ground water

A Fast Direct Solution Method for Nonlinear Equations in an Analytic Element Model

by Henk M. Haitjema[1]

Abstract

The analytic element method, like the boundary integral equation method, gives rise to a system of equations with a fully populated coefficient matrix. For simple problems, these systems of equations are linear, and a direct solution method, such as Gauss elimination, offers the most efficient solution strategy. However, more realistic models of regional ground water flow involve nonlinear equations, particularly when including surface water and ground water interactions. The problem may still be solved by use of Gauss elimination, but it requires an iterative procedure with a reconstruction and decomposition of the coefficient matrix at every iteration step. The nonlinearities manifest themselves as changes in individual matrix coefficients and the elimination (or reintroduction) of several equations between one iteration and the other. The repeated matrix reconstruction and decomposition is computationally intense and may be avoided by use of the Sherman-Morrison formula, which can be used to modify the original solution in accordance with (small) changes in the coefficient matrix. The computational efficiency of the Sherman-Morrison formula decreases with increasing numbers of equations to be modified. In view of this, the Sherman-Morrison formula is only used to remove equations from the original set of equations, while treating all other nonlinearities by use of an iterative refinement procedure.

Introduction

The analytic element method (AEM) requires the determination of one or more "strength" parameters for each analytic element in the model domain (Strack 1989; Haitjema 1995). These strength parameters are solved for by defining appropriate conditions at collocation points, usually at the elements themselves. For instance, to determine the sink density of a constant-strength line-sink, a known potential may be specified at its center. Since all elements contribute to the potential field in an infinite domain, the resulting coefficient matrix for the set of equations is fully populated. The AEM shares this characteristic with the closely related boundary integral equation method (BIEM) (Liggett and Liu 1983). In view of this full matrix, most AEM and BIEM models have employed a direct solution method, such as Gauss elimination. A popular implementation of Gauss elimination is LU decomposition followed by forward elimination and back substitution applied to the "known vector" (Forsythe et al. 1977). The latter procedure is used by the public domain AEM solver GFLOW1 (Haitjema 1995).

Application of the AEM to regional ground water flow problems, however, generally leads to a set of nonlinear equations because (1) Cauchy-type boundary conditions in unconfined aquifers result in at least one of the matrix coefficients (per equation) being dependent on the solution and (2) during the solution procedure, the status of some strength parameters may change from a priori unknown to known. For instance, if a line-sink represents a loosing stream section and the water table drops below the resistance layer that separates the aquifer from the stream, the stream starts percolating water and the sink density (strength) of the line-sink does no longer depend on the head in the aquifer; it can be calculated in advance. Similarly, when solving surface water and ground water conjunctively, loosing stream sections may be found to have no streamflow, which implies a zero sink density for the associated line-sinks (Mitchell-Bruker and Haitjema 1996). This requires iterations between (direct) solutions by which some matrix elements

[1]School of Public and Environmental Affairs, Indiana University, SPEA 439, Bloomington, IN 47405; haitjema@indiana.edu
Received June 2003, accepted July 2005.

doi: 10.1111/j.1745-6584.2005.00145.x

are updated and some equations are eliminated (or reintroduced) at successive iteration steps. Consequently, the construction of the coefficient matrix and its LU decomposition must be repeated for each iteration, which is computationally inefficient.

In this paper, I show how this process can be accelerated by modifying the solution in response to the changes in the system of equations, thus avoiding the repeated reconstruction and decomposition of the coefficient matrix.

Iterative Use of Direct Solutions

The system of equations that results from the AEM formulation may be written as:

$$a_{ij}x_j = b_i \tag{1}$$

by which the coefficient matrix a_{ij} contains the "influences" of all analytic elements on all collocation points and is fully populated. The vector x_i is the "solution vector" and contains the strength parameters of the analytic elements. The vector b_i is the known vector resulting from the boundary conditions formulated at the collocation points. In writing Equation 1, the Einstein summation convention is adopted, implying summation over the dummy index j.

Iterative Refinement

In GFLOW, the solution process is repeated several times, whereby Equation 1 is replaced by (Haitjema 1995):

$$a_{ij}\left(x_j^{(n)} - x_j^{(n-1)}\right) = b_i^{(n)} - b_i^{(n-1)} \tag{2}$$

where $x_i^{(n)}$ and $b_i^{(n)}$ are the solution vector and known vector, respectively, of the current solution step n, while $x_i^{(n-1)}$ and $b_i^{(n-1)}$ are the solution vector and known vector, respectively, at the previous solution step $n - 1$. The formulation 2 has an advantage over Equation 1 in that the numerical accuracy of the Gauss elimination procedure is applied to calculating the change in the strength parameters between iterations, rather than calculating the strength parameters themselves. Since GFLOW1 is coded in double-precision Fortran, successive solutions are likely to lead to an iterative refinement of the strength parameters.

Solving a Nonlinear Problem

As stated in the Introduction, nonlinearity may come about in a number of different ways. Limiting us to the example of a system of equations with line-sinks only, there are two types of nonlinear behavior: (1) line-sinks that represent surface water features that are separated from the aquifer by a resistance layer and (2) line-sinks, the strength of which is dictated by available streamflow (conjunctive surface water and ground water flow solutions). In case 1, the change in a_{ij} constitutes a modification of the diagonal element a_{ii}, while for case 2, the strength parameter can be calculated beforehand and the associated equation is removed from the system of equations. Due to these changes in the system of equations, it is necessary, at each iterative step, to generate a new known vector and a new matrix, decompose that matrix, and apply forward elimination and back substitution to obtain a new solution vector. The bulk of the computational effort, however, is associated with the matrix reconstruction and decomposition.

Using the Sherman–Morrison Formula

It is possible to change matrix coefficients and even eliminate entire equations from an existing set of equations by modifying the solution vector x_i for the original set of equations into a new solution vector for the modified set of equations without rebuilding and decomposing the matrix in successive iterations. To do so, I make use of the Sherman-Morrison formula (Gill et al. 1974; Press et al. 1992; Golub and Loan 1996), which, written in index notation, is:

$$\left(a_{ij} - u_i v_j\right)^{-1} = a_{ij}^{-1} + \frac{a_{ik}^{-1}u_k v_l a_{lj}^{-1}}{1 - \lambda} \tag{3}$$

with

$$\lambda = v_k a_{kl}^{-1} u_l \tag{4}$$

The vector product $u_i v_j$ forms a correction matrix for the original coefficient matrix a_{ij}. The Sherman-Morrison formula provides an expression for the modification of the inverse of the original coefficient matrix based on this correction. In practice, however, the inverse matrix is seldom calculated. Therefore, expression 3 with Equation 4 will be applied to the set of equations 1 to obtain a correction of its solution x_i based on the modification of the coefficient matrix as shown on the left-hand side of Equation 3. In fact, Equation 1 can be seen as representing Equation 2 when x_i and b_i are replaced by the differences of these vectors as shown in Equation 2.

I write the solution x_i as:

$$x_i = a_{ij}^{-1} b_j \tag{5}$$

I write the solution x_i^* to the modified set of equations as:

$$x_i^* = \left(a_{ij} - u_i v_j\right)^{-1} b_j \tag{6}$$

In writing Equations 5 and 6, it is implied that the known vector b_i belongs to the modified set of equations. Applying Equation 3 yields:

$$x_i^* = a_{ij}^{-1} b_j + \frac{a_{ik}^{-1}u_k v_l a_{lj}^{-1} b_j}{1 - \lambda} \tag{7}$$

where λ is defined by Equation 4. Expression 7 may be rewritten by use of Equation 5 as:

$$x_i^* = x_i + \frac{u_i' v_l x_l}{1 - \lambda} \tag{8}$$

where u_i' is defined as

$$u_i' = a_{ij}^{-1} u_j \tag{9}$$

with which λ becomes:

$$\lambda = v_k u'_k \tag{10}$$

In summary, the computation of the solution x_i^* for the modified set of equations requires the following steps: (1) calculate an initial solution x_i based on the original coefficient matrix and the known vector b_i for the modified set of equations; (2) calculate the vector u'_i, which is the solution to a system of equations with the original coefficient matrix and the vector u_i as known vector; and (3) calculate the new solution vector x_i^* by use of Equation 8 with Equation 10. Steps 1 and 2 can be accomplished by any solution method, including the LU decomposition, forward elimination, and back substitution employed in GFLOW1.

Eliminating One Equation

Once a line-sink strength parameter changes its status from unknown to known, its corresponding equation must be removed from the original set of equations. Instead of actually removing the n^{th} equation, however, we will force the solution to contain the specified value for the line-sink strength parameter. Hence, $x_n = 0$ since x_n represents the change in strength. To obtain this result, we replace the original n^{th} equation by:

$$0x_1 + 0x_2 + 0x_3 + \cdots + a_{nn}x_n + \cdots + 0C = 0 \tag{11}$$

where a_{nn} may be any nonzero number, but for numerical reasons is best selected very large. All other equations remain unaltered. Expression 11 implies that all matrix coefficients in row n must be set to zero, except a_{nn}, which must be set to a large number. In the Sherman-Morrison formula (Equation 3), this is accomplished by selecting the vectors u_i and v_i as follows:

$$u_i = 0 \quad (i \neq n); \quad u_n = 1 \tag{12}$$

and

$$v_i = a_{ni} \quad (i \neq n); \quad v_n = 10^{100} \tag{13}$$

The choice for v_n is arbitrary.

Eliminating *N* Equations

If more than one equation is to be removed, which is usually the case, the new set of equations may be written in the form:

$$\left(a_{ij} - \sum_{n=1}^{N} \overset{n}{u}_i \overset{n}{v}_j\right) x_i = b_i \tag{14}$$

The solution x_i to Equation 14 may be obtained by repeated application of the Sherman-Morrison formula. To develop expressions for this procedure, I define the coefficient matrix $\overset{m}{a}_{ij}$ as the original matrix with m modifications:

$$\overset{m}{a}_{ij} = \left(a_{ij} - \sum_{n=1}^{m} \overset{n}{u}_i \overset{n}{v}_j\right) \tag{15}$$

I further define $\overset{m}{x}_i$ as

$$\overset{m}{x}_i = \left(\overset{m-1}{a}_{ij}\right)^{-1} b_j \tag{16}$$

and similarly

$$\overset{m}{u}'_i = \left(\overset{m-1}{a}_{ij}\right)^{-1} \overset{n}{u}_j \tag{17}$$

The solution vector $\overset{m}{x}_i$ is obtained by solving the original system of equations with m equations modified and the vector b_i as known vector. The solution vector $\overset{m}{\overset{n}{u}}'_i$ is obtained by solving the original system of equations with m equations modified and the vector $\overset{n}{u}_i$ as known vector. Successively applying the Sherman-Morrison formula leads to the following recursion formula for $\overset{m}{x}_i$:

$$\overset{m+1}{x}_i = \overset{m}{x}_i + \frac{\overset{m}{\overset{m}{u}}'_i \overset{m}{v}_k \overset{m}{x}_k}{1 - \overset{m}{\lambda}} \tag{18}$$

where

$$\overset{m}{\lambda} = \overset{m}{v}_k \overset{m}{\overset{m}{u}}'_k \tag{19}$$

and where

$$\overset{1}{x}_i = a_{ij}^{-1} b_j \tag{20}$$

Evaluation of Equation 18 with Equation 19 requires the calculation of $\overset{m}{\overset{m}{u}}'_i$ by use of the following recursion formula:

$$\overset{n+1}{\overset{m}{u}}'_i = \overset{n}{\overset{m}{u}}'_i + \frac{\overset{n}{\overset{n}{u}}'_i \overset{n}{v}_k \overset{n}{\overset{m}{u}}'_k}{1 - \overset{n}{\lambda}} \tag{21}$$

where $\overset{n}{\lambda}$ is given by

$$\overset{n}{\lambda} = \overset{n}{v}_k \overset{n}{\overset{n}{u}}'_k \tag{22}$$

and where

$$\overset{1}{\overset{m}{u}}'_i = a_{ij}^{-1} \overset{m}{u}_j \tag{23}$$

Performance

The Sherman-Morrison formula is only efficient if the number of changes to the original system of equations is limited. In view of this, the modification of individual matrix coefficients for Cauchy-type boundary conditions (case 1 of the nonlinearities associated with line-sinks) is not implemented. Instead, this nonlinearity is handled as part of the iterative refinement procedure. The case 2 nonlinearity, the removal or reintroduction of equations, occurs less frequently and can only be handled by either reconstructing or altering the matrix followed by matrix decomposition, or by use of the Sherman-Morrison formula as outlined previously. The computational

efficiency then depends on the number of equations being removed. A standard LU decomposition requires $\frac{1}{2}N^3$ loops of one multiplication and one addition (Press et al. 1986). The Sherman-Morrison procedure requires $3N^2$ such loops, but with u_i, a unit vector, this is reduced to $2N^2$. The two procedures can be expected to be comparable in computational effort when m equations are being removed, so that $\frac{1}{2}N^3 = 2mN^2$, which means $m = 0.25N$. In GFLOW1, solving a system of 800 equations, the break even point was found experimentally to be a little higher: $m = 0.3N$. In other words, the Sherman-Morrison routine was more efficient than LU decomposition as long as the number of equations removed was < 30%.

In GFLOW1, the matrix solution procedure is organized in such a manner that all influence functions have to be calculated only once; they are stored to disk and reused for generating the known vector and verifying boundary conditions during successive iterations. Under these circumstances, the LU decomposition consumes ~95% of the computational effort for all but the first iteration. The Sherman-Morrison procedure for this case leads to significant savings in total solution time. The results reported here are approximate and depend somewhat on the manner in which the procedures are coded. In most practical cases, the equations include not only the effects of line-sinks but also those of line-doublets associated with inhomogeneity domains, where each line-doublet requires two equations (vs. one for each line-sink). Consequently, the number of equations to be removed is usually well below 30% of the original set, which makes the Sherman-Morrison approach attractive.

Acknowledgments

I would like to thank an anonymous reviewer and Prof. Randal Barnes for their helpful comments on my original manuscript.

References

Forsythe, G.E., M.A. Malcolm, and C.B. Moler. 1977. *Computer Methods for Mathematical Computations*. Prentice Hall Inc. Englewood Cliffs, NJ.

Gill, P.E., G.H. Golub, W. Murray, and M.A. Saunders. 1974. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28, no. 126: 505–535.

Golub, G. and A. van Loan. 1996. *Matrix Computations*. Johns Hopkins University Press, Baltimore, Maryland.

Haitjema, H.M. 1995. *Analytic Element Modeling of Groundwater Flow*. Academic Press Inc. San Diego, CA.

Liggett, J., and P.-F. Liu. 1983. *The Boundary Integral Equation Method for Porous Media Flow*. George Allen and Unwin Ltd. London, UK.

Mitchell-Bruker, S., and H.M. Haitjema. 1996. Modeling Steady State Conjunctive Ground Water and Surface Water Flow with Analytic Elements. *Water Resource Research*, 32, no. 9: 2725–2732.

Press, W.H., S. Teukolsky, W.T. Vettering, and B.P. Flannery. 1986. *Numerical Recipes in Fortran 77: The Art of Scientific Computing*, 2nd ed. Cambridge University Press. New York, NY.

Press, W.H., S. Teukolsky, T.W. Vettering, and B.P. Flannery. 1996. *Numerical Recipes in Fortran 90*, 2nd ed. Cambridge University Press. New York, NY.

Strack, O.D.L. 1989. *Groundwater Mechanics*. Prentice Hall. Englewood Cliffs, NJ.