Advances in Water Resources 74 (2014) 185–195

Contents lists available at ScienceDirect

Advances in Water Resources

journal homepage: www.elsevier.com/locate/advwatres

Improved numerical solvers for implicit coupling of subsurface and overland flow ☆

D. Osei-Kuffuor [a,*,1], R.M. Maxwell [b,1], C.S. Woodward [a,1]

[a] Lawrence Livermore National Laboratory, PO Box 808, L-561, Livermore, CA 94551, USA
[b] Department of Geology and Geological Engineering and Integrated Ground Water Modeling Center, Colorado School of Mines, Golden, CO, USA

ARTICLE INFO

Article history:
Received 27 May 2014
Received in revised form 12 September 2014
Accepted 12 September 2014
Available online 28 September 2014

Keywords:
Overland flow
Preconditioning
Newton–Krylov
Implicit flow coupling
Parallel scalability

ABSTRACT

Due to complex dynamics inherent in the physical models, numerical formulation of subsurface and overland flow coupling can be challenging to solve. ParFlow is a subsurface flow code that utilizes a structured grid discretization in order to benefit from fast and efficient structured solvers. Implicit coupling between subsurface and overland flow modes in ParFlow is obtained by prescribing an overland boundary condition at the top surface of the computational domain. This form of implicit coupling leads to the activation and deactivation of the overland boundary condition during simulations where ponding or drying events occur. This results in a discontinuity in the discrete system that can be challenging to resolve. Furthermore, the coupling relies on unstructured connectivities between the subsurface and surface components of the discrete system, which makes it challenging to use structured solvers to effectively capture the dynamics of the coupled flow. We present a formulation of the discretized algebraic system that enables the use of an analytic form of the Jacobian for the Newton–Krylov solver, while preserving the structured properties of the discretization. An effective multigrid preconditioner is extracted from the analytic Jacobian and used to precondition the Jacobian linear system solver. We compare the performance of the new solver against one that uses a finite difference approximation to the Jacobian within the Newton–Krylov approach, previously used in the literature. Numerical results explores the effectiveness of using the analytic Jacobian for the Newton–Krylov solver, and highlights the performance of the new preconditioner and its cost. The results indicate that the new solver is robust and generally outperforms the solver that is based on the finite difference approximation to the Jacobian, for problems where the overland boundary condition is activated and deactivated during the simulation. A parallel weak scaling study highlights the efficiency of the new solver.

1. Introduction

The hydrologic cycle is usually simulated with discontinuous processes, each modeled separately. These processes are, however, naturally coupled and, in order to accurately model the complete cycle, data from one process is typically required and used as input to another process. This situation is particularly true for surface water and groundwater flow, where a dynamic interconnection between overland flow (often simulated by a nonlinear wave equation) and unsaturated subsurface flow (often simulated by a nonlinear diffusion equation) has been demonstrated (e.g. [1–3]). Subsurface flow and overland flow models constitute a significant contribution to the management of fresh water resources. An efficient numerical model that adequately captures the relationship between the two can produce results and insights that determine how renewable water resources are to be managed. In recent years, there has been interest in combining some of these processes in order to yield coupled regimes that better capture the dynamics and intrinsic relationships between the different processes [1,4–13].

The mathematical formulation of this multiphysics problem can be quite complicated leading to a challenging numerical solution. This computational expense is partly due to the inherent challenges of solving each individual process numerically, including nonlinearities, high anisotropy, and heterogeneity in the numerical coefficients of the computational models [14]. Moreover, the multiscale

☆ This work was supported by the Lawrence Livermore National Laboratory LDRD Program and by the United States Department of Energy SciDAC Program.
* Corresponding author.
E-mail addresses: oseikuffuor1@llnl.gov (D. Osei-Kuffuor), rmaxwell@mines.edu (R.M. Maxwell), cswoodward@llnl.gov (C.S. Woodward).
[1] The work of this author was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

http://dx.doi.org/10.1016/j.advwatres.2014.09.006
0309-1708/© 2014 Elsevier Ltd. All rights reserved.

nature of the coupling requires the careful development of numerical strategies to construct a suitable computational model. While a growing body of work presents solutions to these coupled systems, very little work has been done to accelerate the numerical solution. A range of approaches might be employed such as temporal adaptation of the surface and subsurface systems (e.g. [15]). Since the timescales can be similar, a fully coupled approach is often justified. However, efficient preconditioning of the linear system of equations that arises for this multiphysics system is essential to provide substantial efficiency gains (e.g. [14,16]). Furthermore, in order to perform large scale simulations or achieve a faster time to solution for the computational model, it is necessary to take advantage of parallel computing. High performance computing (HPC) plays an important role in climate modeling, and while it provides enhanced capabilities for the simulation of more realistic and complex models, few papers in the literature apply HPC technology in hydrology. Earlier work in [14], and later in [2], have demonstrated the need for efficient numerical solvers and preconditioners for subsurface flow that exhibit good scalability.

In this paper, we present a numerical solution strategy for coupling overland flow with subsurface flow for arbitrary geometries. Variably saturated subsurface flow is modeled by Richards' equation [14]. The equation is completed by imposing a Dirichlet-type boundary condition on the subsurface boundary, and a Neumann-type boundary condition on the surface boundary. Overland flow is typically governed by an approximation to the dynamic wave equations based on a coupling between the continuity equation, and the momentum equation [2,17]. Perhaps the simplest, and most widely applied, form of this approximation is the Kinematic wave approximation, where it is assumed that the local and convective acceleration and pressure gradient of the momentum equation do not impact the flow. In other words, the direction of flow is controlled by the slope of the surface topography. The coupling of overland flow to subsurface flow considered in this paper follows previous work by Kollet and Maxwell [2]. In [2], the coupling is handled by introducing the overland flow model as a boundary condition to the subsurface model. Overland flow is considered active when the subsurface cells are fully saturated, otherwise the imposed boundary condition on the surface still applies. This work presents an approach to precondition the implicit solution of the coupled Richards' and kinematic wave equations, demonstrating a significant improvement in solution time. The preconditioner uses the analytical Jacobian of the coupled system (developed for complex terrain) and a matrix–vector multiplication, which is also presented. Numerical results show that while each of these components individually does not improve solution time greater than the increased cost of their implementation, the two combined are needed for increased solution efficiency. The results also highlight the effect of preconditioning on the efficiency of the numerical solution scheme, for large-scale simulations of the coupled flow model.

This paper is organized as follows: In Section 2 we describe the mathematical formulation for the subsurface and surface models and present a discretization for the coupled model. In Section 3 we discuss the numerical solution strategy used to solve the discretized problem and present a new ordering of the discretized problem that enables the use of the analytical form of the Jacobian in the numerical solution scheme. We introduce some benchmark examples to test our solver in Section 4 and present some numerical results in Section 5. We summarize and conclude in Section 6.

2. Model and discretization

2.1. Mathematical model

We apply Richards' equation as our model for subsurface flow [18],

$$S_s S_w \frac{\partial \psi_p}{\partial t} + \phi \frac{\partial S_w(\psi_p)}{\partial t} = \nabla \cdot [k(x) k_r(\psi_p) \nabla(\psi_p - z)] + q_p, \tag{1}$$

where S_s is the specific storage coefficient, S_w is water saturation, ϕ is the porosity, ψ_p is the subsurface pressure head, $k(x)$ is the saturated hydraulic conductivity, $k_r(\psi_p)$ is the relative permeability, and q_p represents any water source or sink terms. The variable z represents the elevation or depth with respect to some datum or reference point. Here, the convention is that $z > 0$ if this elevation is below the reference point, and $z < 0$ if this elevation is above the reference point. The boundary conditions are assumed of Neumann type and represented as:

$$-[k(x) k_r(\psi_p) \nabla(\psi_p - z)] \cdot \mathbf{n} = q_n, \tag{2}$$

on the boundary Γ, and where $\mathbf{n}$ is the outward unit normal vector to Γ, and q_n represents some imposed flux.

Overland flow is typically modeled after a simplified form of the continuity equation, by assuming that flow is triggered by ponding due to saturation (ie. subsurface infiltration) [2]. This assumption is satisfied by the following equation:

$$\frac{\partial \psi_s}{\partial t} = -\nabla \cdot \vec{v}\psi_s + q_e(x) - q_r(x), \tag{3}$$

where ψ_s is the surface ponding depth, $\vec{v}$ is the depth averaged velocity vector, and $q_r(x)$ and $q_e(x)$ are the rainfall and exchange rates respectively.

The friction slope, S_f, is responsible for the kind of dynamic wave approximation that defines the overland flow model. This approximation is typically realized by coupling the continuity equation with the momentum equation [17] to give

$$\frac{\partial \psi_s}{\partial t} - (\vec{v} \cdot \nabla)\vec{v} - g\nabla\psi_s = g(S_0 - S_f), \tag{4}$$

where g is the acceleration due to gravity and S_0 is the bed slope. One simple approximation, commonly used in the literature and which we follow in this paper, is the kinematic wave approximation. Here, the local and convective acceleration and the pressure gradient terms are dropped from (4). The resulting approximation is $S_f \approx S_0$, and the overland flow is controlled by topographic effects only.

Following the work in [2], coupling between the overland and subsurface flow models is handled by prescribing the overland flow model as a boundary condition to the subsurface flow model. Fig. 1 shows an example of a problem domain representing a slope geometry. The solid circles represent the computational nodes at the cell centers of the subsurface domain. The grid-lined circles represent computational nodes at the top (or land) surface. These nodes represent where the coupling between the models takes place.

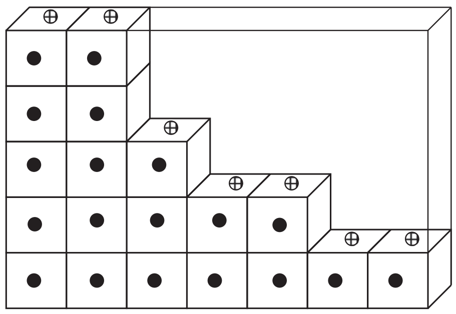


Fig. 1. An example of the domain grid for the subsurface-overland flow model, showing a slope geometry. Solid circles represent computational nodes at the cell centers of the subsurface domain. Grid-lined circles represent computational nodes at the top surface for the overland flow model.

In the coupled model, the Neumann boundary condition term is replaced by the exchange rate between the surface and subsurface, by setting $q_e = q_n$ (see [2]), and yielding

$$q_e(x) = \frac{\partial \psi_s}{\partial t} + \nabla \cdot \vec{v}\psi_s + q_r(x) = -\left[k(x)k_r(\psi_p)\nabla(\psi_p - z)\right]\mathbf{n} = q_n. \qquad (5)$$

2.2. Subsurface discretization

Following [14], discretization of (1) is achieved using a cell-centered finite difference method in space and an implicit backward Euler method in time. Harmonic averaging is applied for interface values of the hydraulic conductivity, $k(x)$, and upwinding for interface values of the relative permeability, $k_r(\psi_p)$. The computational grid is a tensor product with N_x, N_y, and N_z cells in the x, y, and z directions respectively. The computational nodes are defined at the cell centers, resulting in a total of $N_x \times N_y \times N_z$ unknowns.

Applying these discretizations results in a nonlinear system of equations which must be solved at each time step, given by

$$(S_sS_w)_{i,j,k}\frac{\psi^{n+1}_{p_{i,j,k}} - \psi^{n}_{p_{i,j,k}}}{\Delta t} + \phi_{i,j,k}\frac{S^{n+1}_w - S^n_w}{\Delta t} = \nabla \cdot U^{n+1} + q^{n+1}_{p_{i,j,k}}, \qquad (6)$$

where $S^{n+1}_w = S_w(\psi^{n+1}_{p_{i,j,k}})$, $U^{n+1} = [k(x)k_r(\psi^{n+1}_p)\nabla(\psi^{n+1}_p - z)]$, and the term $\nabla \cdot U^{n+1}$ is discretized as

$$\nabla \cdot U^{n+1} = \frac{U^x_{i+\frac{1}{2},j,k} - U^x_{i-\frac{1}{2},j,k}}{\Delta x} + \frac{U^y_{i,j+\frac{1}{2},k} - U^y_{i,j-\frac{1}{2},k}}{\Delta y} + \frac{U^z_{i,j,k+\frac{1}{2}} - U^z_{i,j,k-\frac{1}{2}}}{\Delta z}. \qquad (7)$$

Here,

$$U^x_{i+\frac{1}{2},j,k} = (k(x)k_r(\psi_p))_{i+\frac{1}{2},j,k}\frac{\psi^{n+1}_{p_{i+1,j,k}} - \psi^{n+1}_{p_{i,j,k}}}{\Delta x}, \qquad (8)$$

$$U^y_{i,j+\frac{1}{2},k} = (k(x)k_r(\psi_p))_{i,j+\frac{1}{2},k}\frac{\psi^{n+1}_{p_{i,j+1,k}} - \psi^{n+1}_{p_{i,j,k}}}{\Delta y}, \qquad (9)$$

and

$$U^z_{i,j,k+\frac{1}{2}} = (k(x)k_r(\psi_p))_{i,j,k+\frac{1}{2}}\frac{\left(\psi^{n+1}_{p_{i,j,k+1}} - z_{k+1}\right) - \left(\psi^{n+1}_{p_{i,j,k}} - z_k\right)}{\Delta z}. \qquad (10)$$

Thus, the nonlinear function to be evaluated in each cell is given by,

$$\begin{aligned} F^D_{i,j,k}(\psi^{n+1}_p) = {} & \Delta x\Delta y\Delta z\left[(S_sS_w)_{i,j,k}\left(\psi^{n+1}_{p_{i,j,k}} - \psi^n_{p_{i,j,k}}\right) + \phi_{i,j,k}\left(S^{n+1}_w - S^n_w\right)\right] \\ & - \Delta t\Delta y\Delta z\left[U^x_{i+\frac{1}{2},j,k} - U^x_{i-\frac{1}{2},j,k}\right] - \Delta t\Delta x\Delta z\left[U^y_{i,j+\frac{1}{2},k} - U^y_{i,j-\frac{1}{2},k}\right] \\ & - \Delta t\Delta x\Delta y\left[U^z_{i,j,k+\frac{1}{2}} - U^z_{i,j,k-\frac{1}{2}}\right] - \Delta t\Delta x\Delta y\Delta z q^{n+1}_{p_{i,j,k}} = 0 \end{aligned} \qquad (11)$$

The superscript D is used here to distinguish the nonlinear function as acting on the subsurface nodes only.

2.3. Discretization and coupling of overland flow terms

From the formulation in [2], overland flow should turn on when the top boundary cell is filled. In this case, the pressure within the cell and the pressure at the top surface (grid-lined node in Fig. 1) are assumed to be equal ($\psi_s = \psi_p$). Thus, there is no need to introduce any new unknowns at the top surface. The overland flow discretization can thus be defined on the nodes at the center of the top boundary cells.

To better understand how the coupling is handled, we rewrite (11) corresponding to the top boundary (surface) cell nodes:

$$\begin{aligned} F^{\widehat{D}}_{i,j,k}\left(\psi^{n+1}_p\right) = {} & \Delta x\Delta y\Delta z\left[(S_sS_w)_{i,j,k}\left(\psi^{n+1}_{p_{i,j,k}} - \psi^n_{p_{i,j,k}}\right) + \phi_{i,j,k}\left(S^{n+1}_w - S^n_w\right)\right] \\ & - \Delta t\Delta y\Delta z\left[U^x_{i+\frac{1}{2},j,k} - U^x_{i-\frac{1}{2},j,k}\right] - \Delta t\Delta x\Delta z\left[U^y_{i,j+\frac{1}{2},k} - U^y_{i,j-\frac{1}{2},k}\right] \\ & - \Delta t\Delta x\Delta y\left[q_n - U^z_{i,j,k-\frac{1}{2}}\right] - \Delta t\Delta x\Delta y\Delta z q^{n+1}_{p_{i,j,k}} = 0 \end{aligned} \qquad (12)$$

Here, the superscript $\widehat{D}$ is used to distinguish the nonlinear function as acting on the top surface cell nodes only. Notice that here we have replaced the flux across the top surface boundary, $U^z_{i,j,k+\frac{1}{2}}$ in (11), by the term q_n, as prescribed by the Neumann boundary condition in (2). In the coupled model, overland flow is introduced by replacing q_n in (12) by the exchange rate between the surface and subsurface, which is represented by $q_e(x)$ in (3) and (5) (see [2] for details). Below is a discretization of the overland flow contribution (from (3) and (5)) that replace q_n in (12),

$$\begin{aligned} f_{i,j,k}\left(\psi^{n+1}_p\right) = {} & \Delta x\Delta y\left(\psi^{n+1}_{p_{i,j,k}} - \psi^n_{p_{i,j,k}}\right) \\ & + \Delta t\Delta y\left[\left(\psi^{n+1}_p v^{n+1}_x\right)_{i+\frac{1}{2},j,k} - \left(\psi^{n+1}_p v^{n+1}_x\right)_{i-\frac{1}{2},j,k}\right] \\ & + \Delta t\Delta x\left[\left(\psi^{n+1}_p v^{n+1}_y\right)_{i,j+\frac{1}{2},k} - \left(\psi^{n+1}_p v^{n+1}_y\right)_{i,j-\frac{1}{2},k}\right] + \Delta t\Delta x\Delta y q_r \end{aligned} \qquad (13)$$

where $f_{i,j,k}(\psi^{n+1}_p)$ represents the contribution from the overland flow boundary condition to the nonlinear function evaluated in each control volume of the top boundary cell, and v_x and v_y are the x and y components of the velocity vector $\vec{v}$ respectively. These velocity components are computed by Manning's equation [2,17,19], which controls the depth-discharge relationship of the flow:

$$v_x = \frac{1}{n}\psi_p^{2/3}\sqrt{S_{fx}}, \qquad (14)$$

where n is the Manning's coefficient and S_{fx} is the friction slope in the x-direction.

The coefficients at the interface boundaries of (13) are computed by upwinding as follows (for the east and west faces):

$$\begin{aligned} \left(\psi^{n+1}_p v^{n+1}_x\right)_{i+\frac{1}{2},j,k} &= \max(0, q_{i,j,k}) - \max(-q_{i+1,j,k}, 0) \\ \left(\psi^{n+1}_p v^{n+1}_x\right)_{i-\frac{1}{2},j,k} &= \max(0, q_{i-1,j,k}) - \max(-q_{i,j,k}, 0) \end{aligned} \qquad (15)$$

and similarly for the north and south faces, and the $q_{i,j,k}$'s are computed as

$$q_{i,j,k} = (v_x\psi_p)_{i,j,k} = \frac{1}{n}\psi^{5/3}_{p_{i,j,k}}\sqrt{S_{fx}} \times sgn(-S_{fx}). \qquad (16)$$

Here, the $sgn()$ function simply returns the sign of the argument. Note that since we follow the kinematic wave approximation for overland flow in this work, the friction slope in (14) and (16) would be replaced by the bed slope.

3. Solvers

The numerical solution mechanism is based on a fully implicit parallel solution scheme. To improve scalability and parallel efficiency of the numerical solution scheme, the discretized nonlinear problem is solved by an inexact Newton–Krylov method [20,21].

Newton's method [22] is an iterative procedure for finding the solution to the root-finding problem

$$F(\psi) = 0,$$

by successively correcting an initial guess to the solution. The iterative scheme is derived from the truncated Taylor series expansion about the current iterate ψ^k, and written in the form

$$F(\psi^{k+1}) \approx F(\psi^k) + J(\psi^k)\delta^k + O\left(\|\delta^k\|^2\right), \qquad (17)$$

where $J(\psi^k) = F'(\psi^k)$ is the Jacobian of the nonlinear function evaluated at the point ψ^k, and $\delta^k = \psi^{k+1} - \psi^k$ is the correction or Newton update to the current solution. From a geometric stand point, (17) represents a linear approximation to $F(\psi)$ about the point ψ^k. As a result, δ^k represents a step in the descent direction toward the root of the linear function.

Ignoring the high order terms in (17), we have following linear system at each Newton step:

$$J(\psi^k)\delta^k = -F(\psi^k), \tag{18}$$

and the current solution is obtained with the update $\psi^{k+1} = \psi^k + \delta^k$.

Solving the linear system in (18) can be accomplished by the use of direct or iterative linear solver techniques. However, for large problem sizes, the use of direct methods is prohibitively expensive, in terms of both computational and memory costs, and iterative methods are preferred. The use of an iterative solution scheme to solve the linear system suggests that an exact solution to (18) is not required, and an approximate solution is enough to advance the current Newton step. The resulting scheme is known as the inexact Newton's method. In the most general approach, convergence of the linear system solve is assumed when the residual of the current linear iteration, $J(\psi^k)\delta^k + F(\psi^k)$, satisfies

$$\|J(\psi^k)\delta^k + F(\psi^k)\|_2 < \eta\|F(\psi^k)\|_2, \tag{19}$$

where the constant, $\eta < 1$, is known as the forcing term [23]. Careful choice of η is needed to ensure convergence of the Newton method without requiring too much precision in the linear solve.

3.1. Newton–Krylov method

For many applications, the construction and storage of an analytic Jacobian can be rather expensive, which can make the overall solution scheme inefficient. Alternative approaches for dealing with this issue have led to the development of Newton–Krylov methods [24]. The Newton–Krylov method is simply Newton's method with a Krylov solver for the linear system solve. Krylov methods, such as GMRES, do not require that the linear system matrix be assembled, only that the application of the matrix on a vector, in the form of a matrix–vector product, be known. This can be accomplished through finite-difference approximations of the directional derivative of *F*.

For the subsurface-surface flow coupling application on an arbitrary geometry, the variable dependence pattern, or stencil, associated with the surface flow regime does not generally coincide with that of the subsurface flow regime. This mismatch makes the structural representation of the coupled problem intractable with a single stencil, which can hinder numerical solution strategies that benefit from structure. Fig. 2 shows stencils associated with a node at the top boundary of a 2D geometry example. The figure depicts a hillslope or inclined geometry with the shaded region representing the (strictly) subsurface cells (or the region below the top boundary cells), and the unshaded region representing the top boundary cells where the coupling occurs. We differentiate between the nodes of the stencil by using solid circles to represent strictly subsurface nodes, and solid squares to represent the nodes at the top surface boundary. Furthermore, we have used dashed lines to indicate stencil connections that do not exist since we are on the boundary. This is helps to provide a more complete picture of the 5-point finite difference stencil or connectivity structure of the (2D) discretization. In Fig. 2(a), we show the standard 5-point finite difference stencil corresponding to the subsurface regime only. Here, the shaded square node represents a boundary node that only interacts with its west and south neighbors. In the coupled flow regime, this node would represent a computational node that also interacts with neighboring surface (or top boundary) nodes. Fig. 2(b) shows the stencil for the coupled regime. Notice the additional (shaded square) nodes, and their connections, that are introduced due to the interactions between the surface nodes. With these additional connections, the stencil representing the coupled flow no longer has a regular structure or pattern, particularly since the connection with leftmost top boundary node (the curved line in Fig. 2(b)) lies outside an appropriate stencil location. In previous work [2], Kollet and Maxwell used the finite-difference approach to get around this issue. However, with finite differences, oscillations around the boundary condition activation threshold ($\psi > 0$) severely slowed or prevented convergence in many cases. In this paper, we present an alternative strategy based on reordering the Jacobian into surface and subsurface components, to obtain an analytic representation without compromising the stencil structure of the discretization.

The Newton–Krylov method relies on knowledge of the linear Jacobian system in two ways. First, the matrix–vector multiply used to obtain the Krylov vectors for the linear system solve is required. Second, a preconditioner is generally required to accelerate convergence of the Krylov solver.

3.2. Finite-difference matrix–vector multiply

Since the linear system matrix is the Jacobian of the nonlinear function, the action of the matrix on a vector can be approximated by taking differences of the nonlinear function in the direction of that vector,

$$J(\psi^k)v \approx \frac{F(\psi^k + \epsilon v) - F(\psi^k)}{\epsilon}, \tag{20}$$

where ϵ is a perturbation parameter that controls the quality of the approximation. Typically, ϵ is chosen to be small, since a large value may lead to a poor approximation to the Jacobian. Furthermore, the Newton–Krylov scheme using the finite difference approximation to the Jacobian can exhibit locally quadratic convergence, typical

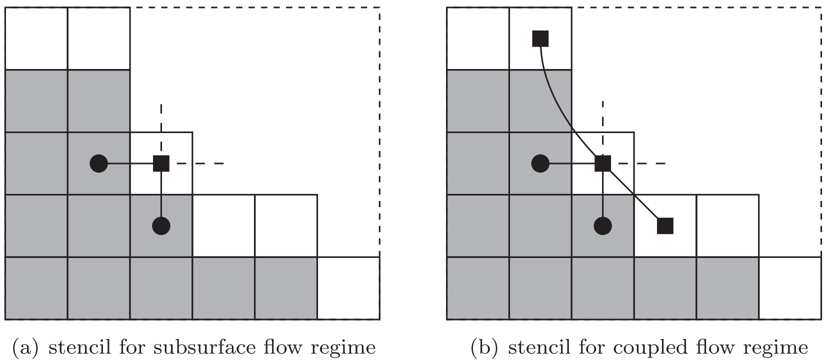

(a) stencil for subsurface flow regime
(b) stencil for coupled flow regime

Fig. 2. Stencils associated with a node at the top boundary of a 2D hillslope geometry example. The shaded grid cells represent the region below the top surface boundary and the unshaded grid cells represent the top surface boundary cells. The shaded circle nodes indicate subsurface nodes and the shaded square nodes indicate top surface boundary nodes.

of Newton's method, provided ϵ is small enough [14,25]. Nonetheless, care must be taken to ensure that ϵ is not too small, as that could result in an approximation that is corrupted by floating point and round-off error [21]. A good choice for ϵ that is commonly used in the literature comes from [24], and is denoted

$$\epsilon = \frac{\beta}{\|v\|_2} max\left\{|\psi^k \cdot v|, \|v\|_1\right\} sign(\psi^k \cdot v), \tag{21}$$

where the parameter β is related to the precision of the nonlinear function evaluation. When the nonlinear function can be evaluated with precision ϵ_o, then a good choice for β is $\beta = \sqrt{\epsilon_o}$. In many cases, ϵ_o is chosen to be the machine unit precision.

The conditional application of the overland boundary condition results in a discontinuity in the (coupled) nonlinear function, F. Thus, very small values of ϵ must be taken when overland flow is active. Furthermore, numerical roundoff errors could yield spurious pressure values for the boundary nodes, which can activate or deactivate the overland boundary condition. The result is that the finite difference approximation to the matrix–vector product relies on inconsistent behavior, and solver failures can occur due to the inability to resolve a descent direction for Newton's method.

3.3. Analytic matrix–vector multiply

It is possible to construct an analytic form of the Jacobian for the coupled subsurface-surface flow model. By using an analytic form of the Jacobian-vector product, the linear solver works with a consistent system even in the presence of discontinuities. Hence linear solver convergence will be more stable, and effectiveness of the Newton–Krylov solve will be improved.

Note that this does not necessarily eliminate any challenges the solver may encounter as a result of the discontinuity in the nonlinear function at the boundary. However, when these challenges are compounded by small errors in the finite difference approximation to the Jacobian matrix–vector product, the analytic form of the Jacobian can significantly improve the effectiveness and performance of the Newton–Krylov method.

Consider a 3D model of the coupled problem. The overland flow boundary condition, when active, acts only on the surface nodes. As a result, a 2D 5pt finite difference stencil is sufficient to represent the discretized overland flow boundary condition. However, as previously noted, the orientation of this stencil may be quite different from that of the stencil associated with the subsurface regime.

To obtain an analytic form of the global Jacobian that preserves the structure of the discretization for each of the flow models, the global problem is decoupled into surface and subsurface contributions (see Fig. 3). This decomposition is equivalent to a reordering of the global Jacobian matrix into the following block form:

$$J = \begin{pmatrix} C & E \\ F & B \end{pmatrix} \tag{22}$$

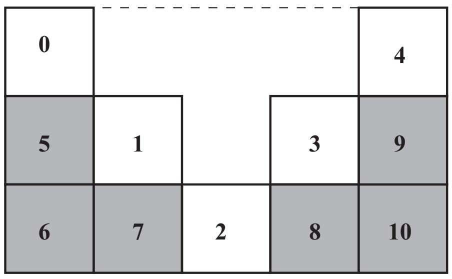


Fig. 3. A 2D example of the domain grid for the coupled flow model, showing the partitioning of the domain into surface nodes (labeled 0–4) and subsurface nodes (labeled 5–10).

where J is the analytic Jacobian, C corresponds to the surface-surface connections, B corresponds to the subsurface–subsurface interactions, E represents the surface–subsurface connections, and F represents the subsurface-surface connections. The stencil associated with the submatrix B remains a 7pt stencil in 3D (the original stencil associated with the subsurface problem grid). The stencils associated with the submatrices C, E, and F are 5pt stencils each. Notice that E and F are off-diagonal contributions only, and the orientation of their stencils coincide with that of B. Fig. 4 illustrates the connectivity pattern for the submatrices C, E, F and B, for the 2D example in Fig. 3.

The matrix–vector multiplication operation using the analytic Jacobian, for some vector x, takes the form:

$$\begin{pmatrix} C & E \\ F & B \end{pmatrix} \begin{pmatrix} x_C \\ x_B \end{pmatrix} = \begin{pmatrix} y_C \\ y_B \end{pmatrix}, \tag{23}$$

which is equivalent to:

$$\begin{aligned} Cx_C + Ex_B &= y_C \\ Fx_C + Bx_B &= y_B \end{aligned} \tag{24}$$

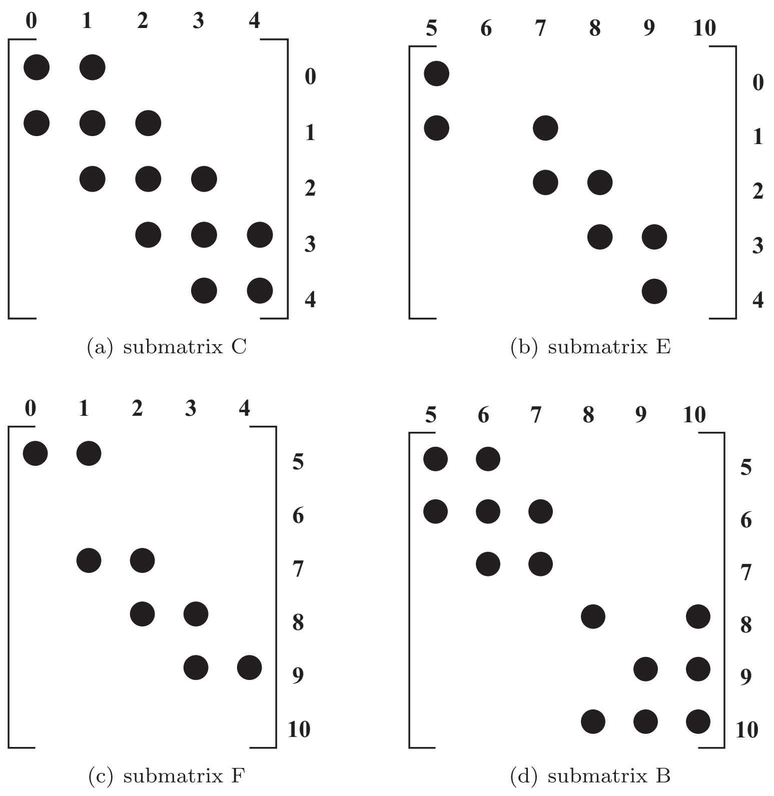
where x_C and x_B are the subvectors of x associated with the surface and subsurface nodes respectively, and y_C and y_B are the corresponding subvectors of the result y associated with surface and subsurface nodes respectively.

3.4. Preconditioning

For an efficient numerical solution by the Newton–Krylov method, a preconditioner may be needed to improve the convergence of the linear system solve. Multilevel methods are a class of preconditioners that are known to be very robust and efficient for solving linear systems. However, many of these methods rely on the algebraic properties of the associated coefficient matrix of the linear system, and do not explicitly exploit the structure of the discretization. For structured problems, it is generally more appropriate to use solvers that benefit from the inherent structure of the problem. Multigrid methods are known to be very scalable, particularly for structured problems, and have been successfully used as preconditioners for both subsurface and overland flow problems [2,14]. While the Newton–Krylov scheme does not require an explicit Jacobian matrix, the preconditioning operation by a multigrid method requires the solution to a linear system that adequately approximates the Jacobian system. Notice that the analytic Jacobian, in its reordered form (22), is not amenable to solution by structured solvers. This limitation arises because the rows of the submatrix C will generally contain nonzero entries at unstructured column positions (see Fig. 2(b) for an example of a case where this could happen). Preconditioning the coupled flow by a structured solver requires approximating the analytic Jacobian without compromising accuracy. One possibility is to consider splitting the analytic Jacobian matrix by block factorization as follows:

$$J = \begin{pmatrix} C & E \\ F & B \end{pmatrix} = \begin{pmatrix} I & 0 \\ FC^{-1} & I \end{pmatrix} \begin{pmatrix} C & E \\ 0 & S \end{pmatrix} \tag{25}$$

where $S = B - FC^{-1}E$ is the Schur complement matrix associated with C. The submatrix S is typically expensive to form, and hence is usually approximated. To approximate S, we note that the term $FC^{-1}E$ introduces surface–subsurface coupling into S. However, some of these entries fall at positions that are outside the nonzero pattern of B. Hence S can be approximated by constraining $FC^{-1}E$ to have a sparsity pattern that agrees with that of B. The preconditioner solve with the resulting approximation to the analytic Jacobian, can thus be done in a series of solves with the submatrices C and S, using a structured multigrid solver. A much less sophisticated


(a) submatrix C (b) submatrix E

(c) submatrix F (d) submatrix B

Fig. 4. Nonzero patterns of the submatrices associated with the 2D example in Fig. 3.

approximation to the analytic Jacobian is to use the Jacobian associated with the subsurface regime, as the preconditioner for the coupled problem. While this offers a cheaper alternative, it can lead to a less effective preconditioner when the simulation is dominated by overland flow. A variant of this approach, which works reasonably well, is to perturb the diagonal of the subsurface Jacobian to partially compensate for contributions from the overland flow terms. This approach was used in [2] to precondition the coupled problem in the absence of an analytic form of the Jacobian. However, note that without an analytic form of the Jacobian, the only accurate contribution from the overland flow discretization comes from the first term of (13).

In this work, we consider a preconditioner that takes into account contributions from the analytic form of the Jacobian. This preconditioner takes the form:

$$M = \begin{pmatrix} \widehat{C} & E \\ F & B \end{pmatrix}. \tag{26}$$

where the matrix $\widehat{C}$ has diagonal entries that match the diagonal of C. The off-diagonal entries of $\widehat{C}$ are chosen such that the entry $\hat{c}_{ij} = c_{ij}$ if nodes i and j are surface nodes at the same depth of the computational domain. Approximating C by $\widehat{C}$ in this way leads to a formulation where all the components for the preconditioner have a structured sparsity pattern, and thus, alleviates the need to perform the above block factorization in (25). The preconditioning operation is then accomplished by solving with M, using a multigrid solver.

From (15) it can be observed that only the upwinded terms will contribute non-zero entries to the nonlinear function in (13). Thus, each row in C will include contributions from the overland flow terms on the diagonal entry, and the off-diagonal entries that correspond to the direction of flow. All other contributions from overland flow will be zero. These upwinded terms typically yield a diagonally dominant contribution to the Jacobian.

4. Test problems

We consider three different types of problems for our numerical experiments. These problems are all well-known and considered benchmark examples in the literature [26].

4.1. Tilted V-Catchment example

This example simulates flow over a domain defined by two inclined planes connected by a sloping channel [2,7,27], see Fig. 5. In this example, we assume that the subsurface is saturated so that the problem simulates overland flow only. The simulation covers 100 min of rain, with a rainfall rate of 1.8×10^{-4} m/min, followed by 200 min of recession and evaporation at a rate of 1.0×10^{-6} m/min. We consider two cases with different slopes for the inclined planes and channel:

1. Slopes of 5% in the x-direction and 2% in the y-direction for the inclined planes, and slopes of 0.0% in the x-direction and 2% in the y-direction for the channel.
2. Slopes of 0.5% in the x-direction and 0.2% in the y-direction for the inclined planes, and slopes of 0.0% in the x-direction and 0.2% in the y-direction for the channel.

4.2. Saturation excess

In this example, the simulation models flow over a hill-slope. The computational domain covers a region 400 m long, 320 m wide, and 1 m thick, with a 5% slope in the x-direction, see Fig. 6. The initial water table is set to a depth of 1 m, and the

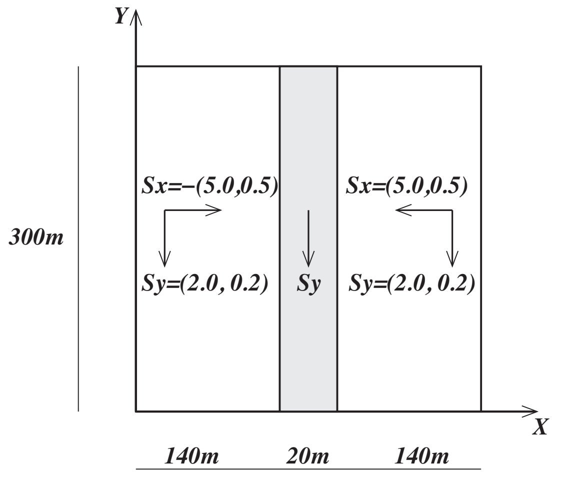


Fig. 5. Tilted V-catchment example. Case 1: Slopes of 5% in the x-direction and 2% in the y-direction for the inclined planes, and slopes of 0.0% in the x-direction and 2% in the y-direction for the channel. Case 2: Slopes of 0.5% in the x-direction and 0.2% in the y-direction for the inclined planes, and slopes of 0.0% in the x-direction and 0.2% in the y-direction for the channel.

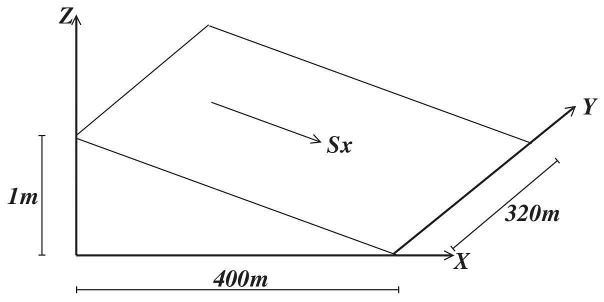


Fig. 6. Hill-slope geometry. Saturation Excess example: slope in x-direction $S_x = 5.0\%$. Infiltration Excess example: slope in x-direction $S_x = 0.5\%$.

simulation covers 200 min of rain at 3.3×10^{-4} m/min, followed by 100 min of recession.

4.3. Infiltration excess

This problem uses the hill-slope geometry as in the saturation excess example, and illustrated in Fig. 6. The slope is set to 0.5% in the x-direction, and the simulation models Hortonian runoff generated by 180 min of rain at a rate of 3.3×10^{-4} m/min, followed by another 120 min of recession. The initial water table is set to a depth of 0.5 m.

The following table details some of the parameters used for the simulation of the different examples. Note that the initial state is assumed to be in hydrostatic equilibrium (see Table 1).

5. Numerical results

The numerical simulations were performed using the ParFlow code [28], originally developed at the Lawrence Livermore National Laboratory. Current development efforts in ParFlow involves a community of collaborators working at the Department of Geology and Geologic Engineering, Colorado School of Mines; the Meteorological Institute, Bonn University, and the Center for Applied Scientific Computing, Lawrence Livermore National Laboratory. ParFlow combines physics, solvers, and parallelism for the high-performance solution of subsurface flow problems. ParFlow utilizes a globalized inexact Newton nonlinear solver within KINSOL [14], and preconditioned Krylov linear solvers with multigrid as the preconditioner. It has been modified to include overland flow coupling using the overland flow boundary condition technique presented in [2]. In the tests that follow, we use GMRES as the Krylov solver, preconditioned with the multigrid solver PFMG from HYPRE [29]. In all the test cases, the parameter ϵ, used in the finite difference matrix–vector multiplication in (20) is set to 1.0×10^{-8}. The nonlinear solver utilizes a backtracking linesearch algorithm to select the appropriate step length for each Newton iteration. The convergence tolerance for the nonlinear solver is set to 1.0×10^{-9}, and we do a maximum of 20 Newton iterations for each time step. When this maximum was achieved, the time step was decreased by a factor of 2, and the step restarted. We set a maximum of 10 coarse grid levels for the multigrid preconditioner used by the Krylov solver, and for each linear iteration, the multigrid preconditioner does 1 iteration from the finest level to coarsest level and then back to the finest level (v-cycle). We set the maximum number of linear iterations to 500, which is large enough to enable comparison of the different preconditioners.

In the numerical results that follow, we wish to highlight two main points. First, we are interested in a comparison of the results for the different matrix–vector products. That is, we compare simulation results using the analytic Jacobian for the matrix–vector product for the Newton–Krylov solver, against that of the finite

Table 1
Parameter values for different test cases.

Parameter (units)	V-Catchment	Saturation excess	Infiltration excess
Horizontal Mesh			
$\Delta x = \Delta y$ (m)	10	10	5
Vertical Mesh			
Δz (m)	0.05	0.05	0.05
Initial water table (m)	0.0	1.0	0.5
Porosity (-)	0.4	0.4	0.4
Saturated Hydraulic	6.94×10^{-4}(inclined planes)	6.94×10^{-4}	6.94×10^{-6}
Conductivity (m/min)	0.0 (channel)		
Mannings ($\text{m}^{-1/3}$ min)	2.5×10^{-4}(inclined planes)	3.31×10^{-3}	3.31×10^{-3}
	2.5×10^{-3} (channel)		
Slope x (%)	0.5, 5 (inclined planes)	5	0.5
	0 (channel)		
Slope y (%)	0.2, 2 (inclined planes)	0	0
	0.2, 2 (channel)		
Rainfall rate (m/min)	1.8×10^{-4}	3.3×10^{-4}	3.3×10^{-4}
Evaporation rate (m/min)	1.0×10^{-6}	0.0	0.0
Van Genuchten Parameters			
α (cm^{-1})	1.0	1.0	1.0
n (-)	2.0	2.0	2.0

difference approach. In the tables summarizing the results, we denote the use of the analytic Jacobian by ‘analytic’, and the use of the finite difference approximation by ‘FD’. Second, we wish to compare the performance of the multigrid preconditioner using the matrix in (26) versus the preconditioner used in [2]. As previously mentioned, the preconditioner used for the numerical simulations in [2] is based on modifying the subsurface Jacobian matrix to account for overland flow terms, in the absence of an analytic form of the Jacobian. In ParFlow, this modification is realized by updating the diagonal of the subsurface Jacobian to include contributions from the first and last terms of the discretized overland flow boundary condition in (13). Notice that this modification does not include contributions from the upwinded terms from the overland flow boundary condition in (13). However, with the new analytic representation of the Jacobian, the exact diagonal contributions from the overland flow boundary condition can be used as the modification. In the table of results, we distinguish between the different preconditioners by denoting the preconditioner in [2] as ‘Default PC’, and the new preconditioner based on (26) as ‘New PC’.

All runs to generate the following results were performed on an Intel Xeon (E5-2670) Linux cluster with 1296 nodes and 16 cores per node, and with 32 gigabytes of memory per node, at the Livermore Computing Center of the Lawrence Livermore National Laboratory. Numerical results for the sequential runs were obtained using a single core of this machine.

5.1. V-Catchment example

Table 2 shows the results for the V-Catchment example. Here, using the finite difference approach for the matrix–vector multiplication for the Newton–Krylov solver failed to converge. However, using the analytic Jacobian for the matrix–vector multiplication successfully solved the problem. The results also indicate some benefit in using the analytic Jacobian to define the matrix for the preconditioner. This benefit is characterized by fewer linear and nonlinear iterations, and a slight improvement in the total computational time, compared to using the default preconditioner. We also ran the easier test case with slopes of 0.5% and 0.2% as described in the second test of Section 4.1, and the results were very similar.

5.2. Saturation excess example

Table 3 presents the results for the saturation excess example. In this example, each of the combinations for the matrix–vector multiply and the preconditioner was successful in solving the problem. However, the results show a significant improvement in terms of the total number of linear and nonlinear iterations, and the total computational time, when the analytic Jacobian is used for the matrix–vector product, compared to the finite difference approach. Furthermore, we observe some backtracking for the nonlinear solver as a result of its inability to find an appropriate Newton step when the finite difference approach is used.

Note that the surface cells do not all get saturated at the same time. Thus, when the analytic form of the Jacobian is used, it is possible to have some of the rows of the Jacobian matrix with overland flow terms added in, and some without any contributions from overland flow. This inclusion of non-symmetric terms from the overland flow contribution (due to upwinding) into the analytic Jacobian, may not have been accounted for in the finite difference approximation to the matrix–vector product. In other words, the submatrix $\widehat{C}$ in (26) includes non-symmetric terms that may be inconsistent with the finite difference approximation to the Jacobian matrix–vector product. The result is that a preconditioner that is derived from the analytic Jacobian matrix is also inconsistent with the finite difference evaluation of the Jacobian for the matrix–vector product, which ultimately leads to solver failure. In this case, dropping the non-symmetric overland flow contribution to the analytic Jacobian used for the preconditioner (that is, the off-diagonal overland flow contribution to $\widehat{C}$ in (26)), can help with solver convergence. This situation explains the result in Table 3 where the solver fails for the combination of the finite difference form of the matrix–vector product with a preconditioner derived from the analytic Jacobian. Here, excluding the non-symmetric terms (due to upwinding) from the off-diagonals of the analytic Jacobian used for the preconditioner, successfully solves the problem in ≈ 80 s; using 203 nonlinear iterations with a total of 4705 linear iterations and backtracking 11 times.

Table 2
Results for different solver combinations for the V-Catchment example with slopes 5 and 2. $\mathbb{F}$ indicates KINSOL convergence failure.

Statistics	FD Default PC	FD New PC	Analytic Default PC	Analytic New PC
Nonlin. iters	$\mathbb{F}$	$\mathbb{F}$	115	113
Linear iters	–	–	3271	842
Backtracks	–	–	0	0
Time (s)	–	–	44.40	14.18

Table 3
Results for different solver combinations for the saturation excess example. An ($\mathbb{F}$) indicates KINSOL convergence failure and the ‘>’ sign indicates that the solution may require more steps or take more time to converge for this preconditioner.

Statistics	FD Default PC	FD New PC	Analytic Default PC	Analytic New PC
Nonlin. iters	187	$> 126(\mathbb{F})$	128	125
Linear iters	4297	> 22422	1233	783
Backtracks	18	> 207	0	0
Time (s)	87.85	> 842.16	20.7	15.52

5.3. Infiltration excess example

Table 4 shows results for the infiltration excess problem. In this example, the top cells are fully saturated before the entire subsurface is saturated. In other words, overland flow is activated prior to the subsurface cells being saturated. As a result, the simulation is dominated by overland flow of ponded water, which leads to a situation where the finite difference matrix–vector product is consistent with the matrix–vector product using the analytic form of the Jacobian. This consistency is seen in the results where, for the same preconditioner, the FD approach and the Analytic approach give similar results in terms of the number of linear and nonlinear iterations to convergence. However, due to the more expensive cost of computing the analytic Jacobian for the matrix–vector product, the FD approach yields a faster time to solution compared to the Analytic approach. Considering each of the approaches for the matrix–vector product separately, the results also indicate significant differences in solver performance based on the choice of the preconditioner. Clearly, including the overland flow terms into the preconditioner helps in the convergence of the Newton iterations. This is even more obvious when we consider the number of linear iterations required over the entire simulation, and the overall computational time. For each case of the matrix–vector product, we observe about a 70% improvement in the number of linear iterations, and a roughly 60% faster in time to solution, when the new preconditioner is used as opposed to the default preconditioner.

Table 5 shows results for a simulation of the infiltration excess problem spanning a 5 h time period, with 30 min of rainfall at a rainfall rate of 8.3×10^{-4} m/min. This example studies what happens with the solver after some initial surface ponding has occurred, leaving enough time for recession and subsurface

Table 4
Results for different solver combinations for the infiltration excess example. Hortonian runoff generated by 180 min of rain at 3.3×10^{-4} m/min and 120 min of recession.

Statistics	FD Default PC	FD New PC	Analytic Default PC	Analytic New PC
Nonlin. iters	113	109	116	112
Linear iters	814	334	824	340
Backtracks	0	0	0	0
Time (s)	66.43	40.27	101.10	57.48

Table 5
Results for different solver combinations for the infiltration excess example. Hortonian runoff generated by 30 min of heavy rainfall at 8.3×10^{-4} m/min and 270 min of recession. Long recession periods lead to subsurface infiltration of ponded water and drying events on parts of the surface.

Statistics	FD Default PC	FD New PC	Analytic Default PC	Analytic New PC
Nonlin. iters	168	157	116	117
Linear iters	1334	504	594	324
Backtracks	170	115	0	0
Time (s)	192.00	122.64	79.84	56.57

infiltration of the ponded water. The results highlight the difficulty of the finite difference approach to solve the problem. This difficulty may be attributed to the fact that the subsurface infiltration and recession leads to drying events on parts of the surface (i.e. overland boundary condition switches off in a non-uniform way). As a result, the finite difference evaluation of the Jacobian can result in an inconsistent Newton step, causing the solver to backtrack several times. Using the analytic Jacobian avoids this inconsistency.

The above results are quite interesting, and highlight the effect of the on/off switching of the overland boundary condition on the numerical solution scheme. When the simulation is such that boundary condition switching is minimal, as in the first problem (Table 4), the finite difference approach does not struggle much, and solves the problem quickly with the New PC option. On the other hand, when the simulation involves sufficient and possibly non-uniform overland boundary condition switching; as in the second problem (Table 5) and the saturation excess and v-catchment examples; the finite difference approach struggles, and the use of the analytic Jacobian becomes necessary.

5.4. Parallel scaling results

In the next example, we investigate the parallel efficiency of using the analytic Jacobian for the Newton–Krylov solver. We perform a weak scaling test of 150,000 unknowns per problem on an example with hard rainfall-runoff down a hill slope as in Fig. 6. We consider a slope of 0.05% in the x-direction, and the simulation covers 60 min of rain at a rate of 1.33×10^{-4} m/min, followed by 60 min of recession. The porosity of the domain is 0.05 and the saturated hydraulic conductivity is set to 6.94×10^{-6} m/min. Manning's roughness coefficient is set to 3.312×10^{-4} and the Van Genuchten parameters for relative permeability and saturation are $\alpha = 1.0$ and $n = 3.0$. The problem on each subdomain covers a region of $150\,\text{m} \times 100\,\text{m} \times 0.5\,\text{m}$, with a mesh spacing of $1.0, 1.0$, and 0.05 in the x, y, and z-directions respectively. The water table is set to the bottom of the domain, so that the top layer is initially dry. The choice of the rainfall rate and the saturated hydraulic conductivity means that the top layer is saturated and overland flow is activated before the subsurface is fully saturated, as in the excess infiltration example. We scale the problem from 1 to 1024 processors, using even powers of 2 and measure the weak scaling and the parallel efficiency of the Newton–Krylov solver. In addition to the total solver performance, we are also interested in the performance of the inexact Newton solver (KINSOL) and the associated Krylov solver preconditioned with PFMG.

Fig. 7 shows the results from the weak scaling tests using both the default preconditioning technique and the new preconditioning technique, on the same problem. In both cases, we use the analytic Jacobian for the matrix–vector product for the Newton–Krylov solver. The results indicate good scalability for the solver in general, and both preconditioners appear to scale well as shown in the PFMG performance results. We do see a slight growth in the total time in the case where the new preconditioner is used. This may be attributed to the additional cost in computing the overland flow terms for the analytic Jacobian that go into the preconditioner. Nonetheless, the results indicate a significant improvement of roughly 50% in Wall clock time, when this new preconditioner is used instead of the default preconditioner. To see the effect of the preconditioners on the efficiency of the solver, we measured the scaled parallel efficiency of the resulting solver performance. The efficiency of a parallel weak scaling study is defined as the ratio of the time taken to solve a problem of size N on a single processing element, to the time taken to solve the same problem of size $p \times N$ on p processing elements. Fig. 8 shows the results. As expected, the results show that using the new preconditioning technique leads to a more efficient solver. Using the new preconditioner, the total number of linear iterations remain the same, with no linear convergence failures, as we scaled up the problem. As a result, the total number of nonlinear iterations and function evaluations also remain identical, leading to an efficient solution scheme. In contrast, when the default preconditioner is used, the total number of linear iterations vary, as some time-steps require several iterations to converge. As a result, there are several linear convergence failures (on average 41 per problem), which leads to a slower convergence for the inexact Newton solver, and thus, leads to an inefficient solver.

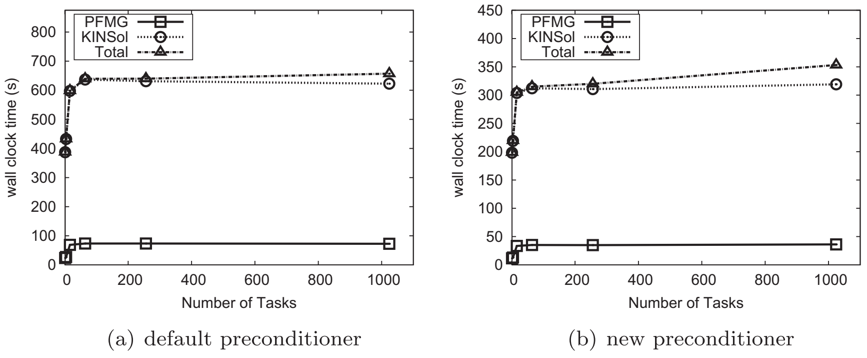
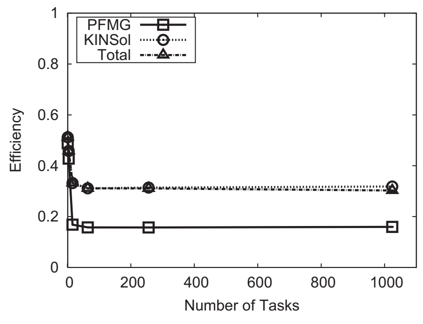
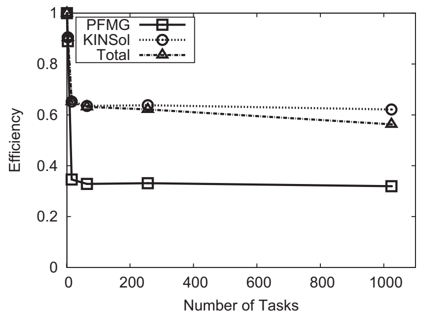


Fig. 7. Weak scaling study of the different preconditioner options, using the analytic Jacobian for the matrix–vector products in the Newton–Krylov solver.



(a) default preconditioner



(b) new preconditioner

Fig. 8. Scaled efficiency for the weak scaling study of the different preconditioner options, using the analytic Jacobian for the matrix–vector products in the Newton–Krylov solver.

Notice that in both test cases, there is a sudden drop in the efficiency between the first and second data points, particularly for the PFMG results. This is not uncommon and may be attributed to the cost of the initial communication being more than that of the computation involved to solve the sparse linear system. In each case, efficiency could be improved by solving a larger (local) problem on each processor. Alternatively, a different parallel architecture, such as the IBM BGQ architecture, may yield a different result. Notice, however, that this issue is independent of the solver performance, since both scale very well. Thus, difference in performance and solver efficiency between the different preconditioner options, will still remain.

5.5. Discussion

The numerical results presented above suggest that for problems where the nonlinear function is easy to compute and has no discontinuities (such as the excess infiltration example in Table 4, and for problems where flow is dominated by either subsurface or overland flow), the finite difference approximation to the Jacobian matrix–vector product yields a faster time to solution for the Newton–Krylov solver compared to using the analytic Jacobian to perform the matrix–vector product. However, the results also indicate that the analytic Jacobian yields a robust solver that can handle more challenging flow and topographic effects. This robustness is evident during any event where the surface transitions from unsaturated to saturated, such as at the onset of the V-Catchment example, or ponding and drying events, such as the excess infiltration example. During these events, the form of the equations change as the physics for overland flow contributes to the solution. These terms in (13) would not be included in the function evaluation for the first term of the finite difference Jacobian in (20) if the updated pressures are slightly below zero. However, the terms would still appear in the second function evaluation of (20), creating a discontinuity in the finite difference Jacobian. Notice that this discontinuity can occur even for small ϵ (see Section 3.2), and may also be triggered by roundoff errors in the finite difference approximation to the matrix–vector product, as discussed in Section 3.2. The analytic Jacobian avoids this discontinuity, since its evaluation relies only on the updated pressures. This situation leads to a consistent form for the Jacobian, which contributes to performance improvements in the solver.

6. Conclusion

This paper presents an effective and efficient numerical solution technique for implicit coupling of subsurface and overland flow models. We have shown a new approach to the solution of the coupled model that enables the use of an analytic form of the Jacobian within a Newton–Krylov solver. The approach relies on reordering the equations of the coupled flow problem to enable the computation and assembly of the analytic Jacobian in a form that benefits a numerical solution strategy using a structured solver. This approach also allows us to construct an effective preconditioning strategy that takes advantage of the availability of an analytic form of the Jacobian to construct a multigrid preconditioner for the Newton–Krylov solver.

The numerical results present strong evidence to support the use of the analytic Jacobian for the coupled flow problem. While the finite difference approximation to the Jacobian is cheaper to compute and hence may yield a faster time to solution for the solver; the analytic Jacobian yields a more robust solver that can handle challenging problems where the nonlinear function has discontinuities due to the conditional application of the overland flow boundary condition. In such cases, the finite difference approximation to the matrix–vector product relies on inconsistent behavior, resulting in solver failure.

Obtaining an effective preconditioner for the coupled flow problem is also another source of difficulty for the solver. Previous strategies to construct the preconditioner were not very effective due to the lack of adequate representation of the overland flow contributions. We have presented a new strategy for constructing the preconditioner that takes into account the non-symmetric contributions that come from upwinding in the overland flow model. The new preconditioner is very effective and significantly improves solver performance. Our results show that using the new preconditioner significantly outperforms a preconditioning strategy previously used in the literature. Furthermore, a parallel weak scaling study shows that the solver exhibits good parallel scaling, and has improved parallel efficiency when the new preconditioner is used.

To summarize, the numerical results suggest that for problems where the nonlinear function is easy to compute and has no discontinuities, it is beneficial to use the finite difference approximation for the Jacobian matrix–vector product. However, for more general cases, is it more appropriate to use the analytic Jacobian for the matrix–vector product. The results further suggest that a preconditioner that includes non-symmetric contributions from the overland flow terms improves the performance of the solver. This is especially true when the preconditioner is used in conjunction with the analytic Jacobian for the matrix–vector product.

Acknowledgments

We wish to thank all the anonymous referees for reviewing this work and for their comments and suggestions to improve the quality of this paper. We also wish to thank Steve Smith and Rob Falgout of LLNL for their time in several discussions on this project.

References

[1] Ebel BA, Loague K, Vanderkwaak JE, Dietrich WE, Montgomery DR, Torres R, Anderson SP. Near-surface hydrologic response for a steep, unchanneled catchment near Coos Bay, Oregon: 2. Physics-based simulations. Am J Sci 2007;307:709–48. http://dx.doi.org/10.2475/04.2007.03.

[2] Kollet SJ, Maxwell RM. Integrated surface-groundwater flow modeling: a free-surface overland flow boundary condition in a parallel groundwater flow model. Adv Water Resour 2006;29(7):945–58. http://dx.doi.org/10.1016/j.advwatres.2005.08.006.

[3] Meyerhoff S, Maxwell R. Quantifying the effects of subsurface heterogeneity on hillslope runoff using a stochastic approach. Hydrogeol J 2011;19:1515–30. http://dx.doi.org/10.1007/s10040-011-0753-y.

[4] VanderKwaak JE, Loague K. Hydrologic-response simulations for the r-5 catchment with a comprehensive physics-based model. Water Resour Res 2001;37(4):999–1013. http://dx.doi.org/10.1029/2000wr900272.

[5] Bixio AC, Gambolati G, Paniconi C, Putti M, Shestopalov VM, Bublias VN, Bohuslavsky AS, Kastelteseva NB, Rudenko YF. Modeling groundwater–surface water interactions including effects of morphogenetic depressions in the chernobyl exclusion zone. Environ Geol 2002;42(2–3):162–77. http://dx.doi.org/10.1007/s00254-001-0486-7.

[6] Camporese M, Paniconi C, Putti M, Orlandin S. Surface-subsurface flow modeling with path-based runoff routing, boundary condition-based coupling, and assimilation of multisource observation data. Water Resour Res 2010;46(2):W02512. http://dx.doi.org/10.1029/2008WR007536.

[7] Panday S, Huyakorn PS. A fully coupled physically-based spatially-distributed model for evaluating surface/subsurface flow. Adv Water Resour 2004;27(4):361–82. http://dx.doi.org/10.1016/j.advwatres.2004.02.016.

[8] Kumar M, Duffy CJ, Salvage KM. A second order accurate, finite volume based, integrated hydrologic modeling (FIHM) framework for simulation of surface and subsurface flow. Vadose Zone J 2009;8:873–90. http://dx.doi.org/10.2136/vzj2009.0014.

[9] Jones JP, Sudicky EA, McLaren RG. Application of a fully-integrated surface-subsurface flow model at the watershed-scale: A case study. Water Resour Res 2008;44:W03407. http://dx.doi.org/10.1029/2006wr005603.

[10] Kollet SJ, Maxwell RM. Capturing the influence of groundwater dynamics on land surface processes using an integrated, distributed watershed model. Water Resour Res 2008;44:W02402. http://dx.doi.org/10.1029/2007WR006004.

[11] Qu Y, Duffy CJ. A semidiscrete finite volume formulation for multiprocess watershed simulation. Water Resour Res 2007;43:W08419. http://dx.doi.org/10.1029/2006wr005752.

[12] Shen C, Phanikumar MS. A process-based, distributed hydrologic model based on a large-scale method for surface-subsurface coupling. Adv Water Resour 2010;33:1524–41. http://dx.doi.org/10.1016/j.advwatres.2010.09.002.

[13] Dawson C. A continuous/discontinuous Galerkin framework for modeling coupled subsurface and surface water flow. Comput Geosci 2008;12:451–72. http://dx.doi.org/10.1007/s10596-008-9085-y.

[14] Jones JE, Woodward CS. Newton–Krylov-multigrid solvers for large-scale, highly heterogeneous, variably saturated flow problems. Adv Water Resour 2001;24(7):763–74. http://dx.doi.org/10.1016/S0309-1708(00)00075-0.

[15] Park Y-J, Sudicky E, Panday S, Matanga G. Implicit subtime stepping for solving nonlinear flow equations in an integrated surface-subsurface system. Vadose Zone J 2009;8:825–36. http://dx.doi.org/10.2136/vzj2009.0013.

[16] Maxwell R. A terrain-following grid transform and preconditioner for parallel, large-scale, integrated hydrologic modeling. Adv Water Resour 2013;53:109–17. http://dx.doi.org/10.1016/j.advwatres.2012.10.001.

[17] Kazezylmaz-Alhan CM, Medina Jr MA, Rao P. On numerical modeling of overland flow. Appl. Math. Comput. 2005;166(3):724–40. http://dx.doi.org/10.1016/j.amc.2004.06.063.

[18] Richards LA. Capillary conduction of liquids through porous mediums. Physics 1931;1:318–33. http://dx.doi.org/10.1063/1.1745010.

[19] Chow VT, Maidment DR, Mays LW. Applied hydrology. In: Eliassen R, King PH, Linsey RK, editors. McGraw-Hill series in water resources and environmental engineering. McGraw-Hill Inc; 1998. p. 572. http://dx.doi.org/10.1036/0070108102.

[20] Kelley CT. Newton–Krylov methods. Philadephia, PA: Society for Industrial and Applied Mathematics; 2003. http://dx.doi.org/10.1137/1.9780898718898.ch3.

[21] Knoll DA, Keyes DE. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. J Comput Phys 2004;193(2):357–97. http://dx.doi.org/10.1016/j.jcp.2003.08.010.

[22] Ypma TJ. Historical development of the Newton–Raphson method. SIAM Rev 1995;37(4):531–51. http://dx.doi.org/10.1137/1037125.

[23] Dembo RS, Eisenstat SC, Steihaug T. Inexact Newton methods. SIAM J Numer Anal 1982;19:400–8. http://dx.doi.org/10.1137/0719025.

[24] Brown PN, Saad Y. Hybrid Krylov methods for nonlinear systems of equations. SIAM J Sci Stat Comput 1990;11(3):450–81. http://dx.doi.org/10.1137/0911026.

[25] Brown PN. A local convergence theory for combined inexact-Newton/finite-difference projection methods. SIAM J Numer Anal 1987;24(2):407–34. http://dx.doi.org/10.1137/0724031.

[26] Maxwell RM, Putti M, Meyerhoff S, Delfs J-O, Ferguson IM, Ivanov V, Kim J, Kolditz O, Kollet SJ, Kumar M, Lopez S, Niu J, Paniconi C, Park Y-J, Phanikumar MS, Shen C, Sudicky EA, Sulis M. Surface-subsurface model intercomparison: a first set of benchmark results to diagnose integrated hydrology and feedbacks. Water Resour Res 2014;50:1531–49. http://dx.doi.org/10.1002/2013WR013725. URL <http://dx.doi.org/10.1002/2013WR013725>.

[27] Sulis M, Meyerhoff SB, Paniconi C, Maxwell RM, Putti M, Kollet SJ. A comparison of two physics-based numerical models for simulating surface water-groundwater interactions. Adv Water Resour 2010;33:456–67. http://dx.doi.org/10.1016/j.advwatres.2010.01.010.

[28] Tompson AFB, Falgout RD, Smith SG, Bosl WJ, Ashby SF. Analysis of subsurface contaminant migration and remediation using high performance computing. Adv Water Resour 1998;22:203–21. http://dx.doi.org/10.1016/S0309-1708(98)00013-X.

[29] Falgout RD, Yang UM. hypre: A Library of high performance preconditioners. In: Preconditioners, lecture notes in computer science; 2002, pp. 632–41. URL <http://computation.llnl.gov/casc/hypre/software.html>.