Efficient Steady-State Solution Techniques for
Variably Saturated Groundwater Flow

Matthew W. Farthing, a,* Christopher E. Kees, b
Todd S. Coffey, c C.T. Kelley b Cass T. Miller a

a Center for the Advanced Study of the Environment, Department of
Environmental Sciences and Engineering, University of North Carolina, Chapel
Hill, North Carolina 27599-7431, USA
b Center for Research in Scientific Computation, Department of Mathematics,
North Carolina State University, Raleigh, North Carolina, 27695-8205, USA
c Mathematical Information and Computational Sciences, Sandia National
Laboratories, Albuquerque, New Mexico, 87185-1110, USA

Abstract

We consider the simulation of steady-state variably saturated groundwater flow using Richards’ equation (RE). The difficulties associated with solving RE numerically are well known. Most discretization approaches for RE lead to nonlinear systems that are large and difficult to solve. The solution of nonlinear systems for steady-state problems can be particularly challenging, since a good initial guess for the steady-state solution is often hard to obtain, and the resulting linear systems may be poorly scaled. Common approaches like Picard iteration or variations of Newton’s method have their advantages but perform poorly with standard globalization techniques under certain conditions.

Pseudo-transient continuation has been used in computational fluid dynamics for
**Efficient Steady-State Solution Techniques for Variably Saturated Groundwater Flow**

**North Carolina State University, Center for Research in Scientific Computation, Raleigh, NC, 27695-8205**

**DISTRIBUTION/AVAILABILITY STATEMENT**

Approved for public release; distribution unlimited

**SUPPLEMENTARY NOTES**

The original document contains color images.

**ABSTRACT**

see report

**SUBJECT TERMS**

- Variably saturated groundwater flow
- Efficient solutions
- Numerical methods
- Scientific computation
some time to obtain steady-state solutions for problems in which Newton’s method with standard line-search strategies fails. It combines aspects of backward Euler time integration and Newton’s method to select intermediate estimates of the steady-state solution. Here, we examine the use of pseudo-transient continuation as well as Newton’s method combined with standard globalization techniques for steady-state problems in heterogeneous domains. We investigate the methods’ performance with direct and preconditioned Krylov iterative linear solvers. We then make recommendations for robust and efficient approaches to obtain steady-state solutions for RE under a range of conditions.

Notation

Roman Letters

\(A\) accumulation term for pressure head form of RE

\(A\) accumulation term contribution to Jacobian

\(J\) Jacobian

\(C\) scaling factor for Dirichlet boundary conditions

\(F\) nonlinear function for DAE formulation, semi-discrete

\(G\) nonlinear function for DAE formulation, discrete

\(K_s\) saturated hydraulic conductivity

\(K_s^*\) surface saturated hydraulic conductivity

* Corresponding author

Email addresses: matthew_farthing@unc.edu (Matthew W. Farthing,), chris_kees@ncsu.edu (Christopher E. Kees,), tscoffe@sandia.gov (Todd S. Coffey,), tim_kelley@ncsu.edu (C.T. Kelley), casey_miller@unc.edu (Cass T. Miller).
$K_e$ effective hydraulic conductivity

$N_{\text{max}}$ maximum number of iterations for nonlinear solution methods

$O_d$ discrete spatial operator

$R$ discrete spatial operator and source term

$S_e$ effective saturation

$T$ extent of temporal domain

$X_d$ extent of spatial domain along $x_d$ axis

$f$ source term for the aqueous phase evaluated at cell centers

$f$ source term for the aqueous phase

$g$ gravitational vector

$g_u$ unit vector, $g/\|g\|$  

$k_i$ intrinsic permeability

$k_r$ relative permeability

$m_v$ parameter for VGM

$n$ unit outward normal for $\Omega$

$n_e$ total number of nodes

$n_v$ parameter for VGM

$n_{x_d}$ number of nodes along $x_d$ axis

$p$ pressure of the aqueous phase

$t$ time coordinate

$u$ mass flux

$u^b$ Neumann boundary value

$u^r$ precipitation rate

$y$ variable for DAE formulation

$y'$ temporal derivative for DAE formulation

$y_{\text{min}}$ lower bound for solution in test for evaluation error

$y_{\text{max}}$ upper bound for solution in test for evaluation error
**Greek Letters**

- $\Delta t$: time step for $\Psi tc$ methods
- $\Delta t_{\text{max}}$: maximum time step for $\Psi tc$ methods
- $\Delta x_d$: spatial increment in $x_d$ direction
- $\Delta y$: Newton increment
- $\Gamma$: boundary of physical domain
- $\Gamma_D$: portion of $\Gamma$ for which Dirichlet boundary conditions are set
- $\Gamma_N$: portion of $\Gamma$ for which Neumann boundary conditions are set
- $\Omega$: physical domain
- $\alpha_v$: parameter for VGM
- $\beta$: compressibility of the aqueous phase
- $\epsilon_c$: switching tolerance in hybrid Newton-Picard method
- $\epsilon_s$: sufficient decrease parameter for Armijo line search
- $\theta$: volume fraction of the aqueous phase
- $\theta_r$: residual volumetric water content
- $\theta_s$: saturated volumetric water content
- $\lambda$: line-search scaling factor
- $\lambda_+$: intermediate scaling factor for quadratic line search
- $\mu$: viscosity of the aqueous phase
- $\rho$: density of the aqueous phase
- $\rho$: normalized density of the aqueous phase
- $\sigma_{\text{min}}$: bound parameter for quadratic line search
- $\sigma_{\text{max}}$: bound parameter for quadratic line search
- $\tau$: TTE parameter for truncation error estimate bound
- $\psi$: pressure head
- $\psi^b$: Dirichlet boundary value
\( \psi^0 \) initial condition for \( \psi \)

**Subscripts and Superscripts**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( d )</td>
<td>coordinate axis identifier (subscript)</td>
</tr>
<tr>
<td>( i )</td>
<td>cell qualifier along ( x_1 ) axis (subscript)</td>
</tr>
<tr>
<td>( j )</td>
<td>cell qualifier along ( x_2 ) axis (subscript)</td>
</tr>
<tr>
<td>( l )</td>
<td>global identifier for solution unknowns (subscript)</td>
</tr>
<tr>
<td>( n )</td>
<td>time level identifier (superscript)</td>
</tr>
<tr>
<td>( s_n )</td>
<td>nonlinear iteration index (superscript)</td>
</tr>
</tbody>
</table>

**Abbreviations**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock</td>
<td>wall clock time (seconds)</td>
</tr>
<tr>
<td>DAE</td>
<td>differential algebraic equation</td>
</tr>
<tr>
<td>Feval</td>
<td>function evaluation</td>
</tr>
<tr>
<td>HAS</td>
<td>two-level hybrid additive Schwarz domain decomposition preconditioner</td>
</tr>
<tr>
<td>Jeval</td>
<td>Jacobian evaluation</td>
</tr>
<tr>
<td>LF</td>
<td>linear iteration failure</td>
</tr>
<tr>
<td>LI</td>
<td>linear iteration</td>
</tr>
<tr>
<td>LU-B</td>
<td>banded LU decomposition from LAPACK</td>
</tr>
<tr>
<td>LU-S</td>
<td>sparse LU decomposition from SPOOLES (version 2.2)</td>
</tr>
<tr>
<td>NILS</td>
<td>Newton’s method with a quadratic line search</td>
</tr>
<tr>
<td>NLF</td>
<td>nonlinear iteration failure</td>
</tr>
<tr>
<td>NLI</td>
<td>nonlinear iteration</td>
</tr>
<tr>
<td>ODE</td>
<td>ordinary differential equation</td>
</tr>
</tbody>
</table>
1 Introduction

Variably saturated groundwater flow is commonly modeled using Richards’ equation (RE) along with a set of constitutive relations describing the interdependence among fluid pressures, saturations, and relative permeabilities ($p$-$S$-$k$ relations). While analytical and semi-analytical approximations for variably saturated flow exist, these are valid for limited sets of auxiliary conditions and domains [30]. As a result, significant effort has focused on developing robust techniques for solving RE numerically [9, 38, 29, 32, 27, 34, 36, 37]. Obtaining solutions for RE for many realistic physical conditions remains a challenge. Infiltration problems are often characterized by sharp fronts in both space and time. Steady-state solution is often nontrivial as well, since the volume fraction can vary steeply for problems with realistic boundary conditions and heterogeneous porous media.

It is the nature of the nonlinearities introduced through standard $p$-$S$-$k$ relations like the van Genuchten [35] and Mualem [28] models that accounts for the majority of the difficulties associated with solving RE. A number of issues associ-
ated with resolving this nonlinear behavior have received attention. These include solution variable transformations [4, 36], the evaluation of $p-S-k$ relations [33], approximation of interface conductivities in standard low-order spatial discretizations [38, 27], choice of dependent variable [37], and time discretization approach [9, 31, 33, 19]. Since the majority of time discretizations are implicit, both transient and steady-state problems typically lead to the solution of a system of discrete nonlinear equations. These systems can be quite large and difficult to solve, especially for problems with heterogeneous domains in two and three spatial dimensions.

The nonlinear system solution method used can then have a strong impact on the overall success of a simulator for RE. The most common approaches have been Picard iteration [9, 10, 19] or a variant of Newton’s method [34, 37, 21]. Each of these methods has its strengths and weaknesses, and several works have compared the robustness and efficiency of Picard and Newton approaches for a variety of problems [29, 25, 27, 8]. Newton’s method combined with globalization techniques like a line search, reduction of time step (backtracking), or the use of Picard iteration to obtain an initial guess has proven more reliable and efficient than Picard iteration for several problems [29, 27]. However, both Newton and Picard approaches have been shown to perform poorly under certain conditions. The solution of nonlinear systems can be particularly difficult for steady-state problems due to the increased difficulty of obtaining a good initial guess for the steady-state solution. In addition, the scaling of the linearized systems is typically worse [15], and there is no longer recourse to reducing time steps when convergence breaks down [29].

Pseudo-transient continuation ($\Psi tc$) has been used in computational fluid dynamics for some time to obtain steady-state solutions for problems in which
Newton’s method with standard line-search strategies fails [23, 24, 11]. \( \Psi tc \) combines features of backward Euler time integration and Newton’s method with the idea of using information about the transient physical problem to guide selection of intermediate iterates to the steady-state solution [15]. Since it uses information from a time-dependent problem, \( \Psi tc \) can be more robust than standard line-search strategies, while incurring less computational expense than full integration of the transient problem [23].

In this work, we seek effective techniques for obtaining steady-state solutions to RE. We examine the use of \( \Psi tc \) methods as well as Newton’s method combined with standard globalization techniques for steady-state problems in homogeneous and heterogeneous domains. We investigate the methods’ performance with both direct and preconditioned Krylov iterative linear solvers. We then make recommendations for robust and efficient approaches to obtain steady-state solutions for RE under various conditions.

2 Background

Historically, Picard iteration has been the most common nonlinear solution method used for RE [9, 31]. It’s appeal can be traced to the fact that it is simple to implement, since it does not involve derivatives of terms involving the \( p-S-k \) relations and produces symmetric linear systems [29]. While Picard iteration is still used [10, 19], Newton and inexact Newton approaches have become significantly more common [14, 18, 21], particularly for more realistic multidimensional problems. An inexact Newton approach can be thought of as Newton’s method where the equation for the Newton update is satisfied only approximately [34]. This may be due to the fact that the Newton update
is solved only approximately using, for example, a Krylov-type iterative linear solver [18]. Or, only an approximate Jacobian may be used in the Newton update. Common approaches for approximating the full Jacobian are to use finite differences (a numerical Jacobian) [14] or to lag its evaluation over a number of iterates (chord or modified Newton’s method) [33].

While these approaches may be conceptually straightforward, a number of subtleties often arise [22]. For instance, the choice of stopping criteria and the associated tolerances for both the nonlinear and linear solvers, the frequency of Jacobian evaluations, and the increment used in evaluating numerical Jacobians can all play a significant role in the success of an approach for a given problem [29, 25, 27, 8, 34]. Since the convergence of Newton’s method can be quite sensitive to the quality of initial guess, supplementing a Newton approach with strategies like a line search and backtracking has improved nonlinear solver performance for some problems [27]. Line-search techniques typically select a fraction of the full Newton correction to update the current iterate with the step chosen to insure that the updated solution represents a sufficient decrease in the nonlinear residual [18, 11]. For transient problems, methods commonly adjust the time step based on nonlinear solver performance. Specifically, in the case of poor nonlinear solver performance, the time step is often reduced [31, 33, 8]. This can improve the scaling of the Jacobian, since the reciprocal of the time step typically appears on the diagonal [15], and can improve the quality of the initial guess obtained from the solution at the previous time step [29]. Another strategy designed to address Newton approaches’ sensitivity to initial guess is a hybrid Newton-Picard algorithm which uses a Picard iteration initially until a certain level of convergence is reached and then switches to Newton’s method [29, 8].
Like the hybrid Newton-Picard algorithm, \( \Psi tc \) attempts to combine the rapid convergence of Newton’s method near the steady-state solution with a more robust iteration far from the solution. \( \Psi tc \) employs a time-stepping approach when the iteration is far from the steady state, and thus exploits the relationship between the nonlinear system for the steady-state problem and the initial value problem from which it is derived. The time-stepping algorithm used in \( \Psi tc \) has properties similar to forward Euler with simple heuristics, which makes it both unstable and inaccurate as a time-integration method for stiff ordinary differential equations (ODE’s). Nevertheless, \( \Psi tc \) can effectively maintain certain important qualities of the transient solution of some problems such as enforcing a CFL condition, and has been shown to be effective in reaching the steady state for models with discontinuous transient phenomena such as shocks. While RE does not exhibit shocks under physically realistic choices of parameters, it does exhibit sharp fronts due to the nonlinearities as well as steep gradients due to discontinuities in the porous media properties.

3 Approach

3.1 Formulation

RE can be formulated in a number of ways. We begin with an expression for the conservation of mass for the aqueous phase in an air-water system where the solid phase is assumed immobile, and interphase mass transfer is neglected. Combining this with the standard extension of Darcy’s law to variable satu-
rated conditions [26] gives

\[
\frac{\partial (\rho \theta)}{\partial t} = \nabla \cdot \rho K_e (\nabla \psi - \rho g_u) + f \quad \text{in } \Omega \times [0, T]
\]

(1)

with

\[
\begin{align*}
\varepsilon &= \varepsilon_0 e^{\beta (\rho - \rho_0)} \\
\rho &= \varepsilon / \varepsilon_0 \\
\psi &= \frac{p}{\varepsilon_0 \left\| g \right\|} \\
g_u &= \frac{g}{\left\| g \right\|} \\
K_s &= \frac{\varepsilon_0 \left\| g \right\| k_i}{\mu} \\
K_e &= k_r(\psi) K_s
\end{align*}
\]

(2)

Here \( p, \varepsilon, \) and \( \mu \) are the pressure, density, and viscosity of the aqueous phase, \( \theta \) is the volume fraction, and \( f \) is a source term for the aqueous phase. \( \varepsilon_0 \) is the density at \( p_0 \), \( \beta \) is the compressibility of the aqueous phase, \( \psi \) is the pressure head, and \( g \) is a vector accounting for the acceleration of gravity. \( K_e \) is the effective hydraulic conductivity, \( K_s \) is the saturated hydraulic conductivity, and \( k_i \) is the intrinsic permeability of the porous medium. \( \Omega \subset \mathbb{R}^2 \) is the spatial domain, and \([0, T]\) is the temporal domain.

The auxiliary conditions for eqn (1) are given by

\[
\begin{align*}
\psi &= \psi^b \quad \text{on } \Gamma_D, t \in [0, T] \\
u \cdot n &= u^b \quad \text{on } \Gamma_N, t \in [0, T] \\
\psi &= \psi^0 \quad \text{in } \Omega, t = 0
\end{align*}
\]

(3)

(4)

(5)

where

\[
u = -\rho K_e (\nabla \psi - \rho g_u)
\]

(6)
is the mass flux, $\psi^b$, and $u^b$ are boundary condition functions. $\psi^0$ is the initial condition, and $\mathbf{n}$ is the unit outward normal for $\Omega$. We have also set $\Gamma = \partial \Omega = \Gamma_D \cup \Gamma_N$ with $\Gamma_D \cap \Gamma_N = \emptyset$.

The steady-state form of eqn (1) is

$$-\nabla \cdot \rho K_e (\nabla \psi - \rho g_n) = f \quad \text{in } \Omega$$

with auxiliary data given by eqns (3) and (4) without temporal dependence.

We use non-hysteretic forms of the $p$-$S$-$k$ relations of van Genuchten [35] and Mualem [28] (VGM). For $\psi < 0$, these are given by

$$S_e = \frac{(\theta - \theta_r)}{(\theta_s - \theta_r)}$$  \hspace{1cm} (8)

$$S_e = [1 + (\alpha_v|\psi|)^{n_v}]^{-m_v}$$  \hspace{1cm} (9)

$$k_r = \sqrt{S_e \left[1 - (1 - S_e^{1/m_v})^m_v\right]^2}$$  \hspace{1cm} (10)

where $S_e$ is the effective saturation, $\theta_r$ is the residual volumetric water content, $\theta_s$ is the saturated volumetric water content, $\alpha_v$ is a parameter related to the mean pore size, $n_v$ is a parameter related to the uniformity of the soil pore-size distribution, and $m_v = 1 - 1/n_v$. For $\psi \geq 0$, the porous medium is fully saturated, and eqns (8)–(10) revert to

$$S_e = 1$$  \hspace{1cm} (11)

$$k_r = 1$$  \hspace{1cm} (12)

The pressure head form of RE is obtained from eqn (1) by applying the chain rule to the left hand side

$$A(\psi) \frac{\partial \psi}{\partial t} = \nabla \cdot \rho K_e (\nabla \psi - \rho g_n) + f$$
\[
A(\psi) = \theta \frac{\partial \rho}{\partial \psi} + \rho \frac{\partial \theta}{\partial \psi}
\]  

(13)

3.2 Spatial discretization

We consider a discretization of \( \Omega = [0, X_1] \times [0, X_2] \subset \mathbb{R}^2 \) into a regular, orthogonal grid with \( n_e = n_{x1} \cdot n_{x2} \) nodes with \( \Delta x_d = X_d / (n_{xd} - 1) \), for \( d = 1, 2 \). We apply a cell-centered finite difference approximation to the right hand side of eqn (1) and write for a cell \( \Omega_{ij} \) in the interior,

\[
\frac{\partial (\rho_{i,j} \theta_{i,j})}{\partial t} = -R_{i,j}
\]

(14)

\[
R_{i,j} = -O_{d,i,j} - f_{i,j}
\]

(15)

\( i = 2, \ldots, n_{x1} - 1, \quad j = 2, \ldots, n_{x2} - 1 \)

For a cell \( \Omega_{i,j} \) in the interior, the discrete spatial approximation is

\[
O_{d,i,j} = \frac{1}{\Delta x_d} \left[ \rho_{i+1/2,j} K_{e,i+1/2,j} \left( \frac{\psi_{i+1,j} - \psi_{i,j}}{\Delta x_1} - \rho_{i+1/2,j} g_{u1} \right) 
- \rho_{i-1/2,j} K_{e,i-1/2,j} \left( \frac{\psi_{i,j} - \psi_{i-1,j}}{\Delta x_1} - \rho_{i-1/2,j} g_{u1} \right) 
\right]
+ \frac{1}{\Delta x_2} \left[ \rho_{i,j+1/2} K_{e,i,j+1/2} \left( \frac{\psi_{i,j+1} - \psi_{i,j}}{\Delta x_2} - \rho_{i,j+1/2} g_{u2} \right) 
- \rho_{i,j-1/2} K_{e,i,j-1/2} \left( \frac{\psi_{i,j} - \psi_{i,j-1}}{\Delta x_2} - \rho_{i,j-1/2} g_{u2} \right) \right]
\]

(16)

where the subscripts in \( g_u = [g_{u1}, g_{u2}]^T \) indicate values for each coordinate, and quantities with a 1/2 subscript denote values estimated at cell interfaces.

For the interface values, we use a harmonic average for saturated hydraulic conductivity, the arithmetic average for density, and upwind the relative permeability

\[
\rho_{i+1/2,j} = \left[ \rho(\psi_{i+1,j}) + \rho(\psi_{i,j}) \right] / 2
\]

(17)
\[ K_{e,i+1/2,j} = k_{r,i+1/2,j} K_{s,i+1/2,j} \\
K_{s,i+1/2,j} = 2K_{s,i,j} K_{s,i+1,j}/(K_{s,i,j} + K_{s,i+1,j}) \]
\[ k_{r,i+1/2,j} = \begin{cases} 
  k_r(\psi_{i+1,j}) & \text{if } \frac{\psi_{i+1,j} - \psi_{i,j}}{\Delta x_i} > \rho_{i+1/2,j} g_{u1} \\
  k_r(\psi_{i,j}) & \text{otherwise} 
\end{cases} \]  

(18)

The corresponding terms along the other coordinate axis are defined symmetrically.

The physical boundaries are located at the nodes (cell centers), so specifying Dirichlet conditions in pressure or volume fraction is straightforward. For example, at a cell \( \Omega_{i,j} \subset \Gamma_P \), we set
\[ C(\psi_{i,j} - \psi_{i,j}^b) = 0, \]  

(19)

where \( C \) is a scaling factor that gives the boundary equation roughly the same scaling as the interior nodes.

For cells along \( \Gamma_N \), we use linear extrapolation to apply the flux at the exterior (artificial) cell boundary rather than the cell center. This approach allows us to use the same nodal equation at the boundary node as we do at the interior points for \( \Omega_{i,j} \subset \Gamma_N \). If, for example, \( \Omega_{1,j} \subset \Gamma_N \), we set
\[ u_{1,1/2,j} = -2u_{1,1,j} - u_{1,3/2,j} \]  

(20)

at the fictitious left cell boundary. Here \( \mathbf{u} = [u_1, u_2]^T \).

### 3.3 Temporal approximation

Since we consider both direct solution of the steady-state problem eqn (7) and integration of eqn (1) to steady state, we begin by applying the cell-centered
finite difference spatial approximation to eqn (13) in a method of lines (MOL) context

\[ A(\psi_{i,j}) \frac{\partial \psi_{i,j}}{\partial t} = -R_{i,j} \]  
\[ i = 2, \ldots, n_x - 1, \ j = 2, \ldots, n_y - 1 \]  

with the Dirichlet and Neumann boundary conditions given by eqns (19) and (20).

The semi-discrete system corresponding to eqn (21) can be written as a set of differential algebraic equations (DAE’s)

\[ \mathbf{F}(t, \mathbf{y}, \mathbf{y}') = 0 \]  

where \( \mathbf{F} \) represents a set of equations that depend on time \( t \), a set of dependent variables \( \mathbf{y} \), and a set of first-order derivatives with respect to time of these dependent variables, \( \mathbf{y}' \).

A variety of approaches can be used to integrate eqn (22). To illustrate the structure of the nonlinear system for transient problems, we use a backward Euler approximation to convert eqn (22) to a fully discrete system [13, 21].

\[ \mathbf{G}(t^{n+1}, \mathbf{y}^{n+1}, \frac{1}{\Delta t^{n+1}}(\mathbf{y}^{n+1} - \mathbf{y}^n)) = 0 \]  

3.3.1 Nonlinear solution for the transient problem

At each time level, a full time integration approach such as backward Euler must solve the nonlinear system eqn (23). A general Newton iteration for eqn (23) can be written

\[ \mathbf{J}^{n+1,s_n} \{ \Delta \mathbf{y}^{s_n+1} \} = - \{ \mathbf{G}^{n+1,s_n} \} \]  

15
where \( \{\Delta y^{s_n+1}\} = \{y^{n+1,s_n+1}\} - \{y^{n+1,s_n}\} \) and \( s_n \) is a nonlinear iteration index. The Jacobian, \( J \), is formed by differentiating eqn (23) with respect to \( y \). For eqn (21), we can write \( J \) as

\[
\begin{align*}
[J^{n+1,s_n}]_l,l &= \frac{1}{\Delta t^{n+1}} [A^{n+1,s_n}] + \left[ \frac{\partial (Ay')^{n+1,s_n}}{\partial y} \right] - \left[ \frac{\partial O_d^{n+1,s_n}}{\partial y} \right]
\end{align*}
\]

where \( A \) is diagonal with \( [A]_{l,l} = A(\psi_{i,j}) \), \( \partial Ay'/\partial y \) is also diagonal, and \( \partial O_d/\partial y \) will be banded with seven non-zero entries. Here, \( l \) is a global identifier corresponding to cell \( \Omega_{i,j} \) with, for example, \( l = (j-1)n_{x1} + i \) for \( i = 1, \ldots, n_{x1} \), \( j = 1, \ldots, n_{x2} \).

For unknowns along \( \Gamma_D \), \( J \) is simply (see eqn (19))

\[
[J^{n+1,s_n}]_{l,l} = C
\]

3.4 \( \Psi tc \) approximation

\( \Psi tc \) attempts to find a solution to eqn (7) by integrating eqn (21) to steady state. The approach is straightforward and can included in many existing steady-state or transient solvers with minor modifications. The fully discrete \( \Psi tc \) system can be obtained from eqn (21) by first applying a backward Euler time discretization as in eqn (23) and then using Newton’s method with \( y^n \) as the initial guess.

3.4.1 Solution for \( \Psi tc \) update

While applying multiple iterations of Newton’s method is possible, the form of \( \Psi tc \) which we present here performs only a single Newton update for eqn
The resulting equation for the $\psi tc$ iterate is

$$[J^n] \{\Delta y^{n+1}\} = - \{R^n\} = \{O_d(y^n)\} + \{f^n\}$$  \hspace{1cm} (27)

with $\{\Delta y^{n+1}\} = \{y^{n+1}\} - \{y^n\}$ since only one Newton iteration is performed and $\{y^{n+1,0}\} = \{y^n\}$. $J$ has the same form as eqn (25), but without a derivative of the accumulation term

$$[J^n] = \frac{1}{\Delta t^{n+1}} [A^n] - \left[ \frac{\partial O_d^n}{\partial y} \right]$$  \hspace{1cm} (28)

Note that when $\Delta t$ is small enough $[J^n] \approx \frac{1}{\Delta t^{n+1}} [A^n]$ and therefore

$$\Delta y^{n+1} \approx -\Delta t^{n+1} [A^n]^{-1} \{R^n\}$$  \hspace{1cm} (29)

which is the update corresponding to the forward Euler method applied to the ODE form of RE.

### 3.4.2 $\psi tc$ step selection

$\psi tc$ solves a series of problems of the form in eqn (27), while adapting the time step $\Delta t^{n+1}$ based on the intermediate solution’s behavior. There are a number of common strategies, for selecting $\Delta t^{n+1}$, including the switched evolution relaxation (SER) [23, 15, 11]

$$\Delta t^{n+1} = \Delta t^n \frac{\|R^{n-1}\|}{\|R^n\|}$$  \hspace{1cm} (30)

The temporal truncation error approach (TTE) attempts to control the time step based on the local temporal truncation error [23, 15, 11]. It chooses $\Delta t^{n+1}$
so that

\[
\left| \frac{(\Delta t^{n+1})^2}{2(1 + |y^n|)} \frac{\partial^2 y_l}{\partial t^2} (t^n) \right| \leq \tau
\]  

(31)

for each component of the solution \(y_l\) and some constant \(\tau\). Here, we estimate \(\partial^2 y_l / \partial t^2\) at \(t^n\) by [11]

\[
\frac{\partial^2 y_l}{\partial t^2} (t^n) \approx \frac{2}{\Delta t^n + \Delta t^{n-1}} \left[ \frac{y^n_l - y^{n-1}_l}{\Delta t^n} - \frac{y^{n-1}_l - y^{n-2}_l}{\Delta t^{n-1}} \right]
\]  

(32)

For both SER and TTE, we also enforce an upper bound \(\Delta t_{\text{max}}\) on the chosen time step.

3.5 Newton’s method for the steady-state problem

Following eqn (21), we can write a cell-centered finite difference spatial approximation of eqn (7) to solve the steady-state problem directly

\[
R_{i,j} = 0
\]  

\[
i = 2, \ldots, n_{x1} - 1, \quad j = 2, \ldots, n_{x2} - 1
\]  

(33)

with Dirichlet and Neumann boundary conditions given by eqns (19) and (20) without temporal dependence.

The Newton update for eqn (33) is

\[
[J^{s_n}] \{\Delta y^{s_{n+1}}\} = - \{R^{s_n}\}
\]

\[
[J] = \left[ \frac{\partial O_d}{\partial y} \right]
\]  

(34)
3.5.1 Globalization techniques

A number of globalization strategies are commonly used to address the sensitivity of Newton’s method to initial guess. The Armijo line-search technique scales the original Newton update by a factor chosen to take a step as close to the original update as possible while insuring a sufficient decrease in the nonlinear residual [18]. At iteration level \( s_n \) for eqn (34), the Armijo line search can be written

1. Solve eqn (34) for \( \Delta y^{s_n+1} \), and set \( \lambda = \lambda_+ = 1 \)
2. While \( \|R(y^{s_n} + \lambda \Delta y^{s_n+1})\| > (1 - \epsilon_s \lambda) \|R(y^{s_n})\| \)
   
   Choose \( \lambda_+ \)
   
   if \( \lambda_+ < \sigma_{\min} \lambda \), \( \lambda_+ = \sigma_{\min} \lambda \)
   
   else if \( \lambda_+ > \sigma_{\max} \lambda \), \( \lambda_+ = \sigma_{\max} \lambda \)
   
   \( \lambda = \lambda_+ \)
3. Set \( \{y^{s_n+1}\} = \{y^{s_n}\} + \lambda \{\Delta y^{s_n+1}\} \)

where \( \epsilon_s \) is a parameter controlling the amount of decrease required in the nonlinear residual and \( \lambda \) is the final scaling factor. Here, we chose \( \lambda_+ \) so that it minimized a three-point parabolic approximation of \( \|R(y^{s_n} + \lambda \Delta y^{s_n+1})\| = f(\lambda) \). Bounds on \( \lambda_+ \) are dictated by \( \sigma_{\min} \) and \( \sigma_{\max} \). The details of this approach can be found in Kelley [22]. For this work we set \( \sigma_{\min} = 0.1, \sigma_{\max} = 0.55, \) and \( \epsilon_s = 10^{-4} \). The line search can fail if \( \lambda \) becomes too small. In this case, the nonlinear solver is said to have failed due to line-search stagnation [11].

To avoid evaluation errors in the constitutive relations and to increase the robustness of the iterations, we also include a test to make sure that the solution is within broad, physically relevant bounds.
(1) Solve eqn (34) for $\Delta y^{s_n+1}$, and set $\lambda = 1$

(2) While $y_t^{s_n} + \lambda \Delta y_t^{s_n+1} \notin [y_{\min}, y_{\max}] \ \forall t$

$$\lambda = \lambda / 2$$

(3) Set $\{y^{s_n+1}\} = \{y^{s_n}\} + \lambda \{\Delta y^{s_n+1}\}$

In the numerical experiments presented below, we use an interval $y_{\min} = -100 \ [m], y_{\max} = 100 \ [m]$.

3.6 Picard iteration

Picard iteration has been used widely in the solution of RE [10, 19]. In brief, a Picard linearization of eqn (33) can be formulated as the Newton update in eqn (34) but with an approximate Jacobian in which the coefficient derivatives $\partial k_r / \partial \psi$ and $\partial k / \partial \psi$ are omitted from $\partial O_d / \partial y$ [29, 25]. The performance of Picard iteration and Newton approaches have been compared in several works [29, 25, 27].

In many cases, Newton’s method with line search has proven more robust than a straightforward application of Picard iteration [27]. However, a hybrid Newton-Picard algorithm has also been suggested for reducing the sensitivity of Newton’s method to the quality of the initial guess [29, 8]. This approach performs an initial number of iterations for eqn (34) using the Picard approximation for $J$ and then switches to a Newton update with the full Jacobian [29, 8]. The motivation for this approach is that the Picard iterations are, in general, cheaper than their Newton counterparts.

Ideally, the switch to Newton’s method is chosen so that (1) a minimal number
of Picard iterations are performed; and, (2) upon switching, $y^{s_n}$ is sufficiently close to the true solution that the asymptotic convergence rate for the Newton updates is realized. There are several possible criteria for determining the crossover from Picard to Newton iteration, including $\|\Delta y^{s_n}\| < \epsilon_c$ [8]. Alternatively, one can base the switch on sufficient decrease in the initial residual,

$$\|R^{s_n+1}\| < \epsilon_c\|R^0\|$$

(35)

We use eqn (35) in the numerical results presented below.

3.7 Linear system solution

We test five algorithms for solving the linear systems arising in the nonlinear iteration. As a direct solver, we use both the banded LU decomposition from LAPACK (LU-B) [1], as well as the implementation of LU from the SPOOLES package (LU-S) [2, 3] in PETSc [6, 7, 5]. We also test the iterative method BiCGstab using ILU preconditioning with zero fill from PETSc (BiCGstab-ILU), and a two-level hybrid additive Schwarz domain decomposition method (BiCGstab-HAS) [17].

4 Results

In the following sections we will present results of several numerical experiments and compare the behavior of Ψtc with Newton’s method and a hybrid Newton-Picard iteration. First we summarize the test problems on which the numerical experiments were carried out.
4.1 Test problems

4.1.1 Problem 1: infiltration example

The first test problem is a relatively simple one-dimensional example and provides a case where each of the methods should perform well. We consider each test case as a transient problem with a corresponding steady-state solution. The first example simulates infiltration in vertical domain \( \Omega = [0, X_1] \). Initial conditions for the infiltration are set to static equilibrium with the water table, located at the bottom of the domain. Constant Dirichlet boundary conditions are set at the top so that the steady-state solution contains both saturated and unsaturated regions. Table 1 summarizes the relevant physical parameters and auxiliary conditions.

4.1.2 Problem 2: hillslope example

The spatial domain for the second problem, \( \Omega = [0, X_1] \times [0, X_2] \), is illustrated in Figure 1. The temporal domain is \( t \in [0, T] \) with \( T \) chosen so that the solution is at steady state. The relevant physical parameters and auxiliary conditions for Problem 2 are given in Tables 2-4. The log of the saturated hydraulic conductivity is given in Figure 2. Note that the domain is rotated 45 degrees to simulate a simple hillslope. The domain has block-heterogeneous medium properties ranging from clay to sand.

The boundary and initial conditions are configured to reflect an impermeable bedrock at the \( X_2^- \) boundary, no flow at the \( X_1^+ \) boundary, and static, saturated equilibrium along \( X_1^- \). The surface boundary condition at \( X_2^+ \) is a nonlinear flux boundary condition that simulates infiltrating precipitation.
Table 1
Fluid and domain properties for Problem 1

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_{u_1}$</td>
<td>-1.0</td>
<td>[-]</td>
</tr>
<tr>
<td>$|\mathbf{g}|$</td>
<td>$7.321 \times 10^{10}$</td>
<td>[m/d$^2$]</td>
</tr>
<tr>
<td>$\varrho_0$</td>
<td>998.2</td>
<td>[kg/m$^3$]</td>
</tr>
<tr>
<td>$\beta$</td>
<td>$6.564 \times 10^{-20}$</td>
<td>[m $\cdot$ d/kg]</td>
</tr>
<tr>
<td>$p_0$</td>
<td>0</td>
<td>[kg/m $\cdot$ d]</td>
</tr>
<tr>
<td>$X_1$</td>
<td>10</td>
<td>[m]</td>
</tr>
<tr>
<td>$\psi^b(x_1 = 0)$</td>
<td>0</td>
<td>[m]</td>
</tr>
<tr>
<td>$\psi^b(x_1 = 10)$</td>
<td>-0.05</td>
<td>[m]</td>
</tr>
<tr>
<td>$\psi^0(x_1)$</td>
<td>$-x_1$</td>
<td>[m]</td>
</tr>
<tr>
<td>$n_v$</td>
<td>4.264</td>
<td>[-]</td>
</tr>
<tr>
<td>$\alpha_v$</td>
<td>5.470</td>
<td>[m$^{-1}$]</td>
</tr>
<tr>
<td>$K_s$</td>
<td>5.040</td>
<td>[m/d]</td>
</tr>
</tbody>
</table>

until the surface becomes saturated. After saturated conditions are reached at the surface, the outward normal flux increases to zero and becomes positive as the pressure rises above atmospheric pressure. This condition reflects a well-drained surface that permits very little ponding at the surface. The boundary conditions can be summarized as follows.

$$
\psi_{X_1^+} = x_2 
$$

(36)

$$
\phi_{X_1^-}^b = 0 
$$

(37)

$$
\psi_{X_2^-} = 0 
$$

(38)

$$
\phi_{X_2^+}^b = \begin{cases} 
    u^r, & \psi < 0 \\
    u^r + K_s^s \psi, & \psi > 0 
\end{cases} 
$$

(39)

where $u^r$ is the precipitation rate and $K_s^s$ is the saturated conductivity at the
surface. For the numerical experiments presented, \( K_s^* = 10 \text{ [m/d]} \)

The steady-state solution of the volume fraction is given in Figure 3. This solution was computed by integrating the transient problem to \( T = 1700 \text{ [d]} \) at which time \( \| R \|_\infty < 10^{-5} \). The solution was obtained using a DAE integrator with relative and absolute integration tolerances of \( 10^{-4} \) [20, 21]. The transient solution exhibits steep moving fronts throughout the domain while the equilibrium solution shown maintains a number of steep moisture gradients due to the layering of the media.

4.2 Numerical experiments

For both test problems, we performed several numerical experiments on a series of grids. On each grid, we ran Newton’s method with a quadratic line search (NILS), the Newton-Picard hybrid approach (PIH), as well as two \( \Psi tc \)
Table 2

Media distribution for Problem 2

<table>
<thead>
<tr>
<th>Medium Type</th>
<th>P1</th>
<th>P2</th>
<th>P1</th>
<th>P1</th>
<th>P1</th>
<th>P2</th>
<th>P2</th>
<th>P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left [m]</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>1.4</td>
<td>2.9</td>
<td>7.1</td>
<td>8.6</td>
<td>2.9</td>
</tr>
<tr>
<td>Bottom [m]</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.1</td>
<td>0.3</td>
<td>0.1</td>
</tr>
<tr>
<td>Right [m]</td>
<td>10</td>
<td>0.71</td>
<td>1.4</td>
<td>2.9</td>
<td>6.4</td>
<td>9.3</td>
<td>9.3</td>
<td>7.1</td>
</tr>
<tr>
<td>Top [m]</td>
<td>0.1</td>
<td>1</td>
<td>0.8</td>
<td>0.3</td>
<td>0.3</td>
<td>0.3</td>
<td>0.8</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 3

Media properties for Problem 2

<table>
<thead>
<tr>
<th>Medium Type</th>
<th>θ_s</th>
<th>θ_r</th>
<th>n_v</th>
<th>α_v [m^{-1}]</th>
<th>K_s [m/d]</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0.41</td>
<td>0.07749</td>
<td>2.090</td>
<td>0.244</td>
<td>1.10808×10^{-5}</td>
</tr>
<tr>
<td>P2</td>
<td>0.40</td>
<td>0.03120</td>
<td>4.264</td>
<td>5.470</td>
<td>5.04000×10^{0}</td>
</tr>
<tr>
<td>P3</td>
<td>0.39</td>
<td>0.03822</td>
<td>2.370</td>
<td>0.478</td>
<td>1.80100×10^{-3}</td>
</tr>
<tr>
<td>P4</td>
<td>0.39</td>
<td>0.02691</td>
<td>3.264</td>
<td>0.244</td>
<td>4.04000×10^{0}</td>
</tr>
</tbody>
</table>

methods, SER and TTE, until ∥R∥_2 < (∥R_0∥_2 + 1)10^{-5}. The initial guess was taken to be the initial conditions for the corresponding transient problem. For the NILS and PIH calculations, we enforced a maximum number of nonlinear iterations s_n ≤ N_{max} = 1000, while the maximum number of steps for the SER and TTE runs was n ≤ N_{max} = 5000. The maximum number of line searches allowed was 1000 and the sufficient decrease parameter for Newton line search was ε_s = 10^{-4}. The PIH approach used a value of ε_c = 10^{-2} in its switching strategy, eqn (35). The Ψtc methods used an initial time step
Table 4

Fluid and domain properties for Problem 2

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_{u_1}$</td>
<td>-0.7071</td>
<td>[-]</td>
</tr>
<tr>
<td>$g_{u_1}$</td>
<td>-0.7071</td>
<td>[-]</td>
</tr>
<tr>
<td>$|g|$</td>
<td>$7.321 \times 10^{10}$</td>
<td>[m/d$^2$]</td>
</tr>
<tr>
<td>$\varrho_0$</td>
<td>998.2</td>
<td>[kg/m$^3$]</td>
</tr>
<tr>
<td>$\beta$</td>
<td>$6.564 \times 10^{-20}$</td>
<td>[m $\cdot$ d/kg]</td>
</tr>
<tr>
<td>$p_0$</td>
<td>0</td>
<td>[kg/m $\cdot$ d]</td>
</tr>
<tr>
<td>$X_1$</td>
<td>10</td>
<td>[m]</td>
</tr>
<tr>
<td>$X_2$</td>
<td>1</td>
<td>[m]</td>
</tr>
<tr>
<td>$u^*$</td>
<td>-0.4</td>
<td>[m/d]</td>
</tr>
</tbody>
</table>

Fig. 2. $\log(K_s)$ for Problem 2
of $\Delta t^0 = 4.050 \times 10^{-5}$, a maximum time step $\Delta t_{\text{max}} = 10^5$, and the TTE algorithm used $\tau = 1.0$. The relevant parameters are summarized in Table 5.

The linear systems for Problem 1 were solved using the LU-B decomposition from LAPACK with each method. For Problem 2, we ran the numerical experiments for each of the four steady-state solution methods with the LU-S direct solver as well the two iterative methods (BiCGstab-ILU and BiCGstab-HAS). A relative residual test was used for BiCGstab with a tolerance of $10^{-6}$. The maximum number of linear iterations allowed was 2000 except where noted. The simulations were performed on a Pentium 4 (2.53 Ghz) workstation with 256 Mbytes of RAM running Redhat Linux 7.3. The elapsed time for the simulations was recorded using the ANSI C clock and time intrinsics.

The results for Problem 1 on spatial grids of size $n_e = 41-161$ are presented in
Table 5

Numerical methods summary

<table>
<thead>
<tr>
<th></th>
<th>NILS</th>
<th>PIH</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_{\text{max}}$</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>$\epsilon_s$</td>
<td>$10^{-4}$</td>
<td>-</td>
</tr>
<tr>
<td>$\epsilon_c$</td>
<td>-</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>TTE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N_{\text{max}}$</td>
<td>5000</td>
<td>5000</td>
</tr>
<tr>
<td>$\Delta t^0$</td>
<td>$4.050 \times 10^{-5}$</td>
<td>$4.050 \times 10^{-5}$</td>
</tr>
<tr>
<td>$\Delta t_{\text{max}}$</td>
<td>$10^5$</td>
<td>$10^5$</td>
</tr>
<tr>
<td>$\tau$</td>
<td>1.0</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 6. The labels are as follows:

- Jeval-Jacobian evaluation
- Feval-function evaluation
- Steps-attempted iterations for method
- NLI-nonlinear iteration
- NLF-nonlinear iteration failure
- LI-linear iteration
- LF-linear iteration failure
- Clock-wall clock time (seconds)
- X-failed to converge

The simulations for Problem 1 were small and involved a homogeneous medium. As a result, we expected the methods to have little difficulty in its solution. Both the NILS method and the PIH iteration performed well, even though the NILS method employed a number of line searches on each spatial grid.
Table 6
LU-B runs for Problem 1

<table>
<thead>
<tr>
<th></th>
<th>Jeval</th>
<th>Feval</th>
<th>Steps</th>
<th>NLI</th>
<th>NLF</th>
<th>LI</th>
<th>LF</th>
<th>LS</th>
<th>Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>$n_e = 41$</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>30</td>
<td>299</td>
<td>30</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>102</td>
<td>0.05</td>
</tr>
<tr>
<td>PIH</td>
<td>14</td>
<td>33</td>
<td>14</td>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.01</td>
</tr>
<tr>
<td>TTE</td>
<td>51</td>
<td>103</td>
<td>51</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.06</td>
</tr>
<tr>
<td>SER</td>
<td>83</td>
<td>167</td>
<td>83</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.04</td>
</tr>
<tr>
<td><strong>$n_e = 81$</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>26</td>
<td>253</td>
<td>26</td>
<td>26</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>88</td>
<td>0.04</td>
</tr>
<tr>
<td>PIH</td>
<td>15</td>
<td>35</td>
<td>15</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.03</td>
</tr>
<tr>
<td>TTE</td>
<td>104</td>
<td>209</td>
<td>104</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.14</td>
</tr>
<tr>
<td>SER</td>
<td>168</td>
<td>337</td>
<td>168</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.09</td>
</tr>
<tr>
<td><strong>$n_e = 161$</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>25</td>
<td>329</td>
<td>25</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>128</td>
<td>0.08</td>
</tr>
<tr>
<td>PIH</td>
<td>16</td>
<td>35</td>
<td>16</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.06</td>
</tr>
<tr>
<td>TTE</td>
<td>207</td>
<td>410</td>
<td>202</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.43</td>
</tr>
<tr>
<td>SER</td>
<td>353</td>
<td>705</td>
<td>351</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.33</td>
</tr>
</tbody>
</table>

PIH took roughly half as many steps as NILS and needed only one line search once it switched to the Newton iteration, indicating that the initial Picard iterations were more successful in advancing the intermediate iterates closer to the steady-state solution than Newton’s method with a line search alone. The $\Psi tc$ methods also converged to the correct solution, but required significantly more steps than either the NILS or PIH approaches. To illustrate the methods’ performance, Figure 4 shows the residual history for NILS and PIH. Both methods achieved quadratic convergence as they approached the root.
Problem 2 was more challenging than Problem 1 due to its dimensionality, heterogeneous medium, and the nonlinear boundary condition at the surface. Results of the numerical experiments with LU-S and BiCGstab are presented in Tables 7-9. The increased wall clock times, iteration counts, and line searches reflect the added difficulty of Problem 2. NILS converged for every linear solver and spatial grid in Tables 7-9. It required a similar number of nonlinear iterations and line searches for a particular grid, regardless of the linear solver used. Unlike NILS, PIH failed for the cases considered. With one exception where it failed in the initial linear solve, PIH did not reduce the original nonlinear residual sufficiently to satisfy eqn (35) and switch to NILS. In these cases, it exhausted the allowed number of nonlinear iterations.

Both the \Psi tc methods converged for each spatial grid and linear solver combination in Tables 7-9, with TTE consistently between 2 and 5 times faster than SER. The run times for NILS and TTE were similar for most of the simulations. NILS was more efficient with the LU-S solver, particularly for the $81 \times 81$ grid. On the other hand, TTE was more efficient for the BiCGstab calculations. The difference in run times increased for the HAS preconditioner, where TTE was 3 and 1.4 times faster than NILS on the $41 \times 41$ and $81 \times 81$ grids respectively.

The results also indicate a difference in the way NILS and TTE behaved as the spatial grid was refined. Namely, NILS required roughly the same number of iterations to converge regardless of spatial grid or linear solver, while the number of steps taken by TTE grew noticeably as $n_e$ increased. To investigate the performance of TTE, we ran an additional set of calculations where $\tau$ was set to $10^{-3}$ and scaled by $n_e$ for each grid. The corresponding $\tau$ values ranged from 0.121 on the $11 \times 11$ grid to 25.9 on a grid with $n_e = 161 \times 161$. As a
result, the TTE time step selection was slightly more conservative on coarser grids than the original choice of $\tau = 1$ and was more aggressive on the finer grids. Table 10 contains the results for TTE with LU-S and BiCGstab for $n_e = 11 \times 11$ to $n_e = 161 \times 161$ as well as the results for NILS on the finest grid.

With the use of a scaled $\tau$, Steps for TTE was roughly the same for $n_e = 11 \times 11$ to $n_e = 81 \times 81$ and each linear solver. There was still, however, an increase in the number of steps for the finest grid, particularly for the BiCGstab-ILU solver. TTE with the scaled $\tau$ took slightly longer than the original $\tau = 1$ calculations on the coarser grids, but reduced the total simulation time as the grids were refined. There was a corresponding improvement for TTE on the larger spatial grids with all three solvers. The computational effort for NILS and TTE with $\tau$ scaled was essentially the same for LU-S except for the finest grid, where NILS was 1.6 times faster. For BiCGstab-ILU and BiCGstab-HAS, TTE with $\tau$ scaled was twice as fast as the simulations with $\tau = 1$ on the $n_e = 81 \times 81$ grid.

We also note that BiCGstab-ILU did not perform well with the $n_e = 161 \times 161$ system. Simulations with each of the steady-state solution methods and BiCGstab-ILU typically required significantly less computational effort than BiCGstab-HAS for the coarser grids. However, for $n_e = 161 \times 161$ the total number of steps, linear iterations, and simulation time increased significantly for TTE while NILS failed after both 2000 and 10000 linear iterations.

As an example of the progress of the $\Psi tc$ iterations in comparison to the Newton and Newton-Picard iterations, we graphed the history of $\|R\|_2$ versus iteration for SER on the $n_e = 81 \times 81$ grid with the LU-S linear solver in Figure
5. As it neared the steady-state solution, the method’s convergence steepened. This reflects the convergence of the Newton iteration that $\Psi_{tc}$ reduces to near the root. Unlike the globalized Newton iteration, however, $\Psi_{tc}$ does not enforce a decreasing sequence of residuals, and Figure 5 shows increased residuals at some points in the SER calculations. The iteration history of TTE was more extreme, often oscillating wildly before nearing the steady-state solution. Figure 6 shows the TTE calculation with $\tau = 1$ on the $n_e = 81 \times 81$ grid. For some problems, it has been found that performing multiple sub-iterations for eqn (27) can improve convergence [23, 15]. However, we investigated this alternative for the SER and TTE updates for the second test problem and found no improvement for either approach.
Table 7
LU-S runs for Problem 2

<table>
<thead>
<tr>
<th></th>
<th>Jeval</th>
<th>Feval</th>
<th>Steps</th>
<th>NLI</th>
<th>NLF</th>
<th>LI</th>
<th>LF</th>
<th>LS</th>
<th>Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n_e = 11 \times 11$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>83</td>
<td>3901</td>
<td>83</td>
<td>83</td>
<td>0</td>
<td>83</td>
<td>0</td>
<td>1806</td>
<td>1.5</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>6173</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>1000</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>49</td>
<td>99</td>
<td>49</td>
<td>0</td>
<td>0</td>
<td>49</td>
<td>0</td>
<td>0</td>
<td>0.26</td>
</tr>
<tr>
<td>SER</td>
<td>145</td>
<td>291</td>
<td>145</td>
<td>0</td>
<td>0</td>
<td>145</td>
<td>0</td>
<td>0</td>
<td>1.65</td>
</tr>
<tr>
<td>$n_e = 21 \times 21$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>132</td>
<td>7057</td>
<td>132</td>
<td>132</td>
<td>0</td>
<td>132</td>
<td>0</td>
<td>3291</td>
<td>2.55</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>2239</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>1000</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>91</td>
<td>183</td>
<td>91</td>
<td>0</td>
<td>0</td>
<td>91</td>
<td>0</td>
<td>0</td>
<td>2.59</td>
</tr>
<tr>
<td>SER</td>
<td>379</td>
<td>759</td>
<td>379</td>
<td>0</td>
<td>0</td>
<td>379</td>
<td>0</td>
<td>0</td>
<td>7.5</td>
</tr>
<tr>
<td>$n_e = 41 \times 41$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>106</td>
<td>4725</td>
<td>106</td>
<td>106</td>
<td>0</td>
<td>106</td>
<td>0</td>
<td>2181</td>
<td>9.92</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>2885</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>1000</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>173</td>
<td>346</td>
<td>172</td>
<td>0</td>
<td>0</td>
<td>173</td>
<td>0</td>
<td>0</td>
<td>12.23</td>
</tr>
<tr>
<td>SER</td>
<td>866</td>
<td>1732</td>
<td>865</td>
<td>0</td>
<td>0</td>
<td>866</td>
<td>0</td>
<td>0</td>
<td>53.29</td>
</tr>
<tr>
<td>$n_e = 81 \times 81$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>149</td>
<td>7369</td>
<td>149</td>
<td>149</td>
<td>0</td>
<td>149</td>
<td>0</td>
<td>3412</td>
<td>56.44</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>3449</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>1000</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>450</td>
<td>888</td>
<td>437</td>
<td>0</td>
<td>0</td>
<td>450</td>
<td>0</td>
<td>0</td>
<td>164.43</td>
</tr>
<tr>
<td>SER</td>
<td>2068</td>
<td>4135</td>
<td>2066</td>
<td>0</td>
<td>0</td>
<td>2068</td>
<td>0</td>
<td>0</td>
<td>709.67</td>
</tr>
</tbody>
</table>

5 Discussion

We began with the goal of identifying effective methods for the steady-state solution of RE. To this end, we performed several numerical experiments for
Table 8

BiCGstab-ILU runs for Problem 2

<table>
<thead>
<tr>
<th></th>
<th>Jeval</th>
<th>Feval</th>
<th>Steps</th>
<th>NLI</th>
<th>NLF</th>
<th>LI</th>
<th>LF</th>
<th>LS</th>
<th>Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n_e = 11 \times 11 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>83</td>
<td>3901</td>
<td>83</td>
<td>83</td>
<td>0</td>
<td>1297</td>
<td>0</td>
<td>1806</td>
<td>1.58</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>6173</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>7371</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>49</td>
<td>99</td>
<td>49</td>
<td>0</td>
<td>0</td>
<td>210</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>SER</td>
<td>145</td>
<td>291</td>
<td>145</td>
<td>0</td>
<td>0</td>
<td>445</td>
<td>0</td>
<td>0</td>
<td>1.77</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n_e = 21 \times 21 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>134</td>
<td>7205</td>
<td>134</td>
<td>134</td>
<td>0</td>
<td>6895</td>
<td>0</td>
<td>3360</td>
<td>4.5</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>2239</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>19419</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>98</td>
<td>197</td>
<td>98</td>
<td>0</td>
<td>0</td>
<td>970</td>
<td>0</td>
<td>0</td>
<td>2.08</td>
</tr>
<tr>
<td>SER</td>
<td>378</td>
<td>757</td>
<td>378</td>
<td>0</td>
<td>0</td>
<td>2559</td>
<td>0</td>
<td>0</td>
<td>7.14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n_e = 41 \times 41 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>111</td>
<td>4863</td>
<td>111</td>
<td>111</td>
<td>0</td>
<td>11153</td>
<td>0</td>
<td>2239</td>
<td>17.01</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>2893</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>38981</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>177</td>
<td>354</td>
<td>176</td>
<td>0</td>
<td>0</td>
<td>2215</td>
<td>0</td>
<td>0</td>
<td>13.49</td>
</tr>
<tr>
<td>SER</td>
<td>865</td>
<td>1730</td>
<td>864</td>
<td>0</td>
<td>0</td>
<td>11693</td>
<td>0</td>
<td>0</td>
<td>54.44</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n_e = 81 \times 81 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>145</td>
<td>7127</td>
<td>145</td>
<td>145</td>
<td>0</td>
<td>31799</td>
<td>0</td>
<td>3301</td>
<td>148.57</td>
</tr>
<tr>
<td>PIH</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2000</td>
<td>1</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>424</td>
<td>847</td>
<td>422</td>
<td>0</td>
<td>0</td>
<td>7934</td>
<td>0</td>
<td>0</td>
<td>136.65</td>
</tr>
<tr>
<td>SER</td>
<td>2068</td>
<td>4135</td>
<td>2066</td>
<td>0</td>
<td>0</td>
<td>50243</td>
<td>0</td>
<td>0</td>
<td>604</td>
</tr>
</tbody>
</table>

Newton’s method with two globalization techniques (NILS and PIH) as well as two versions of \( \Psi tc \) (SER and TTE). Various observations can be made based on the results, which reflect a range of difficulty for the one and two-
Table 9

BiCGstab-HAS runs for Problem 2

<table>
<thead>
<tr>
<th></th>
<th>Jeval</th>
<th>Feval</th>
<th>Steps</th>
<th>NLI</th>
<th>NLF</th>
<th>LI</th>
<th>LF</th>
<th>LS</th>
<th>Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n_e = 11 \times 11 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>84</td>
<td>3959</td>
<td>84</td>
<td>84</td>
<td>0</td>
<td>4200</td>
<td>0</td>
<td>1833</td>
<td>1.13</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>6173</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>24663</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>49</td>
<td>99</td>
<td>49</td>
<td>0</td>
<td>0</td>
<td>425</td>
<td>0</td>
<td>0</td>
<td>0.38</td>
</tr>
<tr>
<td>SER</td>
<td>146</td>
<td>293</td>
<td>146</td>
<td>0</td>
<td>0</td>
<td>1076</td>
<td>0</td>
<td>0</td>
<td>1.99</td>
</tr>
<tr>
<td>( n_e = 21 \times 21 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>131</td>
<td>7059</td>
<td>131</td>
<td>131</td>
<td>0</td>
<td>46899</td>
<td>0</td>
<td>3293</td>
<td>26.3</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>2239</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>51097</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>88</td>
<td>177</td>
<td>88</td>
<td>0</td>
<td>0</td>
<td>1628</td>
<td>0</td>
<td>0</td>
<td>2.63</td>
</tr>
<tr>
<td>SER</td>
<td>379</td>
<td>759</td>
<td>379</td>
<td>0</td>
<td>0</td>
<td>5799</td>
<td>0</td>
<td>0</td>
<td>9.04</td>
</tr>
<tr>
<td>( n_e = 41 \times 41 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>110</td>
<td>4991</td>
<td>110</td>
<td>110</td>
<td>0</td>
<td>24471</td>
<td>0</td>
<td>2305</td>
<td>69.02</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>3019</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>74767</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>168</td>
<td>336</td>
<td>167</td>
<td>0</td>
<td>0</td>
<td>4814</td>
<td>0</td>
<td>0</td>
<td>23.93</td>
</tr>
<tr>
<td>SER</td>
<td>866</td>
<td>1732</td>
<td>865</td>
<td>0</td>
<td>0</td>
<td>26241</td>
<td>0</td>
<td>0</td>
<td>117.69</td>
</tr>
<tr>
<td>( n_e = 81 \times 81 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NILS</td>
<td>148</td>
<td>7315</td>
<td>148</td>
<td>148</td>
<td>0</td>
<td>30762</td>
<td>0</td>
<td>3389</td>
<td>395.15</td>
</tr>
<tr>
<td>PIH</td>
<td>1000</td>
<td>3449</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
<td>69639</td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>TTE</td>
<td>439</td>
<td>867</td>
<td>427</td>
<td>0</td>
<td>0</td>
<td>11964</td>
<td>0</td>
<td>0</td>
<td>288.42</td>
</tr>
<tr>
<td>SER</td>
<td>2068</td>
<td>4135</td>
<td>2066</td>
<td>0</td>
<td>0</td>
<td>78706</td>
<td>0</td>
<td>0</td>
<td>1377.61</td>
</tr>
</tbody>
</table>

dimensional variably saturated flow problems examined in this work.

The use of a line search or hybrid Picard iteration made Newton’s method more robust for Problem 1. PIH required roughly half the iterations of NILS,
Table 10

Runs with $\tau$ scaled by $n_e$ for Problem 2

<table>
<thead>
<tr>
<th>$n_e$</th>
<th>Jeval</th>
<th>Feval</th>
<th>Steps</th>
<th>NLI</th>
<th>NLF</th>
<th>LI</th>
<th>LF</th>
<th>LS</th>
<th>Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>LU-S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTE</td>
<td>11 × 11</td>
<td>114</td>
<td>229</td>
<td>114</td>
<td>0</td>
<td>0</td>
<td>114</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>21 × 21</td>
<td>127</td>
<td>255</td>
<td>127</td>
<td>0</td>
<td>0</td>
<td>127</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>41 × 41</td>
<td>125</td>
<td>250</td>
<td>124</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>81 × 81</td>
<td>146</td>
<td>281</td>
<td>134</td>
<td>0</td>
<td>0</td>
<td>146</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>161 × 161</td>
<td>248</td>
<td>458</td>
<td>209</td>
<td>0</td>
<td>0</td>
<td>248</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NILS</td>
<td>161 × 161</td>
<td>232</td>
<td>12961</td>
<td>232</td>
<td>232</td>
<td>0</td>
<td>232</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BiCGstab-ILU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTE</td>
<td>11 × 11</td>
<td>113</td>
<td>227</td>
<td>113</td>
<td>0</td>
<td>0</td>
<td>447</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>21 × 21</td>
<td>129</td>
<td>259</td>
<td>129</td>
<td>0</td>
<td>0</td>
<td>1132</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>41 × 41</td>
<td>130</td>
<td>260</td>
<td>129</td>
<td>0</td>
<td>0</td>
<td>1739</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>81 × 81</td>
<td>148</td>
<td>289</td>
<td>140</td>
<td>0</td>
<td>0</td>
<td>3474</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>161 × 161</td>
<td>806</td>
<td>1185</td>
<td>378</td>
<td>0</td>
<td>0</td>
<td>92232</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NILS</td>
<td>161 × 161</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>10000</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BiCGstab-HAS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTE</td>
<td>11 × 11</td>
<td>115</td>
<td>231</td>
<td>115</td>
<td>0</td>
<td>0</td>
<td>979</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>21 × 21</td>
<td>125</td>
<td>251</td>
<td>125</td>
<td>0</td>
<td>0</td>
<td>2173</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>41 × 41</td>
<td>131</td>
<td>262</td>
<td>130</td>
<td>0</td>
<td>0</td>
<td>3782</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>81 × 81</td>
<td>152</td>
<td>288</td>
<td>135</td>
<td>0</td>
<td>0</td>
<td>7192</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TTE</td>
<td>161 × 161</td>
<td>283</td>
<td>472</td>
<td>188</td>
<td>0</td>
<td>0</td>
<td>20806</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NILS</td>
<td>161 × 161</td>
<td>236</td>
<td>13153</td>
<td>236</td>
<td>236</td>
<td>0</td>
<td>70484</td>
<td>0</td>
<td>6130</td>
</tr>
</tbody>
</table>

36
while both it and NILS took considerably fewer steps than either SER or TTE to reach steady state. The first test problem was intended to represent behavior for mild problems since the boundary conditions were simple and
the conductivity was homogeneous. The resulting linear systems were also small and tridiagonal. Under these ideal conditions, NILS had little difficulty, and PIH was able to speed convergence to the steady-state solution. The Ψtc methods offered no benefit in this situation.

The methods’ relative performance changed though, as we moved to more realistic conditions. The controlling factor in their performance for Problem 2 was the difficulty associated with solving the resulting linear systems for each method. In addition to nonlinear boundary conditions, Problem 2 involved a two-dimensional, block-heterogeneous domain with medium properties corresponding to sand and clay. As a result, the linear systems for Problem 2 were more poorly scaled than in Problem 1. On the first step of the $81 \times 81$ grid the condition number of the NILS Jacobian was approximately $10^{19}$ and after 150 iterations was $10^9$ (condition numbers calculated using the dgbtrf/dgbcon routines from LAPACK).

Poor scaling of the Jacobians manifested itself in a number of ways, including the added work required by BiCGstab to converge with both preconditioners. In general, the difficulty of solving the linear systems can also translate into less accurate search directions as, for example, one is forced to use coarser linear solver tolerances to obtain results given computational limitations. The performance of NILS with the LU-S solver illustrates the impact of the Jacobian scaling on direct methods and the result of inaccurate search directions. The results in Table 7 indicate that NILS was the most efficient approach when LU-S was used for the LU decompositions. However, when we performed the tests with the LU solver from PETSc which does not include pivoting, the solution was less accurate, and NILS either exhausted the allowed nonlinear iterations or stagnated in the line search.
The PIH approach did not perform as it was intended for Problem 2 because
the initial Picard iterations did not adequately reduce the initial residual.
The convergence of PIH could have been improved by relaxing the switching
criterion in eqn (35) or enforcing a maximum number of Picard iterations,
so that it would have switched to the Newton updates before exhausting the
allowed iterations. However, this would be, in essence, just NILS and would
have masked the ineffectiveness of the initial Picard iterations for Problem 2.

In contrast, SER was robust but relatively inefficient. It converged for each of
the runs and linear solvers, but was significantly slower than TTE for most
cases. The two most efficient methods for Problem 2 were NILS and TTE.
Their relative efficiency was dictated largely by the computational expense
associated with the linear solvers. With both methods, the number of steps
required for a given spatial grid was similar using the LU-S and BiCGstab
solvers. However, the computational effort required changed significantly. Us-
ing a robust direct solver, NILS was three time faster than TTE with \( \tau = 1 \)
on the 81 \( \times \) 81 grid. However, it required four times as many total linear itera-
tions with BiCGstab-ILU and three times as many using BiCGstab-HAS. As a
result, it was 9\% slower using BiCGstab-ILU and 28\% slower using BiCGstab-
HAS.

The total number of iterations for NILS was fairly consistent on the spatial
grids considered. On the other hand, TTE with \( \tau = 1 \) did not scale as well
when the spatial grids were refined. The total number of steps required to
converge increased with \( n_e \), making TTE more expensive. Varying \( \tau \) based on
\( n_e \) led to more consistent results for TTE and improved its performance for
finer grids. NILS was still more efficient with a direct solver on the 161 \( \times \) 161
grid. On the other hand, TTE was between 2 and 3 times faster than NILS on
the finer grids using the iterative linear solvers and converged for BiCGstab-ILU on the $161 \times 161$ grid when NILS failed.

The value of a number of parameters like $\epsilon_c, \epsilon_s, \sigma_{\text{min}}, \sigma_{\text{max}}$ and the linear solver tolerances for BiCGstab effected each of the methods considered to some degree. The performance of TTE for different $\tau$ values demonstrates that the impact of parameter choices can be significant at times. Finding a successful configuration for a given method and problem usually requires some effort. While different choices of parameter values may lead to improved performance, the basic behavior of the methods was the same for the range of parameters we investigated. NILS was preferable for the simulations with a robust direct solver like LU-S, but BiCGstab performed better with the $\Psi_{\text{tc}}$ algorithms. TTE was more aggressive than SER and more efficient than NILS when BiCGstab was used with either ILU or HAS preconditioning.

As a reference point, we also compared the performance of the $\Psi_{\text{tc}}$ methods to time-accurate integration using a variable-order, variable step-size DAE integrator [20]. As might be expected, the full time integration was the most reliable approach for obtaining steady-state solutions, but the computational expense incurred was from 5 to 25 times greater than that of SER, which was similarly robust.

An initial guess closer to the final solution would have improved the performance of each of the methods considered. Since the goal of PIH and the $\Psi_{\text{tc}}$ methods is to approximate a Newton iteration near the steady-state solution, one can expect the advantage of NILS to increase as the initial guess better approximates the steady-state solution. However, we used static equilibrium as a reasonable initial guess, since we are interested in evaluating the meth-
ods for problems where good initial guesses for the steady-state solution are difficult to obtain.

The range of spatial grids presented for the numerical experiments was relatively coarse and the resulting linear systems were small to moderate in size. Still, the Jacobians were ill-conditioned for Problem 2, which proved to be a significant test for the direct and iterative linear solvers. For larger systems arising from realistic two and three-dimensional problems, we can only expect the difficulties associated with the linear systems to become more severe. Moreover, large simulations will often have to be solved in parallel to obtain results in a reasonable timeframe. For the numerical experiments presented here, NILS combined with the LU-S solver was the most efficient approach on each spatial grid. However, preconditioned Krylov methods and sparse direct solvers have their own advantages and disadvantages depending on the problem and architecture [12, 16]. A general comparison of their relative merits is beyond the scope of this paper.

6 Conclusions

Our numerical experiments for Ψtc approaches as well as Newton’s method with various globalization techniques lead us to the following conclusions and recommendations:

- For problems where use of a robust direct solver is feasible, Newton’s method with a line search is the most efficient approach for obtaining steady-state solutions to RE.
- Using an initial number of Picard iterations for Newton’s method with line
search can improve performance in some instances, but it does not necessarily lead to more robust performance for difficult problems.

- Inexact Newton methods with standard globalization techniques have particular difficulty when the Jacobian is poorly scaled due to factors such as heterogeneous conductivity fields.
- If Newton’s method fails or performs poorly for a given steady-state problem, it is worth examining a range of linear solver and line-search parameters before abandoning a Newton approach.
- {$\Psi$}tc is a relatively simple approach that can improve the efficiency and robustness of existing steady-state solvers for RE on difficult problems, particularly if iterative linear solvers are used.

Acknowledgments

The authors would like to thank Dr. T.-C. Yeh for several helpful suggestions.

The research of CEK and CTK was supported in part by National Science Foundation grants DMS-0070641, DMS-0112542, and Army Research Office grant DAAD19-02-1-0391. The efforts of MWF and CTM were supported by grants 5 P42 ES05948 from the National Institute of Environmental Health Sciences and DMS-0112653 from the National Science Foundation. Computational support was provided by the North Carolina Supercomputer Center.

References


[28] Y. Mualem. A new model for predicting the hydraulic conductivity of


