A MULTI-GRID BASED SOLVER FOR MIXED FINITE-ELEMENT APPROXIMATIONS TO GROUNDWATER FLOW

Myron B. Allen

1992

Proceedings WWRC-92-10

In

Computational Methods in Water
Resources IX, Volume 1:
Numerical Methods in Water Resources

T.F. Russell, et al., eds.

Submitted by

Myron B. Allen
Department of Mathematics
University of Wyoming
Laramie, Wyoming

A Multigrid-Based Solver for Mixed Finite-Element Approximations to Groundwater Flow

Myron B. Allen[1]
*Department of Mathematics, University of Wyoming,
Laramie, WY 82071, U.S.A.*
Mark C. Curran[2]
*Applied and Numerical Mathematics Division, Sandia
National Laboratories, Albuquerque, NM 87185, U.S.A.*

ABSTRACT

Mixed finite-element methods have several features that are attractive in the numerical simulation of groundwater flow. Chief among these is the possibility of computing Darcy velocities whose accuracies are comparable to those of the computed hydraulic heads. Much current research centers on solving the large linear systems that arise from mixed finite-element discretizations. We examine an iterative method that largely overcomes the poor conditioning associated with fine spatial grids and highly variable aquifer properties. The method incorporates a multigrid scheme inside an outer iteration whose convergence rate is independent of grid mesh size and variations in hydraulic conductivity. As we demonstrate, the multigrid algorithm is amenable to effective parallelization on distributed-memory machines, making the overall algorithm a highly efficient one in such computing environments.

[1] The Wyoming Water Research Center partially supported this work through a grant-in-aid

[2] This work received support from the Applied Mathematical Sciences Program, U.S. Department of Energy Office of Energy Research. The work was performed in part at Sandia National Laboratories for the U.S. DOE under contract number DE-AC04-76DP00789.

1. INTRODUCTION

The equations governing the steady flow of a single fluid in a two-dimensional porous medium Ω with no gravity drive have the following forms:

$$\begin{aligned} \mathbf{u} &= -K \operatorname{grad} p \quad \text{in } \Omega, \\ \operatorname{div} \mathbf{u} &= f \quad \text{in } \Omega. \end{aligned} \tag{1}$$

Here $\mathbf{u} = (u^x, u^y), p$, and f represent the Darcy velocity, hydraulic head, and source term, respectively. In many natural groundwater aquifers, the hydraulic conductivity $K(x, y)$ exhibits irregular variations depending upon the lithology of the host rock. This heterogeneous structure causes many difficulties for numerical modelers, among which are two sources of poor conditioning in linear systems that approximate the differential equations. One source of poor conditioning is the need to use fine spatial grids to resolve the complexities of the medium and the resulting variations in p and $\mathbf{u}$. Another source is the variability in K itself, which affects the coefficients in the matrices of the linear system. These difficulties afflict essentially all discrete approximations to Equations (1).

Among the enormous variety of such methods, mixed finite-element methods have attracted a great deal of attention over the past decade. These methods, together with appropriate choices of trial spaces, yield solutions for p and $\mathbf{u}$ that have the same order of accuracy as the grid mesh size $h \to 0$ (Douglas et al.[1], Raviart and Thomas [2]). This property stands in contrast to many standard Galerkin and finite-difference formulations, where one first solves for p and then numerically differentiates to compute a less accurate approximation to $\mathbf{u}$. Thus mixed methods are particularly well suited to problems where accurate velocities are critical to the prediction of underground contaminant movements.

This paper examines an iterative scheme for solving the lowest-order mixed finite-element approximations to Equations (1) on rectangular grids. The overall structure of the scheme, analyzed in detail by Allen et al.[3], consists of an outer iteration, whose convergence rate is independent of h and of spatial variations in K, coupled with an inner iteration on an elliptic linear system. Rapid execution of this inner iteration is crucial to the efficiency of the scheme. We use a highly parallelizable multigrid method to perform the inner iterations.

Section 2 reviews the mixed finite-element method. Section 3 discusses the iterative scheme, reviews its theoretical properties, and describes the multigrid method used in the inner iteration. Section 4 presents numerical results that indicate the efficiency of the scheme. In Section 5 we briefly draw some conclusions.

2. THE MIXED FINITE-ELEMENT METHOD

Consider Equations (1), subject to the boundary condition $p = 0$ on $\partial\Omega$. To discretize this system via the lowest-order mixed finite-element method, we establish a rectangular grid Δ on Ω having vertical grid lines at $x_0, x_1, \ldots, x_m$ and horizontal grid lines at $y_0, y_1, \ldots, y_n$, as drawn in Figure 1. The mesh size h of Δ is the maximum distance between adjacent grid lines $x = x_i$ or $y = y_j$. With Δ we associate trial spaces[2] Q_x, Q_y, and V for the x-velocity u^x, the y-velocity u^y, and the hydraulic head p, respectively. Functions in Q_x are piecewise linear in x and piecewise constant in y; functions in Q_y are piecewise constant in x and piecewise linear in y, and functions in V are piecewise constant on Δ.

Each of these trial spaces has a finite nodal basis consisting of tensor products of the usual one-dimensional bases for piecewise constant and piecewise linear interpolation. As Figure 1 illustrates, we associate a nodal value $p_{i,j}$ of head with the centroid of each cell $[x_{i-1}, x_i] \times [y_{j-1}, y_j]$ formed by the grid Δ, a nodal value $u^x_{i,j}$ of x-velocity with the midpoint $(x_i, y_{j-1/2})$ of each vertical cell edge, and a nodal value $u^y_{i,j}$ with the midpoint $(x_{i-1/2}, y_j)$ of each horizontal cell edge.

Given these trial spaces, the mixed finite-element method for solving Equations (1) is as follows: Find trial functions $\mathbf{u}_h \in Q_x \times Q_y$ and $p_h \in V$ such that

$$\begin{aligned} &\int_\Omega \frac{\mathbf{u}_h \cdot \mathbf{v}}{K}\, dx\, dy - \int_\Omega p_h \operatorname{div} \mathbf{v}\, dx\, dy = 0, \quad \forall\, \mathbf{v} \in Q_x \times Q_y, \\ &\int_\Omega (\operatorname{div} \mathbf{u}_h - f) q\, dx\, dy = 0, \qquad\qquad\qquad \forall\, q \in V. \end{aligned} \tag{2}$$

This finite-element discretization yields approximations $\mathbf{u}_h$ and p_h whose global errors are both $O(h)$ in the norm $\|\cdot\|_{L^2(\Omega)}$ (see Raviart and Thomas[2]).

Under a natural ordering of equations and unknowns, Equations (2) yield a linear system having the following block structure:

$$\begin{bmatrix} A & N \\ N^T & 0 \end{bmatrix} \begin{bmatrix} U \\ P \end{bmatrix} = \begin{bmatrix} 0 \\ F \end{bmatrix}. \tag{3}$$

Here, U stands for a vector containing the nodal values of the velocities u^x and u^y, and P is a vector containing nodal heads. The block matrix A is symmetric and positive definite and has the block structure

$$A = \begin{bmatrix} A^x & 0 \\ 0 & A^y \end{bmatrix}.$$

The blocks $A^x \in \mathbb{R}^{(m+1)n \times (m+1)n}$ and $A^y \in \mathbb{R}^{m(n+1) \times m(n+1)}$ are tridiagonal, and their entries are integrals involving the variable hydraulic conductivity K. The matrix N has the block structure

$$N = \begin{bmatrix} N^x \\ N^y \end{bmatrix},$$

where $N^x \in \mathbb{R}^{(m+1)n \times mn}$ and $N^y \in \mathbb{R}^{m(n+1)\times mn}$. These blocks reduce to the usual difference approximations to $\partial/\partial x$ and $\partial/\partial y$. The vector $F \in \mathbb{R}^{mn}$ contains integrals involving the source function f. For details concerning the construction of this linear system, we refer readers to Allen et al.[3]

3. AN ITERATIVE SCHEME

We solve the system (3) iteratively, using the following matrix splitting:

$$\begin{bmatrix} D & N \\ N^\top & 0 \end{bmatrix} \begin{bmatrix} U \\ P \end{bmatrix}^{(k+1)} = \begin{bmatrix} 0 \\ F \end{bmatrix} + \begin{bmatrix} D-A & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U \\ P \end{bmatrix}^{(k)}. \tag{4}$$

Here, D is a diagonal matrix that can have any of several structures, the simplest of which is just the diagonal part of A. This scheme has convergence rate that is independent of the mesh size h and the variations in hydraulic conductivity K; in fact, each iteration reduces the error by a factor no greater than $\frac{1}{2}$ (see Allen et al.[3]).

Computationally, the scheme (4) requires the following steps:

(*i*) $G^{(k-1)} \leftarrow -F + N^\top D^{-1}(D-A)U^{(k-1)}$.

(*ii*) Solve $N^\top D^{-1} N P^{(k)} = G^{(k-1)}$.

(*iii*) $U^{(k)} \leftarrow D^{-1}(D-A)U^{(k-1)} - D^{-1}NP^{(k)}$.

Steps (*i*) and (*iii*) in this algorithm require only matrix multiplication and addition and hence are quite cheap. Step (*ii*), however, requires more work, since the matrix $N^\top D^{-1} N$ has the same pentadiagonal structure as the usual five-point finite-difference approximation to operators of the form div$(K\,\text{grad})$.

To execute this step efficiently, we use a multigrid scheme. Instead of solving step (*ii*) exactly, we perform several V-cycles to get an approximate solution for $P^{(k)}$, then move on to step (*iii*). Each V-cycle involves two Gauss-Seidel iterations at each level in a nest $\Delta = \Delta_0 \supset \Delta_1 \supset \cdots \supset \Delta_L$ of grids, ranging from the original grid Δ through coarser subgrids to the coarsest grid Δ_L, then back up to the finest grid Δ. To map the problem from fine grids to coarse grids, we use full weighting as a restriction operator. To map from coarse grids to fine grids, we use bilinear interpolation as a prolongation operator.

One attractive feature of the multigrid scheme is its amenability to parallel processing. Tuminaro and Womble[4] discuss this prospect. By adopting a red-black ordering for the cells in each grid, we can decompose each Gauss-Seidel relaxation sweep into two sets of calculations. In particular, we label each cell $[x_{i-1}, x_i] \times [y_{j-1}, y_j]$ in a grid as "red" or "black," depending on whether $i + j$ is even or odd. We can update each of the "red" cells using old values in the "black" cells, then use these updated values in "red" cells to update each of the "black" cells. Since the calculations for "red" cells in any sweep are independent of each other, we can perform the arithmetic concurrently on a parallel computer. Similarly, the updates for "black" cells are also mutually independent and can be computed concurrently.

This idea works especially well on distributed-memory machines, where it is feasible to have a large number of processors that communicate through message passing instead of accessing a shared memory. In coding the algorithm, we decompose the spatial domain of the problem so that each processor in a parallel machine performs the calculations for a subset of the grid.[4] At any instant during the calculations, a given processor is performing either "red" or "black" updates. The "red" and "black" processors work simultaneously, stopping synchronously to exchange results and "change colors."

4. COMPUTATIONAL PERFORMANCE

Since Allen et al.[3] discuss the performance of the serial precursor to our scheme in the presence of a variety of heterogeneous conductivity fields $K(x, y)$, we focus here on the performance of the parallel version. To assess this performance, we examine the execution time required by our code on a 1024-processor nCUBE 2 having a distributed-memory hypercube archtecture. To gain some appreciation for the degree of parallelism in the code, we investigate the execution time required on subcubes of the machine having dimension 0 (1 processor), 1 (2 processors), ..., 10 (1024 processors).

We base our assessment on the notion of *scaled speedup.* To gauge the scaled speedup, we run problems of proportionately larger size on larger subcubes. Specifically, we run a problem in which we assign the work associated with a 32×32-cell grid with each processor of a subcube of the machine. Thus we use a 32×32-cell grid on the one-processor subcube, one involving a 64×32-cell grid on the two-processor subcube, one involving a 64×64-cell grid on the four-processor subcube, and so forth, eventually running a problem involving a 1024×1024 grid on the 1024-processor cube. Since the ratio of problem size to number of processors remains constant in this sequence, an algorithm possessing ideal parallelism would require the same execution time for all runs. In practice, interprocessor communication and computational overhead, such as setup time, interfere with this ideal relationship.

Table 1 lists the timings for the runs. The table shows the total execution times, along with the times associated with problem setup (e.g. initialization and matrix assembly) and interprocessor communication, for various subcubes of the machine. The subcubes range in size from dimension zero (one processor operating on a 1024-cell grid) to dimension 10 (1024 processors acting on a 1,048,576-cell grid). Each run represents 20 outer iterations of the scheme (4), each iteration of which requires five V-cycles in step (*ii*). The times listed in the second through fourth columns are averages over all processors, while the times listed in the last column are the maximum times over all processors and thus more closely reflect the apparent execution time observed by a user. These timings suggest that the algorithm possesses excellent parallelism in addition to its good performance in the presence of heterogeneities and fine grids.

Table 1: RUNTIMES (SECONDS) FOR SCALED PROBLEMS ON THE nCUBE 2.

Number of processors	Setup time	Communication time	Average total time	Maximum total time
1	6.515	0.205	41.119	41.119
2	6.298	1.783	43.131	43.133
4	6.109	3.694	45.389	45.454
8	6.114	4.640	46.607	46.647
16	6.150	5.774	48.088	48.160
32	6.223	5.832	48.357	48.526
64	6.371	5.907	48.681	48.998
128	6.671	5.949	49.096	49.721
256	7.290	5.979	49.789	51.035
512	8.565	6.028	51.149	53.693
1024	11.141	6.077	53.795	58.981

5. CONCLUSIONS

Our algorithm appears to promise excellent opportunities for parallel computing as well as a reasonable way to overcome some of the numerical difficulties associated with heterogeneities. Given this promise, we see our next task as the extension of the method to time-dependent and nonlinear problems, which have more general applicability to underground contaminant modeling.

ACKNOWLEDGMENT

The authors thank Dick Ewing, whose insights guided much of our work.

REFERENCES

1. Douglas, J., Ewing, R.E., and Wheeler, M.F., "The Approximation of the Pressure by a Mixed Method in the Simulation of Miscible Displacement," *R.A.I.R.O. Analyse Numerique 17*, pp. 17-33, 1983.

2. Raviart, P.A., and Thomas, J.M., "A Mixed Finite Element Method for Second Order Elliptic Problems," in *Mathematical Aspects of Finite Element Methods,* Lecture Notes in Mathematics vol. 606, ed. by I. Galligani and E. Magenes, pp. 292-315, Springer-Verlag, Berlin, 1977.

3. Allen, M.B., Ewing, R.E., and Lu, P., "Well Conditioned Iterative Schemes for Mixed Finite-Element Models of Porous-Media Flow," to appear in *SIAM Jour. Sci. Stat. Comp.*

4. Tuminaro, R.S., and Womble, D.E., "Analysis of the Multigrid FMV Cycle on Large-Scale Parallel Machines," to appear in *SIAM Jour. Sci. Stat. Comp.*

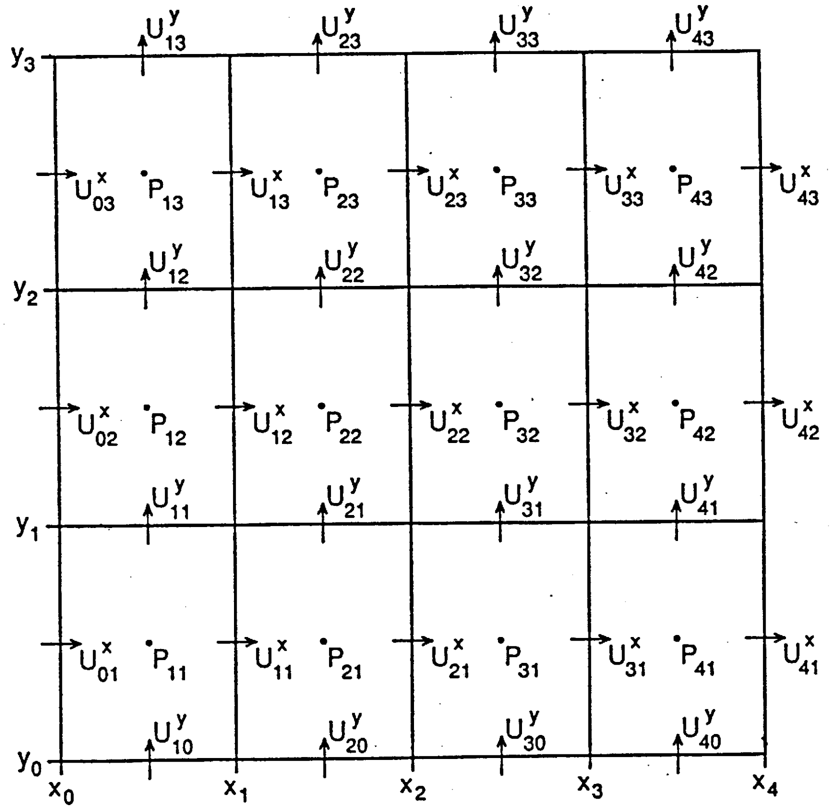

Figure 1. Sample grid for the mixed finite-element method, showing nodes for the hydraulic head and the x- and y-velocities.