domain is tilted with respect to the x-axis by 22.5 degrees. By tilting the domain, we let the lighter gas front sweep against the direction of the buoyancy force, whereas water is flowing with the buoyancy force, giving the potential for circulations near the front. We let the injector well be rate controlled and normalize the time step so that $\Delta t = 1$ is the same as one PVI. For a variety of time steps ranging from $\Delta t = 10^{-4}$ to $\Delta t = 10$, we simulate injection with the nine different relative permeability curves.

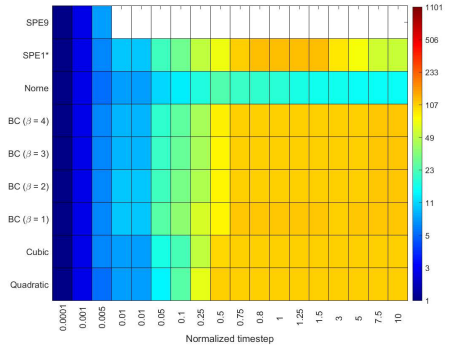
The convergence is plotted in Figure 13. We immediately see that standard Newton struggles with time steps larger than $5 \cdot 10^{-4}$ for most fluid models due to the strong nonlinearity of the problem. By introducing saturation chopping, however, long time steps can be taken for the analytical relative permeability curves. Occasionally, the solver based on standard chopping will fail to solve certain steps, even though both larger and smaller time steps may converge, which is a behavior often seen when the true solution in specific cells lies close to some inflection point or kink. The flux-search solver based on global chopping is able to converge unconditionally, but it uses significantly more iterations for some of the analytical relative permeability curves than modified Newton with saturation chopping. This happens as the solver will always cross inflection lines and upwind changes in a step-wise fashion, which may be needlessly conservative if the true solution is nowhere near the problematic regions. By using the graph-based local chopping strategy, we significantly reduce the number of iterations required until convergence for most cases. Even though only a single front is progressing through the domain, there are still a large number of updates that can be performed independently of each other. Comparing with the line-search solver, we observe that the flux-search solver uses less or a comparable amount iterations for the short and intermediate time-steps. For the longer time-steps, the line-search performs uses generally less iterations, except for the SPE9 fluid model where line-search does not converge in 2000 iterations for 0.75 PVI and in general requires a large number of iterations.

4.5 Example 5: Gas and water injection in anisotropic medium

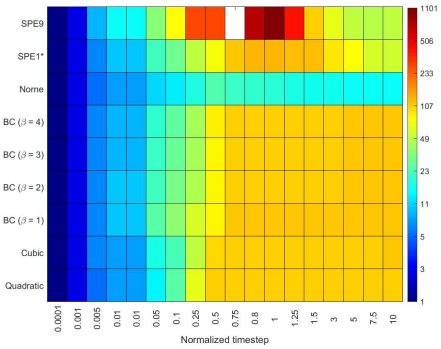
The next problem is an anisotropic test case with a highly irregular flow field. The petrophysical parameters are taken from a subset of the SPE10, Model 2 benchmark problem [5]. The permeability and porosity fields were originally intended as a benchmark for upscaling methods, and as such the contrasts are highly challenging. We take the top 35 layers, corresponding to the Tarbert formation and extract a vertical slice



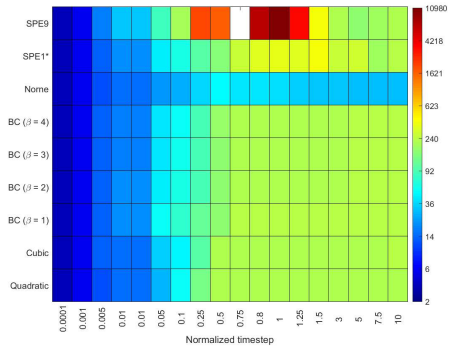
(a) **Iterations:** Newton's method



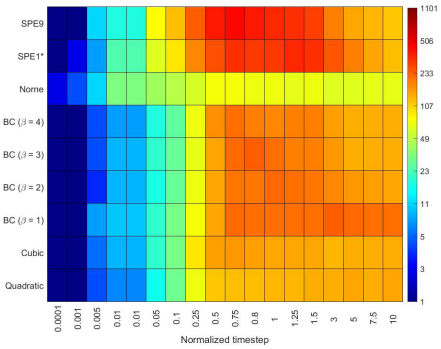
(b) **Iterations:** Newton's method ($\Delta S_{max} = 0.1$)



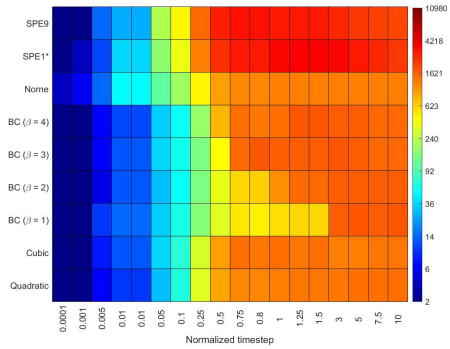
(c) **Iterations:** Line search iterations



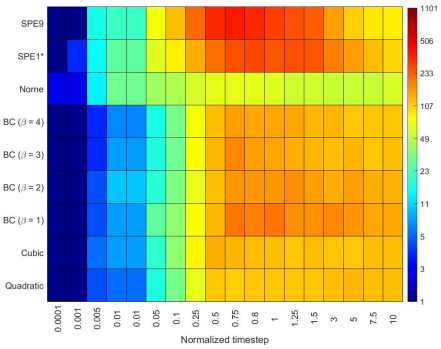
(d) **Residual evaluations:** Line search



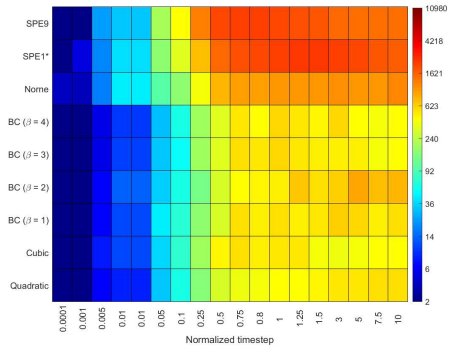
(e) **Iterations:** Global flux search iterations



(f) **Residual evaluations:** Global flux search

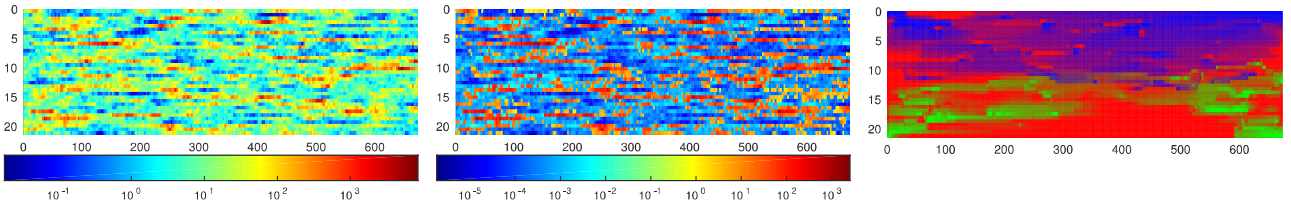


(g) **Iterations:** Local flux search iterations



(h) **Residual evaluations:** Local flux search

Figure 13: Iterations and residual evaluations required to achieve convergence for the three-phase sloped injection problem. The time scale represents the number of pore volumes injected.



(a) Horizontal permeability (mD) (b) Vertical permeability (mD) (c) Saturation after 50 years (1 PVI), $k_{r\alpha} = S_\alpha^2$

Figure 14: The model setup for the vertical SPE10 Model 2 slice.

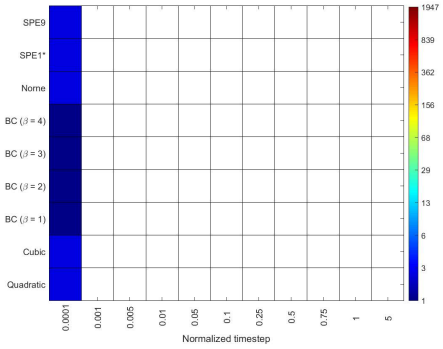
along the middle of the model. The resulting grid has 220×35 cells as shown in Figure 14, and is in some ways more difficult to solve than a 3D subset, as the lateral connections around the impermeable and low-permeable zones have been removed when considering a single vertical slice. We place water injectors in the upper corners of the model as well as gas injectors at the deepest corners, with a single producer in the middle of the grid. By letting the injection rate be relatively slow at one pore volume injected over 50 years, we get both the effects of gravity segregation, in which gas flows up and water flows down, as well as viscous flow when the fluids volumes are swept towards the producer. Parts of the model contain nearly impermeable vertical connections, forcing the fluids to follow the layered structure until it reaches a permeable zone. Once again, we use the same fluid models as in the previous three-phase examples and simulate for a wide variety of time steps and relative permeability curves.

The results for the different configurations are shown in Figure 15. We can see that the standard Newton solver only converges for the smallest time step, while the Newton with saturation chopping performs fairly well for analytical fluid models. As in previous examples, we observe that the piecewise linear relative permeability curves from SPE1 and SPE9 are clearly the hardest to solve, as even the modified Newton does not improve significantly upon the results from standard Newton. When focusing on the two flux-search solvers, we see that both converge for all time-step and fluid combinations, with the exception of certain time-steps around one PVI for the SPE 9 fluid model. It is not known if the convergence failure is due to limitations in the trust-region approach or if the solvers would eventually converge slowly after more than the allotted 2000 iterations, but it is clear that the SPE 9 fluid model represents a highly challenging system, which may not be well posed for long time steps. The larger time steps can often result in initial Newton updates to cells being on the order of several thousands, far beyond the neatly scaled unit values allowable for saturations. The line-search performs comparably to both flux-search solvers, although more iterations are required for the very small time-steps. As in the previous examples, the piecewise linear relative permeability curves are in general more difficult to converge regardless of solver choice.

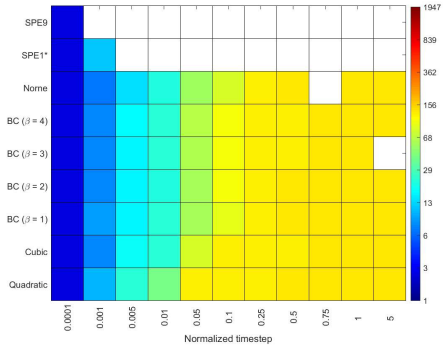
The local flux-search strategy performs comparably or better than the global strategy for modifying the Newton update. The results clearly show that the localized approach outperforms the global reduction, while still being competitive with Newton with saturation chopping. To quantify the behavior of the local and global solvers, we take the modified SPE1 fluid and simulate 1 PVI with different number of time steps. The finest schedule uses 300 equal steps and the coarsest uses a single step. The results are shown in Figure 16, where the local reduction in general outperforms the global solver with a factor 1.48 to 2.44.

4.6 Example 5: Water and gas injection in the Norne field

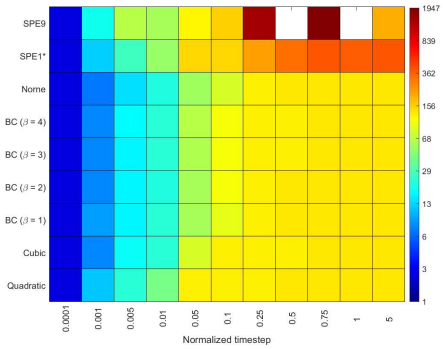
The simulation model of the Norne oil field located in the Norwegian Sea has recently been released as a open data set. The simulation grid contains 44,420 active cells after processing and is complex in the sense that it has a number of faults, eroded cells, and non-neighboring connections. We extract a representative subset of six vertical injectors and seven horizontal producers from the large number of wells present in the full historical model. The injectors are modified to inject an equal volume mixture of gas and water, and are together assigned fixed rates corresponding to one pore volume injected over 50 years for the entire model. This gives a time-averaged water-alternating-gas (WAG) or simultaneous water-alternating-gas (SWAG) scenario. The initial model is equilibrated with all three phases present. We set the water phase to have a density of 1000 kg/m^3, the oil phase to 500 kg/m^3, and the gas phase to 10 kg/m^3 to give vastly different impact from gravity to the phases. The phase viscosities are 1, 5, and 0.1 cP for water, oil, and gas, respectively.



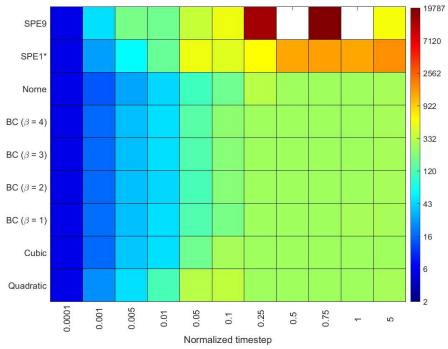
(a) **Iterations:** Newton's method



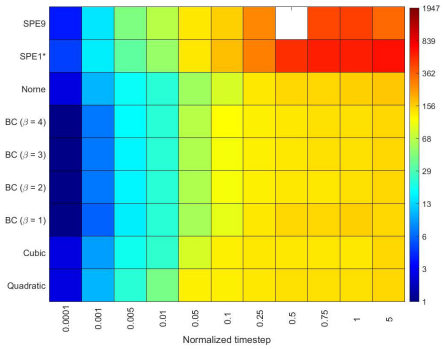
(b) Newton's method iterations ($\Delta S_{max} = 0.1$)



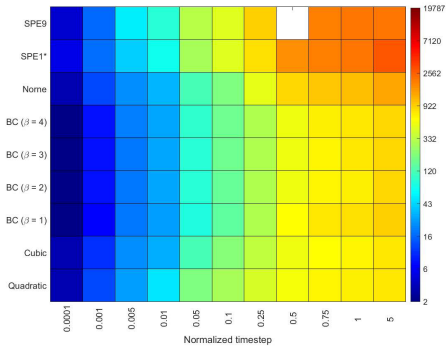
(c) **Iterations:** Line search



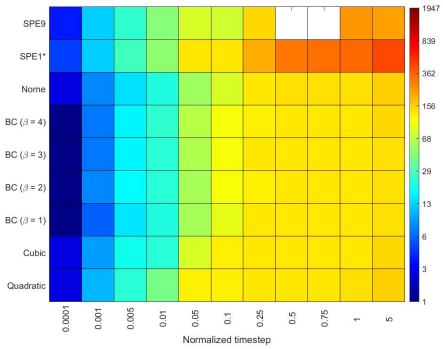
(d) **Residual evaluations:** Line search



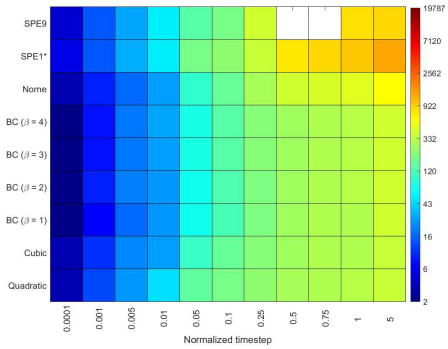
(e) **Iterations:** Global flux search iterations



(f) **Residual evaluations:** Global flux search

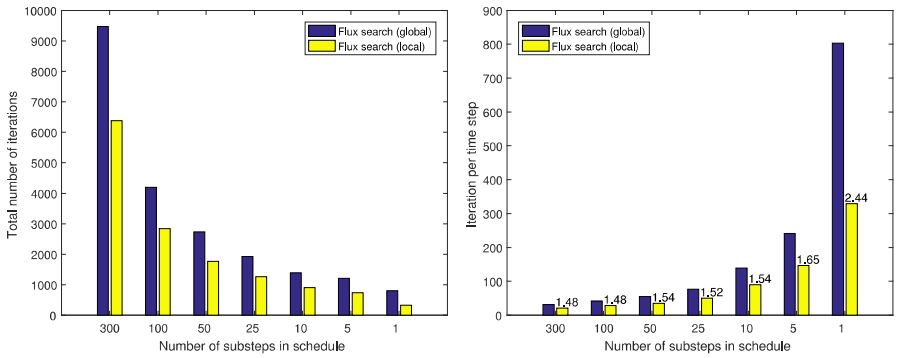


(g) **Iterations:** Local flux search



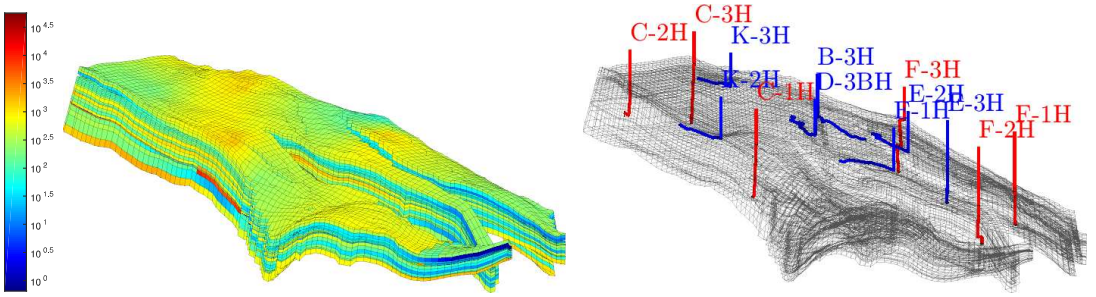
(h) **Residual evaluations:** Local flux search

Figure 15: Iterations until convergence for a vertical cross section of the Tarbert formation from SPE10, Model 2. The time scale represents the number of pore volumes injected.



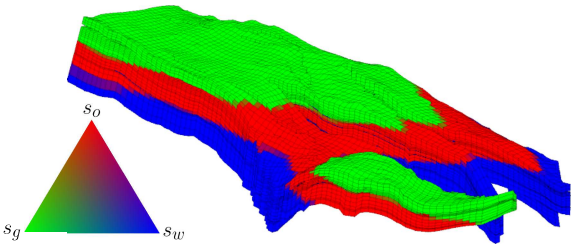
(a) Breakdown of the total number of iterations (b) Number of iterations per step as a function of the number of steps

Figure 16: Time-step study for the vertical SPE10 cross section. The model is given the modified SPE1 relative permeabilities and 1 PVI is simulated for several different number of time steps ranging from 300 relatively small steps, to one big step. The numbers above the graph shows the relative increase in iterations when going from the local to the global flux-search solver.

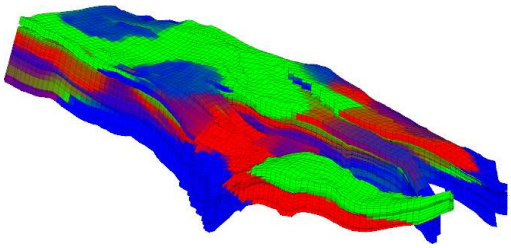


(a) Vertical permeability (mD)

(b) Grid and wells: Injectors are red and producers are blue



(c) Phase saturations at $t = 0$



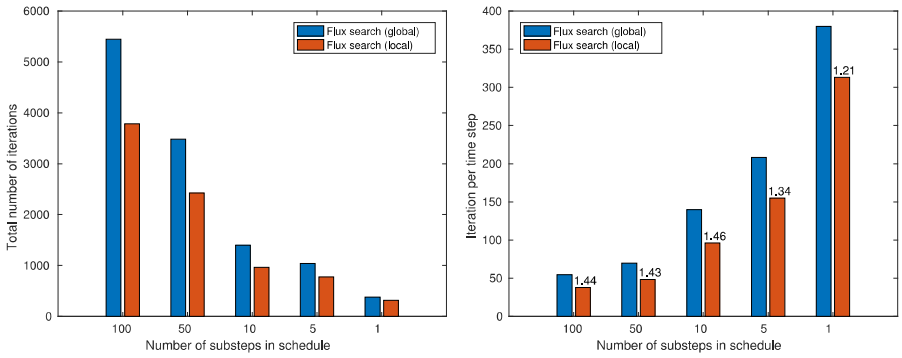
(d) Water, oil and gas saturations after 1 PVI

Figure 17: Permeability, grid, well configurations and saturations for the Norne example.

We use the same Norne relative permeability system as in the earlier examples and simulate until one pore volume has been injected. By simulating a full scenario, we will expose the nonlinear solvers to many of the situations present in a real reservoir simulation. Five different time-step schedules were created, ranging from 100 time steps of moderate length, to one long step encompassing the entire scenario. The grid, permeability and saturations at $t = 0$ and $t = 50$ years can be seen in Figure 17.

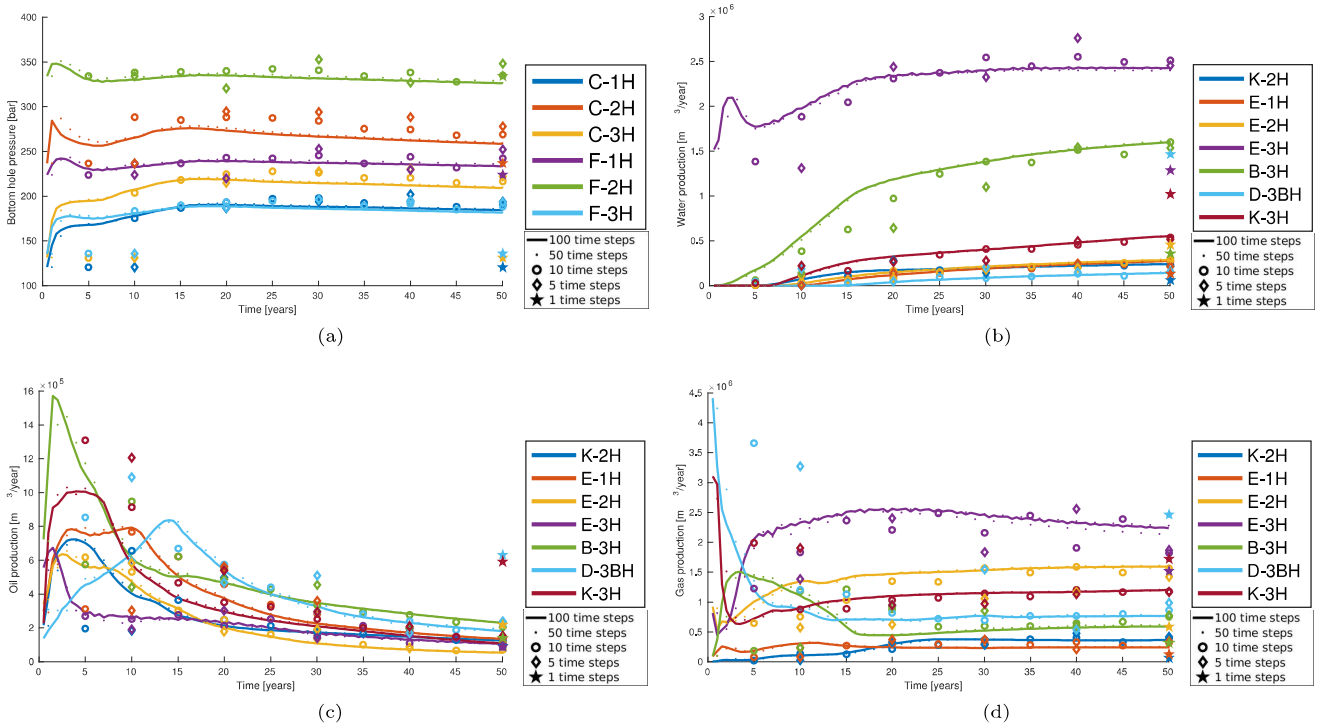
Due to the strong nonlinearities of the injection process, neither Newton nor modified Newton were able to solve the scenario without halving the time steps more times than allowed by the simulator. The trust-region solvers were able complete all scenarios without any time-step failures. Visualization of the iterations as a function of the number of time steps can be seen in Figure 18, where we see that the global solver on average seems to use between 20 and 50 % more iterations to converge than to the local solver. For the extreme case

of a single 50-year time step, we observe that the difference between the solvers is smaller as the local solver uses 313 iterations and the global solver uses 380 iterations. As the time step becomes longer, information from the different parts of the reservoir will interact and make it more difficult to localize the updates.



(a) Breakdown of the total number of iterations (b) Number of iterations per time step

Figure 18: Time-step study for the Norne field model. The model uses a subset of the real wells, and 1 PVI is simulated for several different number of time steps ranging from 100 moderate steps, to one big step. The numbers above the graph report the relative increase in iterations when going from the local to the global flux-search solver.

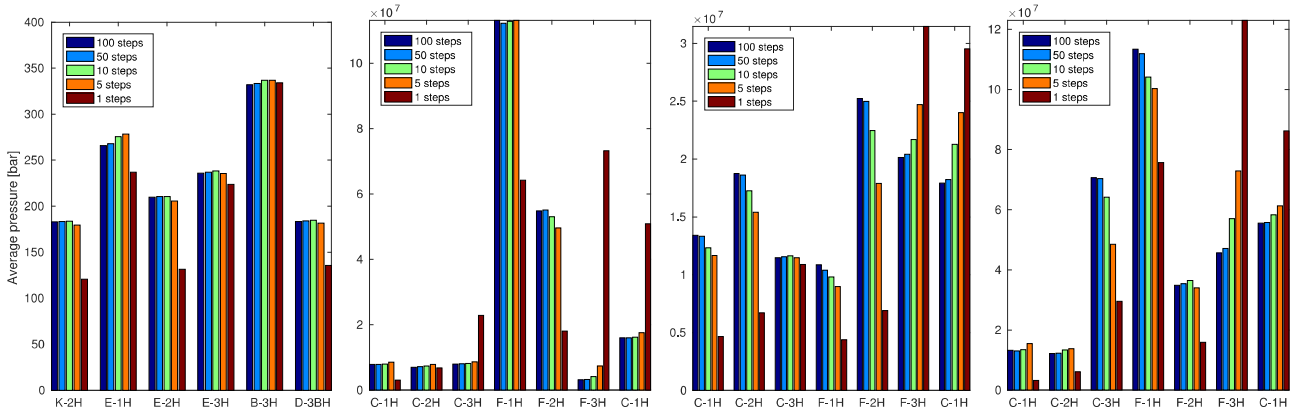


(a) (b)

(c) (d)

Figure 19: Bottom-hole pressure for the injectors and water, oil, and gas rates for the producers for a wide variety of different time-step lengths for the Norne model.

In the previous examples, we only considered the iterations until convergence for the different time steps. Systematic tests have value in that they show the convergence behavior of the different solvers, but in practice the results from the simulation are more important. When long time steps are taken, the time truncation error increases and the accuracy of the simulation results will suffer. The degree to which large errors in



(a) Averaged injector BHP (b) Produced water (c) Produced oil (d) Produced gas

Figure 20: The produced volumes and time-averaged injector bottom-hole pressures for different time-step lengths applied to the Norne model

the results are acceptable varies depending upon the purpose of the simulation. If the goal is to resolve fine-scale fluid behavior and correctly predict water or gas breakthrough within days, small time steps should be taken. For optimization and ballpark estimates, however, long time steps can be sufficiently predictive, especially when uncertainty in the underlying data can be large. It is outside the scope of this paper to determine the optimal time-step length to balance error and convergence rates. Nevertheless, we report well curves for the different time-stepping strategies in Figure 19. From the figure, we see that the quantitative details of each curve deteriorates much faster than the qualitative behavior. The total produced volumes for each well in Figure 20 indicate that even with only 10 time steps, the lifetime prediction is relatively accurately reproduced. Not surprisingly, the single long time step gives well curves that are far off for most wells, and this simulation is only included as a proof-of-concept for the improved convergence achieved by the trust-region solver with flux-search.

5 Conclusion

For implicit schemes, time-step restrictions on the transport in porous media are more often than not limited by the capability of the nonlinear solver rather than stability restrictions in temporal discretization. This happens in part because the parameters used to simulate specific physical phenomena result in highly coupled non-linear equations. In this work, we have extended the trust-region approach to significantly more complex problems by the way of three key innovations. Firstly, we have built a trust-region solver capable of solving for three (or more) fluid phases by projecting the updates along the Newton path and applying a flux-search algorithm. Secondly, we have introduced the ability to handle general flux functions, where the underlying parameters are not necessarily smooth by using a monotone interpolation scheme. And finally, we have introduced a local update strategy based on the flow of information in the hyperbolic problems with gravity, where the graph of the Jacobian is decomposed into strong and weak connections.

The numerical examples, while limited to immiscible and incompressible flow, are highly nonlinear due to the combination of anisotropic permeability, semi-unstructured grids, non-smooth relative permeability curves, and strong coupling between gravity and viscous forces. Our new flux-search, trust-region solver compared favorably with standard Newton and modified Newton with saturation chopping and Newton with saturation chopping and a residual line-search with backtracking. Generally, the results indicate that the flux-search performs better than the line-search for small time-steps that always converge. For longer time-steps, the flux-search is comparable or slightly slower to converge than the line-search due to systematic cutting of all inflection points and upwind changes. For problems considering piecewise linear relative permeability curves from standard benchmark datasets, the flux-search generally performed better than the line-search.

6 Acknowledgments

This research was funded in part by Schlumberger Information Solutions and by the Research Council of Norway under grant no. 226035. The author would like to thank Bradley Mallison and Hamdi Tchelepi for constructive discussions, as well as Knut-Andreas Lie for proofreading my drafts. The author also wants to thank Statoil (operator of the Norne field) and its license partners ENI and Petoro for the release of the Norne data. Further, the author acknowledges the IO Center at NTNU for coordination of the Norne cases.

References

[1] Baker, L.: Three-phase relative permeability correlations. In: SPE Enhanced Oil Recovery Symposium. Society of Petroleum Engineers (1988)

[2] Bratvedt, F., Gimse, T., Tegnander, C.: Streamline computations for porous media flow including gravity. Transp. Porous Media 25(1), 63–78 (oct 1996)

[3] Brenier, Y., Jaffré, J.: Upstream differencing for multiphase flow in reservoir simulation. SIAM Journal on Numerical Analysis 28(3), 685–696 (1991)

[4] Brooks, R.H., Corey, A.T.: Hydraulic properties of porous media and their relation to drainage design. Transactions of the ASAE 7(1), 26–0028 (1964)

[5] Christie, M.A., Blunt, M.J.: Tenth spe comparative solution project: A comparison of upscaling techniques. In: SPE Reservoir Simulation Symposium. Society of Petroleum Engineers (2001)

[6] Fritsch, F.N., Carlson, R.E.: Monotone piecewise cubic interpolation. SIAM Journal on Numerical Analysis 17(2), 238–246 (1980)

[7] Gmelig Meyling, R.H.J.: A characteristic finite element method for solving non-linear convection-diffusion equations on locally refined grids. In: Guerillot, D., Guillon, O. (eds.) 2nd European Conference on the Mathematics of Oil Recovery. pp. 255–262. Editions Technip, Arles, France (Sept 11-14 1990)

[8] Gmelig Meyling, R.H.J.: Numerical methods for solving the nonlinear hyperbolic equations of porous media flow. In: Third International Conference on Hyperbolic Problems, Vol. I, II (Uppsala, 1990), pp. 503–517. Studentlitteratur, Lund (1991)

[9] Gries, S., Stüben, K., Brown, G.L., Chen, D., Collins, D.A., et al.: Preconditioning for efficiently applying algebraic multigrid in fully implicit reservoir simulations. SPE Journal 19(04), 726–736 (2014)

[10] Hamon, F.P., Tchelepi, H.A.: Analysis of hybrid upwinding for fully-implicit simulation of three-phase flow with gravity. SIAM Journal on Numerical Analysis 54, 1682–1712 (2016)

[11] IO Center, NTNU: The Norne benchmark case. url: `http://www.ipt.ntnu.no/~norne/wiki/doku.php` (2012)

[12] Jenny, P., Lee, S.H., Tchelepi, H.A.: Multi-scale finite-volume method for elliptic problems in subsurface flow simulation. J. Comput. Phys. 187, 47–67 (2003)

[13] Jenny, P., Tchelepi, H.A., Lee, S.H.: Unconditionally convergent nonlinear solver for hyperbolic conservation laws with s-shaped flux functions. Journal of Computational Physics 228(20), 7497–7512 (2009)

[14] Killough, J., Wheeler, M., et al.: Parallel iterative linear equation solvers: An investigation of domain decomposition algorithms for reservoir simulation. In: SPE Symposium on Reservoir Simulation. Society of Petroleum Engineers (1987)

[15] Killough, J., et al.: Ninth spe comparative solution project: A reexamination of black-oil simulation. In: SPE Reservoir Simulation Symposium. Society of Petroleum Engineers (1995)

[16] Kippe, V., Aarnes, J.E., Lie, K.A.: A comparison of multiscale methods for elliptic problems in porous media flow. Comput. Geosci. 12(3), 377–398 (2008)

[17] Kwok, F., Tchelepi, H.: Potential-based reduced newton algorithm for nonlinear multiphase flow in porous media. Journal of Computational Physics 227(1), 706–727 (2007)

[18] Lee, S., Efendiev, Y.: C1-continuous relative permeability and hybrid upwind discretization of three phase flow in porous media. Advances in Water Resources 96, 209–224 (2016)

[19] Lee, S., Efendiev, Y., Tchelepi, H.: Hybrid upwind discretization of nonlinear two-phase flow with gravity. Advances in Water Resources 82, 27–38 (2015)

[20] Li, B., Tchelepi, H.A.: Unconditionally convergent nonlinear solver for multiphase flow in porous media under viscous force, buoyancy, and capillarity. Energy Procedia 59, 404–411 (2014)

[21] Li, B., Tchelepi, H.A.: Nonlinear analysis of multiphase transport in porous media in the presence of viscous, buoyancy, and capillary forces. Journal of Computational Physics 297, 104–131 (2015)

[22] Lie, K.A., Natvig, J.R., Nilsen, H.M.: Discussion of dynamics and operator splitting techniques for two-phase flow with gravity. Int. J Numer. Anal. Mod. (Special issue in memory of Magne Espedal) 9(3), 684–700 (2012)

[23] Lie, K.A., Nilsen, H.M., Rasmussen, A.F., Raynaud, X., et al.: Fast simulation of polymer injection in heavy-oil reservoirs on the basis of topological sorting and sequential splitting. SPE Journal 19(06), 991–1 (2014)

[24] Møyner, O., Lie, K.A.: A multiscale restriction-smoothed basis method for high contrast porous media represented on unstructured grids. Journal of Computational Physics 304, 46–71 (2016)

[25] Natvig, J.R., Lie, K.A.: Fast computation of multiphase flow in porous media by implicit discontinuous galerkin schemes with optimal ordering of elements. Journal of Computational Physics 227(24), 10108–10124 (2008)

[26] Odeh, A.S.: Comparison of solutions to a three-dimensional black-oil reservoir simulation problem. Journal of Petroleum Technology 33(01), 13–25 (1981)

[27] Schlumberger: ECLIPSE 2013.2 Technical Description (2013)

[28] Sheth, S., Younis, R.: Localized computation of newton updates in fully-implicit two-phase flow simulation. Procedia Computer Science 80, 1392–1403 (2016)

[29] Stone, H.: Probability model for estimating three-phase relative permeability. Journal of Petroleum Technology 22(02), 214–218 (1970)

[30] Stone, H.: Estimation of three-phase relative permeability and residual oil data. Journal of Canadian Petroleum Technology 12(4) (1973)

[31] The Open Porous Media Initative: The Norne dataset. url: `https://github.com/OPM/opm-data` (2015)

[32] Thiele, M.R., Batycky, R.P., Blunt, M.J.: A streamline-based 3d field-scale compositional reservoir simulator. In: SPE Annual Technical Conference and Exhibition. San Antonio, Texas, USA (5–8 October 1997), sPE 38889

[33] Trangenstein, J.A., Bell, J.B.: Mathematical structure of the black-oil model for petroleum reservoir simulation. SIAM J. Appl. Math. 49(3), 749–783 (1989)

[34] Trottenberg, U., Oosterlee, C.W., Schuller, A.: Multigrid. Academic press (2000)

[35] Voskov, D.V., Tchelepi, H.A., et al.: Compositional nonlinear solver based on trust regions of the flux function along key tie-lines. In: SPE Reservoir Simulation Symposium. Society of Petroleum Engineers (2011)

[36] Voskov, D., Tchelepi, A.: Tie-simplex-based nonlinear solver for mass-variables compositional formulation. In: ECMOR XIII-13th European Conference on the Mathematics of Oil Recovery (2012)

[37] Wallis, J., et al.: Incomplete gaussian elimination as a preconditioning for generalized conjugate gradient acceleration. In: SPE Reservoir Simulation Symposium. Society of Petroleum Engineers (1983)

[38] Wang, X., Tchelepi, H.A.: Trust-region based solver for nnonlinear transport in heterogeneous porous media. Journal of Computational Physics 253, 114–137 (2013)

[39] Watts, J.W.: A compositional formulation of the pressure and saturation equations. SPE J. 1(3), 243–252 (1986)

[40] Younis, R., Tchelepi, H.A., Aziz, K., et al.: Adaptively localized continuation-newton method–nonlinear solvers that converge all the time. SPE Journal 15(02), 526–544 (2010)

Appendix

Algorithm 2 Algorithm for a global line search with backtracking.

procedure LINESEARCH($\vec{x}_0, \Delta\vec{x}$)
  $\alpha \leftarrow 1$
  for $i \leq i_{max}$ **do**
    $\vec{x} \leftarrow \vec{x}_0 + \alpha\Delta\vec{x}$
    if $|\vec{x}\|_\infty < |\vec{x}_0\|_\infty$ **then**
      return $\vec{x}$
    else
      $\alpha \leftarrow \alpha/2$
    end if
  end for
end procedure

Algorithm 3 The algorithm for selecting the optimal viscous relaxation factor.

procedure COMPUTEVISCOUSRELAXATION($\mathbf{F}', \mathbf{w}$)
  $\mathbf{F}''_* \leftarrow$ `evaluateApproximateDerivative`$(\mathbf{F}', \mathbf{w})$
  $i \leftarrow 1$
  $\omega^V \leftarrow 1$
  for $i \leq$ `numel`$(\mathbf{F}''_*)$ **do**
    if $\text{sign}(\mathbf{F}''_*[i]) \neq \text{sign}(\mathbf{F}''_*[1])$ & $\mathbf{F}''_*[i] \neq 0$ **then**
      $\omega^V \leftarrow \mathbf{w}[i]$
      break
    end if
  end for
  if $\omega^V < 1$ **then**
    $\mathbf{w}^V_{next} \leftarrow$ `addSamplePoints`$(\mathbf{w}, i)$
  else
    $\mathbf{w}^V_{next} \leftarrow [\quad]$
  end if
  return $\omega^V, \mathbf{w}^V_{next}$
end procedure

Algorithm 4 The algorithm for selecting the optimal relaxation factor based on changes in the upwind direction.

procedure computeGravityRelaxation($\mathbf{G}, \mathbf{w}$)
  $i \leftarrow 1$
  $\omega^G \leftarrow 1$
  for $i \leq$ `numel`($\mathbf{G}$) **do**
    if $\mathbf{G}[i] \neq \mathbf{G}[i-1]$ **then**
      $\omega^G \leftarrow \mathbf{w}[i]$
      break
    end if
  end for
  if $\omega^G < 1$ **then**
    $\mathbf{w}^G_{next} \leftarrow$ `addSamplePoints`($\mathbf{w}, i$)
  else
    $\mathbf{w}^G_{next} \leftarrow [\quad]$
  end if
  return $\omega^G, \mathbf{w}^G_{next}$
end procedure

Algorithm 5 Evaluation of second derivative using a spline at the midpoints of each interval.

procedure evaluateApproximateDerivative($\mathbf{F}', \mathbf{w}$)
  $p(\omega) \leftarrow$ `spline`($\mathbf{F}', \mathbf{w}$)
  $g(\omega) \leftarrow \frac{\partial p}{\partial \omega}$
  for $i \leq$ `numel`($\mathbf{w}$) **do**
    if $i = 1$ **then**
      $F''_*[i] \leftarrow g(\mathbf{w}[1])$ ▷ Evaluate at $\omega = 0$
    else
      $F''_*[i] \leftarrow g(\mathbf{w}[i-1] + \frac{\mathbf{w}[i]-\mathbf{w}[i-1]}{2})$ ▷ Evaluate at interval midpoint
    end if
  end for
end procedure

Algorithm 6 Procedure for adding additional sample points given the index i of the first relaxation factor to be outside the trust region.

procedure addSamplePoints($\mathbf{w}, i$)
  if $i = 2$ **then**
    $\mathbf{w}_{next} \leftarrow \{\mathbf{w}[i-1] + \frac{\mathbf{w}[i]-\mathbf{w}[i-1]}{2}\}$
  else
    $\mathbf{w}_{next} \leftarrow \{\mathbf{w}[i-2] + \frac{\mathbf{w}[i-1]-\mathbf{w}[i-2]}{2}, \mathbf{w}[i-1] + \frac{\mathbf{w}[i]-\mathbf{w}[i-1]}{2}\}$
  end if
  return $\mathbf{w}_{next}$
end procedure