Optimal Time-Varying Pumping Rates for Groundwater Remediation: Application of a Constrained Optimal Control Algorithm

LIANG-CHENG CHANG,1 CHRISTINE A. SHOEMAKER, AND PHILIP L.-F. LIU

School of Civil and Environmental Engineering, Cornell University, Ithaca, New York

A numerically efficient procedure is presented for computing optimal time-varying pumping rates for remediation of contaminated groundwater described by two-dimensional numerical models. The management model combines a pollutant transport model with a constrained optimal control algorithm. The transport model simulates the unsteady fluid flow and transient contaminant dispersion-advection in a two-dimensional confined aquifer. A Galerkin’s finite element method coupled with a fully implicit time difference scheme is applied to solve the groundwater flow and contaminant transport equations. The constrained optimal control algorithm employs a hyperbolic penalty function. Several sample problems covering 5–15 years of remediation are given to illustrate the capability of the management model to solve a groundwater quality control problem with time-varying pumping policy and water quality constraints. In our example, the optimal constant pumping rates are 75% more expensive than the optimal time-varying pumping rates, a result that supports the need to develop numerically efficient optimal control–finite element algorithms for groundwater remediation.

1. INTRODUCTION

Overview

A widely used technique for detoxifying contaminated aquifers is groundwater withdrawal followed by treatment [Lehr and Nielsen, 1982]. The objective of this paper is to develop a modeling approach to determine cost-effective time-varying strategies for this detoxification procedure. The planning model combines a nonlinear, distributed parameter, groundwater flow and contaminant transport simulation model with a control theory algorithm to compute optimal time-varying pumping rates.

Management models that incorporate a full convective-dispersive equation in an optimization formulation have been proposed by Gorelick et al. [1984] and Ahlfield et al. [1988a, b]. Gorelick et al. [1984] presented a model that minimizes the pumping cost while requiring that the contaminant concentration eventually falls below some standard at specified locations. They combined the two-dimensional finite element simulation model SUTRA [Voss, 1984] with the nonlinear programming package MINOS [Murtagh and Saunders, 1977, 1978, 1980]. Their optimization scheme required multiple simultaneous simulations of SUTRA in order to compute numerically the derivatives of the model output with respect to model input. These derivatives were then used to evaluate the Jacobian matrix. Ahlfield et al. [1988a, b] also combined a two-dimensional Galerkin’s finite element model with MINOS. Instead of approximating the derivatives by finite differences, Ahlfield et al. [1988a] employed adjoint sensitivity theory to derive the formula for computing these derivatives. Both Gorelick et al. and Ahlfield et al. considered constant pumping policies in time. Ahlfield [1990] extended this approach to consider two pumping periods. Ahlfield et al. [1988a, b] and Ahlfield [1990] only consider steady state flow equations during constant pumping periods although the equations for concentrations are transient. Atwood and Gorelick [1985] use a combination of simulation and linear programming to consider time-varying pumping rates for hydraulic gradient control. The use of linear programming has significant computational advantages, but, as is discussed by Atwood and Gorelick, this approach cannot guarantee identification of the optimal pumping strategy for a general remediation problem.

Control theory algorithms have been used in several water resource applications. Murray and Yakowitz [1979] presented a constrained differential dynamic programming (DDP, developed by Jacobson and Mayne [1970]) algorithm and applied it to the multireservoir control problem. They formulated the problem as a discrete optimal control problem with a linear transition function and linear constraints on the state and control variables. Their algorithm adapted the quadratic programming method into the DDP framework. As an extension of the same algorithm, Yakowitz [1986] presented a stagewise Kuhn-Tucker condition to ensure the convergence of the algorithm with the assumption of linear constraints. For Yakowitz’s algorithm to converge, the correct identification of the active constraint set is crucial. Yakowitz and coworkers did not apply their algorithms to groundwater contamination problems, which are nonlinear in contrast to the reservoir problems they considered that have linear transition equations. Jones et al. [1987] have applied what they called differential dynamic programming to a groundwater hydraulic control problem for an unconfined aquifer. Their formulation involved embedding a linearized finite difference flow equation into the management model. Their formulation did not consider groundwater quality. Andricovic and Kitanidis [1990] have combined a control algorithm with an extended Kalman filtering scheme to solve a groundwater remediation problem with parameter uncertainty. Their study focuses on uncertainty, not on numerical efficiency. They provided a very simple numerical example, which had one-dimensional flow, only one pumping well and one observation well. Their example covered a period of only 5 months, whereas realistic groundwater remediation problems typically require many years, often...
several decades. Their example had no adsorption, a single constraint (on concentration at one point) and no constraints on pumping rates. Like Jones et al. [1987] the Andricic and Kitandis study used a finite difference model to describe the groundwater flow and pollutant transport and they linearized the transition equations. Lee and Kitandis [1991] extended the Andricic and Kitandis paper to a two-dimensional example covering 1.4 years, four potential pumping wells, and three contaminant constraints, at supply wells.

The work developed in this paper differs from these earlier efforts in that we are (1) using a penalty function method (rather than an active set method) for constraints, (2) coupling an optimal control algorithm to a finite element model (rather than a finite difference model), and (3) solving two-dimensional groundwater remediation problems over long time periods with a large number of contamination constraints at observation points and numerous wells with time-varying pumping rates. To make our approach computationally efficient, we have used a new penalty function to enhance our ability to compute quickly solutions to constrained optimal control problems with a large number of constraints associated with observation points and bounds on pumping rates. We have also departed from earlier control theory applications by using an implicit finite element model of groundwater transport because finite element procedures have advantages over finite difference procedures including their adaptability for irregularly shaped regions. We have used implicit rather than explicit time discretization because this reduces the number of time steps required. Algebraic computation of the derivatives used in the control algorithm is considerably more complex for implicit than for explicit equations. These changes result in significantly different analytical equations for all derivatives used in the optimization and hence constitute a very different algorithm than those used earlier.

**Coupling Optimal Control and Numerical Models**

The approach we are following is to omit the calculation of second derivatives in the backward sweep of the DDP algorithm. This is a similar approach to that used by Jones et al. [1987] when they linearized the transition equation (which is equivalent to setting the second derivatives equal to zero). This approach is not actually differential dynamic programming but rather an earlier control theory technique that involves an approximation of a general nonlinear problem by a linear quadratic regulator (LQR) problem. The approximation involves the linearization of the transition equation and the approximation of the objective function by a second-order polynomial in the state and control vectors. In the approach used in this paper and in studies by Jones et al. [1987], and by Kitandis and coworkers, there is a simulation forward in time of the nonlinear simulation, which is a form of successive approximation. Hence we will refer to this method as SALQR (successive approximation LQR). In this "forward sweep" the full nonlinear transition equations \( T(x, u) \) are simulated forward in time

\[
x_{i+1} = T(x_i, u_i)
\]

where \( x_i \) is a vector of hydraulic heads and contaminant concentration. In our approach to the groundwater remediation problem, \( T \) is the set of equations in the implicit finite element model of groundwater flow and contaminant transport. The vector \( u_i \) is computed according to

\[
u_i = a_i(x_i - x_i^{-1}) + \beta_i + u_i^{-1}
\]

in which \( i \) is the number of the iteration and \( t \) is the time step; and \( a_i \) and \( \beta_i \) have been computed in the "backward sweep." In the backward sweep the computation of \( a_i \) and \( \beta_i \) is based on the solution of the optimization equations in time step \( i + 1 \). Hence, the backward sweep starts at time step \( t = N \) and goes backward in time to \( N - 1, N - 2, \ldots, 1 \). Details of the DDP algorithm are given by Yakowitz and Rutherford [1984]. The distinction between the DDP method and the SALQR approximation method is that DDP incorporates the second derivatives of the transition equation \( T \) in the backward sweep whereas the SALQR method omits them.

The results presented in this paper and in the works by Jones et al. and Kitandis and coworkers are based on SALQR theory, not on DDP. Chang [1990] discusses the computational difference between DDP and SALQR for groundwater problems. Both our approach and that used by Jones et al. [1987] and by Kitandis and coworkers differ from the conventional SALQR problem by the inclusion of constraint equations. Jones et al. [1987] and Kitandis and coworkers use an active set method similar to that developed by Yakowitz [1986]. We have used a penalty method in conjunction with an optimal control algorithm related to the approach employed by Liao and Shoemaker [1990]. Our work differs from the latter in that we use a different penalty function which was suggested in Lin [1990] and is described in section 4 of this paper.

In contrast to control theory algorithms like DDP or the SALQR approximation, nonlinear programming algorithms, such as MINOS, do not explore the sequential time structure of the groundwater water supply or water quality problems. It is possible to reformulate the MINOS problem so that decision (control) variables are defined at each time step. Based on computational complexity analysis, the computational requirement per iteration for second-order nonlinear programming will increase rapidly (typically \( N^R \), for \( 2 \leq R \leq 3 \)) with the number of time periods \( N \), whereas for DDP or SALQR the computational effort will increase only linearly (i.e., in proportion to \( N \)) with the number of time steps [Liao and Shoemaker, 1991]. Since \( N \) is typically in the range of 20–100 for groundwater remediation problems, the control theory algorithm would be expected to be more efficient for computing time-varying pumping policies than general purpose nonlinear programming packages like MINOS.

Because of the nonconvexities inherent in the contaminant transport equation, the solution of the groundwater contamination problem is much more computationally difficult than the groundwater quantity management problem considered by Jones et al. [1987], who point out the convexity of the water quantity problem. Nonconvexity in DDP can be overcome by applying a shift method developed by Liao and Shoemaker [1991].

2. **Numerical Simulation of Contaminant Transport**

To develop a cost-effective design for groundwater remediation, a simulation model is needed to predict the response
of hydraulic head and contaminant transport to the operation of pumping wells. This study will focus on the two-dimensional transient contaminant transport in a confined aquifer. The mathematical model consists of two partial differential equations representing transient effects of time-varying pumping on the flow velocity as well as contaminant advection-dispersion and adsorption [Pinder and Frind, 1972; Pinder and Gray, 1977]. The simulation model is called the transition function in the management model. The hydraulic head and concentration of the pollutant are the state variables, and the pumping rates at each well in each time period are the control variables. In this section, we will define the system of governing equations, outline the procedure of finding numerical solutions and then define the transition function. The flow and contaminant transport equations for a two-dimensional confined aquifer can be expressed as [Ahlfeld, 1987]:

\[ L_h(\vec{h}) = \nabla \cdot (\overline{T} \nabla \vec{h}) + f_a + \sum_{i \in Q} \frac{\partial (x_i, y_i)}{\partial t} \vec{w}_i = 0 \]  

(1)

with boundary condition

\[ \alpha_h \nabla \vec{h}_n = \beta_h (\vec{h} - \vec{h}_b) + \gamma_h (x, y) \in \Gamma \]  

(2)

and initial condition

\[ \vec{h} = \vec{h}_0 (x, y) \]  

(3)

Darcy’s velocity is computed by Darcy’s law:

\[ v = -K \cdot \nabla \vec{h} \]  

(4)

The contaminant transport equation is defined as

\[ L_c(\vec{c}) = \nabla \cdot (b \overline{D} \cdot \nabla \vec{c}) - b \nabla \cdot \nabla \vec{c} - f_a (\vec{c} - c_f) \]  

\[ - \sum_{i \in Q} (x_i, y_i) \delta (x_i, y_i) \theta \partial c \vec{w}_i = 0 \]  

(5)

with boundary condition

\[ \alpha_c \nabla \vec{c}_n = \beta_c (\vec{c} - c_b) + \gamma_c (x, y) \in \Gamma \]  

(6)

and initial condition

\[ \vec{c} = c_0 (x, y) \]  

(7)

where all gradient operators are with respect to the x and y spatial coordinates. The unit of each variable is defined as follows:

\[ \overline{T} \] transmissivity \((L^2/T)\);
\[ b \] saturated thickness of aquifer \((L)\);
\[ K \] hydraulic conductivity tensor \((L/T)\);
\[ \overline{h} \] vertically averaged hydraulic head \((L)\);
\[ f_a \] fluid flux into aquifer through adjacent aquifer (leakage term) \((L/T)\);
\[ Q \] set of pumping sites;
\[ \vec{w}_i \] pumping rate located at \((x_i, y_i)\) \((L^3/T)\);
\[ \delta (x_i, y_i) \] Dirac delta function evaluated at \((x_i, y_i)\);
\[ S_a \] specific storage \((1/L)\);
\[ \nabla \vec{h}_n \] fluid flux normal to the boundary \((L/L)\);
\[ h_b \] head value specified on the boundary \((L)\);
\[ h_0 (x, y) \] initial head specified in the domain \((L)\);

### Table 1. Types of Boundary Conditions

<table>
<thead>
<tr>
<th>Boundary Type</th>
<th>( \alpha_h )</th>
<th>( \beta_h )</th>
<th>( \gamma_h )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dirichlet boundary</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Neumann boundary</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Robin boundary</td>
<td>( \alpha )</td>
<td>( \beta )</td>
<td>0</td>
</tr>
</tbody>
</table>

Here, \( \alpha \) and \( \beta \) are specified constants.

### Equations

\[ v = \text{Darcy’s velocity} \left( \frac{L}{T} \right) \]
\[ \theta = \text{effective porosity} \ (\text{dimensionless}) \]
\[ c = \text{concentration of the contaminant} \ \left( \frac{M}{L^3} \right) \]
\[ c_f = \text{contaminant concentration in leakage fluid} \ \left( \frac{M}{L^3} \right) \]
\[ c' = \text{contaminant concentration in the injection fluid} \ (\text{for extraction, equal } c) \ \left( \frac{M}{L^3} \right) \]
\[ R = \text{retardation coefficient, defined as } (1 + \rho_b K_D/(\theta)) \ (\text{dimensionless}) \]
\[ \rho_b = \text{soil bulk density} \ \left( \frac{M}{L^3} \right) \]
\[ K_D = \text{solute partition coefficient} \ \left( \frac{L^3}{M} \right) \]
\[ \nabla \vec{c}_n = \text{dispersive contaminant flux normal to the boundary} \ \left( \frac{M}{L^4} \right) \]
\[ c_b = \text{specified concentration value in the boundary} \ \left( \frac{M}{L^3} \right) \]
\[ c_0 (x, y) = \text{initial concentration specified in the domain} \ \left( \frac{M}{L^3} \right) \]
\[ \overline{D} = \text{the hydrodynamic dispersion tensor} \ [\text{Scheidegger, 1961}] \ \left( \frac{L^2}{T} \right) \]

For isotropic porous media the components of the dispersion tensor \((\overline{D})\) can be defined as

\[ \overline{D}_{xx} = (d_L u_x^2 + d_T v_x^2)/|v| + d \]  

(8)

\[ \overline{D}_{yy} = (d_L u_y^2 + d_T v_y^2)/|v| + d \]  

(9)

\[ \overline{D}_{xy} = \overline{D}_{yx} = (d_L u_x v_y + d_T v_x u_y)/|v| + d \]  

(10)

where, \( d_L \) is longitudinal dispersivity \((L)\), \( d_T \) is transverse dispersivity \((L)\), and \( d \) is the molecular diffusion coefficient \((L^2/T)\). Equations (1)–(10) completely define the groundwater transport problem. Equations (1) and (5) are the two key partial differential equations that characterize the transient response of groundwater flow and contaminant advection-dispersion to the unsteady pumping. Besides (1) and (5), to properly describe the problem, boundary conditions and initial conditions as shown in (2), (3), (6) and (7) are required. Equations (5), (8), (9), and (10) show the influence of the velocity, through the dispersion, to the solute transport process. In (2), \( \alpha_h \), \( \beta_h \), and \( \gamma_h \) are used to define different types of boundary conditions, and Table 1 shows the relationship between different values of parameters and types of boundary conditions. The units of \( \alpha_h \) and \( \beta_h \) are determined by their associated types of boundary condition. For example, \( \beta_h \) is unitless for the Dirichlet boundary condition, while \( \alpha_h \) is \( L/T \) and \( \beta_h \) is \( L^2/T \) for the Robin boundary condition. Similarly, in (6), \( \alpha_c \), \( \beta_c \), and \( \gamma_c \) are used to define the boundary condition for concentrations and, by changing the subscript \( h \) into \( c \), Table 1 can also be applied.

The numerical solution of (1) and (5) was obtained by applying Galerkin’s finite element method in space and an implicit difference scheme in time. The detailed derivation of the procedure can be found in the works by Pinder and Frind [1972], Pinder and Gray [1977], and Bredehoeft and Pinder.
\[ e_{i,k} = e_{p(i)} = (0, \cdots, 1, 0, \cdots, 0)^T \]

where there are \( n_1 \) elements in the vector and the \( 1 \) is the \( p(i) \) element, and

\[ e_{i,c} = e_{q(i)} = (0, \cdots, 1, 0, \cdots, 0)^T \]

where there are \( n_2 \) elements in the vector and the \( 1 \) is the \( q(i) \) element. A backward difference scheme is used to approximate the time derivative and all the other terms including the pumping rates, velocity and spatial derivatives are written at the new time level. Hence

\[ h_{t+1} = h(t + 1) \Delta t \]
\[ c_{t+1} = c(t + 1) \Delta t \]

\[ \frac{d\tilde{h}}{dt} = \frac{h_{t+1} - h_t}{\Delta t} \]
\[ \frac{d\tilde{c}}{dt} = \frac{c_{t+1} - c_t}{\Delta t} \]

where \( \Delta t \) is the size of the time step. For convenience, \( \Delta t \) is assumed constant. By (11), (17) and Darcy’s law (4), the \( x \) and \( y \) components of approximated Darcy’s velocity \( \tilde{v} \) are calculated by

\[ \tilde{v}_{x,t+1}(x,y) = \sum_{k=1}^{n_x} K_{xx} h_{t+1,k} \frac{\partial w_k(x, y)}{\partial x} \]

\[ \tilde{v}_{y,t+1}(x,y) = \sum_{k=1}^{n_y} K_{yy} h_{t+1,k} \frac{\partial w_k(x, y)}{\partial y} \]

where \( n_x \) is the number of nodes on a specific element. It is assumed that the coordinate system for (19) and (20) has been chosen to diagonalize the conductivity tensor so that \( K_{xy} = K_{yx} = 0 \). For the convenience of computation, the average velocity \( \bar{v} \) for each element is used to evaluate the components of the matrices of the dispersion tensor by

\[ \bar{v}_{x,t+1} = \frac{1}{n_{g_x}} \sum_{l=1}^{n_{g_x}} (\tilde{v}_{x,t+1}(x_l, y_l))^e \]

\[ \bar{v}_{y,t+1} = \frac{1}{n_{g_y}} \sum_{l=1}^{n_{g_y}} (\tilde{v}_{y,t+1}(x_l, y_l))^e \]

where \( n_{g_x} \) is the number of Gaussian integration point and the superscript \( e \) indicates that the quantities are calculated at element \( e \). According to (17)–(21), the time discretization of the ordinary differential equations (15) and (16) is as follows:

\[ ([A] + [B] \Delta t) \{h_{t+1}\} = [B] \Delta t \{h_t\} - [F_h] + [P_h] \{q_{t+1}\} \]

\[ ([\tilde{N} (\tilde{v}_{t+1}), \tilde{v}_{t+1}] + [M] \Delta t) \{c_{t+1}\} = [M] \Delta t \{c_t\} - \{F_c\} \]

\[ - \sum_{i=1}^{m} q_{t+1,i} [P_d] \{c_{t+1} - c\} \]
The derivation of the coefficient matrices in (22) and (23) is given in Appendix A. For the convenience of embedding these equations into the optimization model, (22) and (23) are combined as a single matrix equation, which is the discrete time transition function in the SALQR model. Define the decision variable \( u_t \) at stage \( t \) to be the pumping rate at time \( t + 1 \) or \( u_t = q_{t+1} \). The state variables, \( x_t \), are defined as

\[
\{x_t\} = \begin{cases} h_t \\ c_t \end{cases}
\]  

(24)

The state variables only include head and concentration but not velocity. It can be shown that these two quantities are sufficient to characterize the system by incorporating (19), (20) and (22). From (19), (20) and (21), the velocity \( \mathcal{V}_{t+1} \) is function of \( h_{t+1} \) so that in (23) \( \mathcal{N}(\mathcal{V}_{t+1}) \) can be rewritten as \( \mathcal{N}(h_{t+1}) \). Equation (22) can be rewritten as

\[
\{h_{t+1}\} = (\{A\} + \{B\}/\Delta t)^{-1}(\{B\}/\Delta t h_t - \{F_h\} + \{P_h\}u_t) \]

(25)

From (25), the coefficient matrix \( \mathcal{N}(h_{t+1}) \) can be expressed as \([N(h_t, u_t)]\). Therefore, one can replace the \( \mathcal{N}(\mathcal{V}_{t+1}) \) in (23) with \([N(h_t, u_t)]\) and get a matrix equation that only includes variables \( h_t \), \( c_t \) and \( u_t \). Hence, we can rewrite (23) as

\[
([N(h_t, u_t)] + [M]/\Delta t)\{c_{t+1}\} = \frac{\{M\}}{\Delta t} \{c_t\} - \{F_c\} - \sum_{i=1}^{m} q_{t+1,i}[P_c]((c_{t+1} - c') \]

(26)

Since \( u_t \) are the control variables, it is sufficient to characterize the system by choosing only \( h_t \) and \( c_t \) as state variables. It is possible to include the velocity as one of the state variables but this will increase the computational workload. If velocity is included as one of the state variables, the transition function would involve (22), (23), (19) and (20).

The final step is to express the transition function as a single matrix equation based on (22) and (23). Replace the \( \mathcal{N}(\mathcal{V}_{t+1}) \) in (23) with \([N(h_t, u_t)]\) and define

\[
[P] = \begin{bmatrix} P_h & \cdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \cdots & F_c \end{bmatrix}, \quad \{F\} = \begin{bmatrix} 0 \\ \cdots \\ \sum_{i=1}^{m} u_{t,i}[P_c](c') \end{bmatrix}
\]

(27)

\[
[E(h_t, u_t)] = \begin{bmatrix} [E_{11}] & [E_{12}] \\ [E_{21}] & [E_{22}] \end{bmatrix}
\]

(28)

where

\[
[E_{11}] = \{A\} + \{B\}/\Delta t
\]

\[
[E_{22}] = \{N(h_t, u_t)] + [M]/\Delta t + \sum_{i=1}^{m} u_{t,i}[P_c]
\]

\[
\begin{bmatrix} [E_{12}] = [E_{21}] = 0 \\ [K] = \begin{bmatrix} \{B\}/\Delta t & 0 \\ 0 & \{M\}/\Delta t \end{bmatrix} \end{bmatrix}
\]

(29)

One can rewrite the system equations (22) and (23) in a single matrix form:

\[
[E(h_t, u_t)]\{x_{t+1}\} = \{K\}\{x_t\} + \{P\}u_t - \{F\}
\]

(30)

or

\[
\{x_{t+1}\} = ([E(h_t, u_t)]^{-1}\{K\}\{x_t\} + \{P\}u_t - \{F\}) = T(x_t, u_t)
\]

(31)

The matrix equation (31) is the transition function for the groundwater remediation problem, and it represents the dynamics of groundwater flow and contaminant transport. From the simulation point of view, the pumping rates \( u_t \), as in (30), are prescribed when one uses the model to predict the flow and concentration. This implies that matrix \( E \) is a known constant matrix because \( h_t \) is also specified in the previous time step; consequently, (31) is a linear equation. On the other hand, the optimization model treats the pumping rates as unknown variables that it strives to identify and solve for these unknown variables is an optimal policy. In (31), the inverse of matrix \( ([E(h_t, u_t)]^{-1}) \) involves the unknown variable \( u_t \). This indicates that the transition function is nonlinear. The next section will present the embedding procedure of this transition function into the optimization framework.

3. **Optimization Formulation of the Groundwater Remediation Problem**

Minimizing the cost of groundwater remediation can be formulated as a discrete optimal control problem:

\[
\min_{\{u_t\}_{t=1}^{N}} J(u) = \min_{\{u_t\}_{t=1}^{N}} \left( \sum_{t=1}^{N} g_t(x_t, u_t) \right)
\]

(32)

subject to

\[
x_{t+1} = T(x_t, u_t, t) \quad t = 1, \cdots, N
\]

(33)

\[
f_t(x_t, u_t) \leq 0 \quad t = 1, \cdots, N
\]

(34)

where \( x_t \) is the state variable at time step \( t \), \( u_t \) is the control variable at time step \( t \), \( n \) is the number of state variables, \( m \) is the number of control variables, and \( r_t \) is the number of constraints at time step \( t \). It is assumed that the dimension of control variable is less than or equal to the state variable dimension (\( m \leq n \)). The objective function \( J(u) \) is assumed to be a separable function in time and can be represented by the sum of the \( g_t(x_t, u_t) \), which measure the cost incurred at each time step. The function \( T \) in (33) is called the transition function. It describes the physical laws of the system which will allow us to predict the response of the control policy. Equation (34) contains functional constraints on the acceptable control and state variables at time step \( t \).
Equations (33) and (34) together define the feasible region of the control variable.

For groundwater remediation, the objective function in (32) is the total cost of pumping and treatment. The constraints are the water quality requirements at some locations at the end of the planning period. By choosing the potential pumping sites, the constrained DDP or SALQR algorithm will find the optimal time-varying pumping policy that satisfies the water quality requirement. In our application, the cost function at each time step which appears in (32) is

\[ g_i(x_t, u_t) = \sum_{i \in \Omega} a_1 u_{t,i} + \sum_{i \in \Omega} a_2 u_{t,i}(h_u - h_{t+1,i}) \]

for constants \( a_1 \) and \( a_2 \). In (35), the first term represents water treatment cost, and the second term reflects the pumping cost. The operation cost at each well is assumed to be the product of the extraction rates \( u_t \) and the lift \( (h_u - h_{t+1,i}) \).

The assumption in (35) is that the treatment costs are linear in the amount of water treated, \( c_1 u_{t,i} \), and that the pumping cost has a linear portion, \( c_2 u_{t,i} \), plus a quadratic term, \( c_3 u_{t,i}(h_u - h_{t+1,i}) \). The coefficient \( a_1 \) then equals \( c_1 + c_2 \) and \( a_2 = c_3 \). In this example, \( a_1 \) and \( a_2 \) are assumed to be constant in time, which is equivalent to assuming that the inflation in the cost of \( a_1 \) over time equals the interest rate so that the discounted (in terms of the present value) values of \( a_1 \) and \( a_2 \) are constant in time. For the methodology described in the following section there is no difficulty in using alternative forms of the objective function, including time-varying values of \( a_1 \) and \( a_2 \) or equations that are not quadratic as long as the cost function is twice differentiable in \( x_t \) and \( u_t \). Computationally, it is also helpful if the cost function \( g \) is strictly convex in \( x_t \) and \( u_t \). If it is not, it is necessary to use the shift procedures described by Liu and Shoemaker [1991] which can result in longer computational times than would occur for convex problems. On the other hand, inclusion of fixed costs will result in a nondifferentiable \( g \) which cannot be handled by a gradient-based optimization method like SALQR or MINOS. As a result, none of the previous applications of nonlinear programming to groundwater management [e.g., Ahfeldt et al., 1988a; b; Lee and Kitandis, 1980; Jones et al., 1987] have considered fixed costs. In practice, the effect of fixed costs on optimal solutions can be estimated by repeated solutions of (36)–(38) with varying constraints on the number of active wells or with changes in the shape of the cost function to approximate the (discontinuous) fixed cost function by a differentiable function.

Substituting (35) into (32) and (33) yields the following optimization problem:

\[
\min_{\{u_t\}^N_N} \sum_{t=1}^{N} \left( \sum_{i \in \Omega} a_1 u_{t,i} + \sum_{i \in \Omega} a_2 u_{t,i}(h_u - h_{t+1,i}) \right) \]

subject to

\[
\begin{align*}
\{x_{t+1}\} & = ( [E(h_t, u_t)]^{-1} [K] [x_t] + [P][u_t] + [F] ) \\
& = T(x_t, u_t) \quad t = 1, \cdots, N \\
f_i(x_t, u_t) & = c_{i,j} - c_{\text{max}} \leq 0 \quad t = N, j \in \Phi \\
f_{ij}(x_t, u_t) & = u_t \leq 0 \quad t = 1, \cdots, N
\end{align*}
\]

(38)

where \( f_i \in \mathbb{R}^n \), \( T(x_t, u_t) \in \mathbb{R}^n \) and the variables are defined as follows:

- \( x_{t+1} \) state vector including head \( (h_t) \) and concentration \( (c_t) \), equal to \( (h_{t+1}, c_{t+1})^T \);
- \( h_{t+1} = (h_{t+1,1}, h_{t+1,2}, \cdots, h_{t+1,n})^T \);
- \( c_{t+1} = (c_{t+1,1}, c_{t+1,2}, \cdots, c_{t+1,n})^T \), in which \( h_{t+1,i} \) and \( c_{t+1,i} \) are the head and concentration, respectively, at node \( i, f \in \Psi \);
- \( u_t \) control variables (pumping rates), equal to \( q_{t+1} \);
- \( \Psi \) set of nodes in the finite element model specified by \( T(x_t, u_t) \);
- \( \Omega \) set of pumping nodes, subset of \( \Psi \);
- \( \Phi \) set of observation nodes, subset of \( \Psi \);
- \( n \) total time step;
- \( m \) number of control variables (pumping);
- \( n \) total number of state variables;
- \( q \) number of observation nodes;
- \( r_i \) number of constraints at time \( t \), equal to \( m + 1 \) when \( t = 1, \cdots, N - 1 \); equal to \( m + 1 + q \) when \( t = N \).

The first line of (38) is a constraint that requires the final concentration \( c_{\text{max}} \) to be less than the water quality goal \( c_{\text{max}} \) at all observation wells \( j \in \Phi \). In remediation programs, typically the final goal for water quality \( (c_{\text{max}}) \) and the time allowed to meet the goal \( (N) \) is specified either by the Environmental Protection Agency (EPA) or by the courts. It is not usually specified that intermediate water quality goals must be met and hence none have been included in (36)–(38).

However, the numerical methodology described does allow the inclusion of intermediate water quality goals \( (c_{\text{max}}) \), which would have the form

\[
f_i(x_t, u_t) \leq c_{\text{max}} \quad t = 1, \cdots, N
\]

(38)

This equation would then replace the first line of (38).

In (36)–(38), the pumping rate \( u_{t,i} \) is allowed to change every time period. Culver and Shoemaker [1992] develop a method to use optimal control techniques for a problem in which the pumping rates are restricted to be constant over prespecified management periods.

Equations (36)–(38) constitute an optimization model of groundwater remediation with the structure of a discrete optimal control problem. The groundwater transport finite element model (31) becomes the transition function in (37) that relates the hydraulic head and concentration \( (h_{t+1}, c_{t+1}) \) at stage \( t + 1 \) to the head and concentration \( (h_t, c_t) \) at stage \( t \) given the extraction rates \( u_t \). Equation (38) sets the water quality standard at the end of the planning period and also specifies that the wells extract water, not inject it. Moreover, (38) also limits the total volume of water that has been extracted at each time step. It is also possible to specify that wells can inject water by adding an additional control variable for each injection well. Equations (36)–(38) specify a nonlinear, discrete time, optimal control problem, which will be solved by a constrained optimal control algorithm as discussed in the following section.
4. Constrained Optimal Control With
A Hyperbolic Penalty Function

The control problem given by (36) and (37) becomes an unconstrained problem if (38) is omitted. In this situation, the conventional unconstrained SALQR can be used. By adding (38), equations (36)–(38) become a constrained problem and require a constrained optimal control algorithm. We will follow the approach of Liao and Shoemaker [1990] to convert the constrained optimal control problem given by (36)–(38) into an unconstrained optimal control problem with the use of a penalty function.

The problem (36)–(38) is replaced by the following:

\[
Z = \min_{\{u_i\}_{i=1}^{N}} \left\{ \sum_{i=1}^{N} \left[ g(x_i, u_i) + \sum_{j=1}^{r_i} \gamma_j f_{ij}(x_i, u_i, w_i) \right] \right\}
\]

\[
x_{t+1} = T(x_t, u_t)
\]

where \(y_{ij}\) is a penalty function and \(w_i\) is the penalty factor for the \(i\)th constraint \(f_{ij}\). Liao and Shoemaker’s [1990] approach differs from earlier penalty function methods (such as that used by Andricic and Kitandis [1990]) that result in an ill-conditioned Hessian matrix as the weighting on the penalty increases. Liao and Shoemaker’s procedure avoids these numerical difficulties by partitioning the Hessian matrix and using QR factorization to compute the inverse of the Hessian.

Our work differs from that of Liao and Shoemaker in that we use a different form of the penalty function:

\[
y_i = \xi_i, \quad \xi_i \leq 1
\]

\[
y_i = a \xi_i^2 + b(\xi_i)^{1/2} + c, \quad \xi_i \geq 1
\]

where

\[
\xi_i = (w_i f_{ij}^2 + e_i^2)^{1/2} + f_{ij}
\]

with \(f_{ij}\) being the \(i\)th constraint of (38), \(w_i\) the weighting coefficients of the \(i\)th constraints, and \(e_i\) a shape parameter of the hyperbolic function \(\xi_i\). The penalty function given by (40) and (41) was found to give better results for the groundwater problem than the penalty function \(y_i = \max(0, f_{ij}(x_i, u_i))^2\) used by Liao and Shoemaker [1990]. Appendix B gives additional details on the penalty functions.

The algorithm requires computation of the derivatives \(\partial g/\partial x, \partial g/\partial u, \partial^2 g/\partial x^2, \partial^2 g/\partial u^2, \partial^2 f/\partial x^2, \partial^2 f/\partial u^2, \partial^2 f/\partial x \partial u, \partial T/\partial x, \text{and } \partial T/\partial u\). The first and second derivatives of \(g(x, u_i)\) given by (35) are easy to compute analytically and are not given here. The derivatives of the penalty and transition functions are more complex and are given in Appendices B and C. All of the derivatives are in analytical form based on mathematical differentiation of the transition, objective and penalty function. By contrast, in the nonlinear programming approach used by Gorelick et al. [1984] the derivatives of the transition function must be computed numerically by repeated simulation of the groundwater transport model.

5. Application Examples

General Description

The numerical results given in this section serve two purposes. First, the results indicate that it is possible to compute optimal time-varying pumping strategies for the large-scale problems arising in groundwater remediation. Secondly the results indicate the potential importance of considering time-varying pumping rates rather than the constant pumping rates analyzed in earlier applications of nonlinear programming to groundwater remediation.

The physical setup of the test problem is similar to Ahlfeld [1987]. The example problem is a hypothetical, homogenous, isotropic confined aquifer without leakage \((f_a = 0)\) with dimensions of 900 m by 1500 m. The finite element mesh, associated boundary conditions for both hydraulic head and contaminant concentration, location of potential pumping wells, and location of observation wells are shown in Figure 1. The boundary conditions are intended to represent an underground channel between two surface rivers. The boundaries on the top and bottom of the figure are impermeable. The boundaries on the left and right sides of the figure are connected to the surface water. The initial distribution of the contaminant concentration is shown in Figure 2. The boundary conditions and initial condition are applied to all the following three test examples. Table 2 lists the hydraulic and numerical parameters for all the test cases.

Since we are using an iterative control algorithm, we must make an initial guess of what the values of the \(u_i\) \(i=1, N\) should be. We will call this initial guess the “nominal policy” \(u_i^0\) \(i=1, N\). The speed of convergence of an iterative algorithm can be strongly influenced by the selection of the nominal policy.
TABLE 2. Hydraulic and Numerical Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hydraulic conductivity (K)</td>
<td>0.000431 m/s</td>
</tr>
<tr>
<td>Longitudinal dispersivity (d_x)</td>
<td>70 m</td>
</tr>
<tr>
<td>Transverse dispersivity (d_y)</td>
<td>3 m</td>
</tr>
<tr>
<td>Diffusion coefficient (d)</td>
<td>0.0000001 m^2/s</td>
</tr>
<tr>
<td>Specific storage (S_p)</td>
<td>0.001</td>
</tr>
<tr>
<td>Retardation coefficient (R)</td>
<td>3.60</td>
</tr>
<tr>
<td>Porosity (θ)</td>
<td>0.2</td>
</tr>
<tr>
<td>Aquifer thickness (b)</td>
<td>10 m</td>
</tr>
<tr>
<td>Total volume of water</td>
<td>2.7 × 10^6 m^3</td>
</tr>
<tr>
<td>Area size</td>
<td>900 × 1500 m^2</td>
</tr>
<tr>
<td>Element size</td>
<td>150 × 150 m^2</td>
</tr>
<tr>
<td>Total number of nodes (n_T)</td>
<td>66</td>
</tr>
<tr>
<td>Total number of elements (n_e)</td>
<td>18</td>
</tr>
<tr>
<td>Total potential pumping nodes (control variables) (m)</td>
<td>13</td>
</tr>
<tr>
<td>Number of observation wells (φ)</td>
<td>14</td>
</tr>
<tr>
<td>Dirichlet boundary nodes for head (n_T - n_φ)</td>
<td>14</td>
</tr>
<tr>
<td>Dirichlet boundary nodes for concentration (n_T - n_p)</td>
<td>14</td>
</tr>
<tr>
<td>Total number of state variables (n = n_1 + n_2)</td>
<td>126</td>
</tr>
<tr>
<td>Size of time step (Δt)</td>
<td>2190 hours</td>
</tr>
<tr>
<td>Total time step (N)</td>
<td>20</td>
</tr>
<tr>
<td>Total planning time (NΔt)</td>
<td>5 years</td>
</tr>
</tbody>
</table>

Results of Example Problems

Case 1. In the first example, there are 18 nodes chosen as the potential location of the pumping wells. The spatial distribution of those pumping wells is shown in Figure 1. Figure 1 also shows 13 observation wells that enforce the water quality requirement. In Case 1, the value of u_max in (38) is set sufficiently high so that the last constraint is not binding. The term u_max is binding in the later cases 3, 4 and 5; and in all five cases the nonnegativity constraint on u_1 is binding as well as are the contaminant constraints given by (38). Figure 3 shows the nominal policy assumed as an initial guess for the pumping policy. Selecting the nominal policy is similar to selecting policies for evaluation by trial and error simulation in that we attempt to find the best possible pumping policy based on a reasoning of the dynamics of the system. Since the plume is moving to the right in Figure 1, the nominal policy assumes the wells in operation would initially be the well on the left side of Figure 1 (nodes 18) where the initial concentration is highest and would eventually (in time periods 3–10) include wells in the middle (nodes 25 and 32) and finally (in time periods 15–20) would include only the two wells further right (nodes 32 and 34). The nominal policy also assumes that it would be most effective to pump along the center of the plume where concentrations are higher.

Table 3 shows the results of the optimization analysis. These results indicate that the objective function cost can be greatly reduced by replacing the nominal policy with the optimal policy computed with our SALQR penalty function algorithm applied to (36)–(38). For Case 1 the objective function including the penalty function (Z in (39)) of the optimal value is 62% less than the value of the same objective function for the nominal policy given in Figure 3. The nominal policy did not satisfy all the constraint equations, and part of the difference in the value of the objective function Z is the sum of penalty function y_t associated with the violation of the constraints. The penalty function in (40) and (41) includes a weighting factor w_t. The last row in Table 3 gives the sum of the actual constraint violations. With the nominal policy, the contaminant concentration for node 60 at the end of planning period is 0.72 which is 44% over the water quality standard of 0.5 while with the optimal policy the violation is very small. In fact, for the optimal policy in Figure 3, the maximum violation of the constraints is 4.89 × 10^-2 m^3/s. The values of the objective function without penalty (Σ, g(x_t, u_t)) indicate that the cost for pumping and water treatment are 57% less for the optimal policy than for the nominal policy, which does not even satisfy the constraints. It should be recalled that the nominal policy was selected to be a reasonable guess of what the optimal policy might look like. Many other nominal policies would have even large objective function values and comparison of the optimal policy to them would result in an even larger percentage reduction in cost resulting from the use of the optimization model.

Figure 4 describes the optimal pumping policy associated with optimal policy given in Table 3. Well 18 used in the nominal policy is replaced by wells 17 and 19 in the optimal policy. The other wells 25, 32 and 39 used in the nominal policy (Figure 3) are retained in the optimal policy but the total amount of pumping and the timing is different from the nominal policy.

It is more computationally demanding to solve for time-varying pumping rates than to solve a static problem with time-invariant pumping policies. Hence, one pertinent question is, How much of a reduction in the cost is obtained by including time-varying pumping rates in the analysis rather than

![Fig. 3. Initial policies of case 1.](image-url)
than restricting the pumping rates to those that are constant in time?

To explore the benefits of time-varying pumping, Table 4 compares the optimal time-varying policy to the value of the loss function for several time invariant pumping policies. Policy 1 assumed the same total pumping (summed over time) at each well as for the optimal policy (in Table 3 and Figure 4) but assumed that the pumping was spread evenly over all 20 time periods. Hence in policy 1 the amount of pumping at well \( i \) in time period \( t \) is \((\sum_{t=1}^{20} u_i^t) / 20\) where \( u_i^t \) is the amount of pumping at well \( i \) in time \( t \) in the case 1 optimal policy. Table 4 indicates that this policy cannot satisfy the constraints and results in a high penalty function. Hence the ability to change the pumping rates in time has a major impact on the effectiveness of a policy to detoxify contaminated groundwater. Policies 2 and 3 are also constant pumping policies but with higher pumping rates than policy 1. Table 4 shows that even though these policies increased the pumping and treatment costs \((\Sigma_t g(x_t, u_t))\) (by 5% and 20% respectively), they still violated the water quality constraints, as indicated by the high penalty function value (828 and 626).

Policy 4 in Table 4 is the optimal time invariant pumping policy for case 1 computed by Culver and Shoemaker [1992]. The time invariant policy satisfies the same constraints as for the time-varying optimal policy from Table 3 and Figure 4. However, the best time invariant pumping policy is 75% more expensive than the optimal policy in terms of pumping and water treatment costs \((\Sigma_t g(x_t, u_t))\).

These results support the use of a control algorithm to compute time-varying pumping policies. For situations in which the contaminant plume moves a significant distance during the remediation period, implementing time-varying pumping policies can result in potentially very large cost savings.

**Case 2.** One difficulty with both the nonlinear programming algorithms (used by Gorelick et al. and Ahfeldt et al.) and the control algorithm we are using is that the convergence point may be a local minimum. One pragmatic way to check if the solution is the global minimum is to use different starting values (nominal policies) to see if the procedure converges to the same optimal policy. Case 2 is developed for this purpose.

Case 2 is the same as case 1 except that the nominal policies are different. The differences include the location of the active wells and the time-varying pumping rates for each well. For case 2, the locations of the active wells are 18, 24, 25, 26, 31 and 33 while they are 18, 25, 32 and 39 for case 1. Although wells 18 and 25 are both active in case 1 and case 2, the pumping rates for each well are different between cases 1 and 2 for the nominal policies.

The optimal policy obtained for case 2 is the same as for case 1 (to within \( 10^{-4} \ m^3/s \)). Since the nominal policies for cases 1 and 2 are very different, the result supports the supposition that the policy in Figure 4 is the global minimum.

**Case 3.** One of the drawbacks of a time-varying policy is that the amount of water passing through a water treatment facility will also vary. Our cost function (35) includes a cost for water treatment that is linearly related to the amount of water being treated (for activated carbon units, for example). However, there may also be a capacity constraint on the total amount of water that can be detoxified per unit of time. Let \( u_{\text{max}} \) be the maximum capacity of the water treatment facility. To incorporate this, requires the last type of constraints \((38)\) or

\[
\sum_i |u_{i,t}| \leq u_{\text{max}} \quad \text{for all } t
\]

The example in case 3 is the same as case 1 and case 2 except for the addition of the constraint \((42)\) and for a different nominal policy. For case 3, we use the optimal policy of case 1 as a nominal policy. With this nominal policy, it is expected that all the constraints except \((42)\) should be satisfied. The \( u_{\text{max}} \) is 0.04 \( m^3/s \) for case 3. The sum of actual violation \((\Sigma \sum_{t} (0, f_i(x_t, u_{i,t}))\) is 0.08 (which is only 0.2% of the objective function value). Table 5 and Figure 5 give the objective function values and time-varying pumping rates for the optimization problem specified by \((36)-(38)\) and \((42)\). By comparing Figure 4 and Figure 5, we see that the constraint \((42)\) has caused a redistribution of the pumping in time. This redistribution has had only a small impact on the objective function value. This additional

---

**TABLE 4.** Time-Varying Optimal Pumping Policy Versus Time Invariant Pumping Policies for Case 1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Z)</td>
<td>42.35</td>
<td>945.48</td>
<td>871.80</td>
<td>675.57</td>
<td>74.22</td>
<td>75.0%</td>
</tr>
<tr>
<td>(\Sigma_i g(x_i, u_i))</td>
<td>42.18</td>
<td>41.14</td>
<td>43.23</td>
<td>49.53</td>
<td>74.21</td>
<td>76.0%</td>
</tr>
<tr>
<td>(\Sigma_i \Sigma_i y_{i,i})</td>
<td>0.17</td>
<td>904.34</td>
<td>828.59</td>
<td>626.04</td>
<td>0.007</td>
<td>...</td>
</tr>
</tbody>
</table>

---

**Note:** All values are in units of \( m^3/s \).
TABLE 5. Summary of Nominal and Optimal Policies for Case 3

<table>
<thead>
<tr>
<th>Computation Summary</th>
<th>Nominal Policy</th>
<th>Optimal Policy</th>
<th>Percentage Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective function with penalty $Z$</td>
<td>58.36</td>
<td>43.24</td>
<td>26</td>
</tr>
<tr>
<td>Objective function without penalty ($\sum_i g_i(x_i, u_i)$)</td>
<td>42.18</td>
<td>43.12</td>
<td>-2</td>
</tr>
<tr>
<td>Penalty function ($\sum \psi_i(x_i)$)</td>
<td>16.18</td>
<td>0.12</td>
<td>99</td>
</tr>
<tr>
<td>Sum of constraint violation $(\sum \max(0, f_i(x_i, u_i)))$</td>
<td>0.08</td>
<td>0.</td>
<td>100</td>
</tr>
</tbody>
</table>

restriction does increase the pumping and treatment cost ($\sum_i g_i(x_i, u_i)$) by 2%.

**Case 4.** Case 4 is similar to case 3 except that the former has a smaller value of $u_{\max}$ (from (42)) and hence a more stringent limitation on the total amount of water extracted in a time period. As discussed in Table 4 for case 1, the optimal time invariant policy calculated by Culver and Shoemaker [1992] is 75% more expensive than the optimal time-varying policy without the restriction on total pumping rates. The total pumping rates for this optimal time invariant policy is 0.0348 m$^3$/s. It will be of interest to choose this value as the constraint on total pumping rates and to see what the optimal time-varying pumping policy will be. The results are shown in Table 6. As indicated in Table 6, the cost for time invariant pumping and treatment ($\sum_i g_i(x_i, u_i)$) is 74, which is still 70% more expensive than the optimal time-varying policy, which does not require any additional treatment capacity. Figure 6 further shows the optimal objective function value for constant pumping policy and time-varying policies (cases 1-4) with different $u_{\max}$ in (38). It indicates again that the optimal constant pumping policy is considerably more expensive than the optimal time-varying alternatives.

**Case 5.** Case 5 was developed to demonstrate the capacity of our model for solving a remediation problem that required a much longer pumping period (15 years). The planning periods for cases 1-4 are 5 years. Often remediation requires a decade or more to detoxify a contaminated aquifer, especially for strongly adsorbing chemicals. The basic setup in case 5 is the same as for cases 1-4 except the following: the thickness of aquifer is 20 m instead of 10 m, the conductivity is decreased to 0.000143 m/s and the $u_{\max}$ is set to 0.02 m$^3$/s. Figure 7 gives the time distribution of optimal policies and indicates that, under this condition, pumping is required for the whole planning period. Table 7 summarizes the computational results. The cost for the optimal time invariant pumping policy is approximately 50% greater than the optimal time-varying pumping policy for case 5.

**Computational effort.** Table 8 summarizes the computational effort for the first three cases considered. The number of iterations required is difficult to predict and depends upon the nominal policy. Usually, the closer the nominal policy is to the optimal policy, the fewer iterations are required. For example, the nominal policy for case 1 is closer to the optimal policy than the nominal policy for case 2 and the number of iterations required for case 1 is much smaller. The total CPU time required is approximately a linear function of the number of iterations. The addition of extra constraints also tends to increase computational effort.

**Computational analysis.** Cases 1-3 given in this paper were implemented on an IBM 3090 600E V/F while cases 4-5 were computed on an upgraded machine (IBM ES 3090 6001 V/F), all with the vector computation facility. The backward sweep of DDP usually requires more computational effort than the forward finite element simulation. For case 4, as shown in Table 9, 1.9 hours of CPU time were required and 63% of this time was spent in the backward sweep. For case 5, as shown in Table 10, 5.17 hours of CPU time were

![Fig. 5. Optimal policies of case 3.](image)

![Fig. 6. Optimal objective values of different cases.](image)
required and the backward sweep took 53% of this time. The memory requirements for cases 1–4 are 70 megabytes. The times reported are for the grid given in Figure 1. Computational times per iteration are expected to increase as a nonlinear function of the number of nodes in the finite element mesh. This rate of increase is expected to be greater than $O(n)$ and equal to or less than $O(n^3)$ [Liao and Shoemaker, 1991] where $n$ is the number of degrees of freedom in the finite element model.


Our results in Tables 3–6 indicate that time-varying pumping policies can be much more cost-effective than time invariant pumping policies. However, Table 8 indicates that the computational effort is significantly greater for time-varying pumping optimization problems than that required for the time invariant pumping optimization considered by Gorelick et al. [1984] and Ahlfeldt et al. [1988a, b].

Our results show that for the example we considered, the best time invariant policy is 43–75% more expensive than the best time-varying pumping policy. Given that remediation programs can cost tens or hundreds of millions of dollars, a savings of even a few percentage points justifies a sizable computer budget. An hour of supercomputer time currently costs of the order of $100 to $200 per hour for commercial users and this cost is expected to fall. Hence the cost for computing the solution in Tables 3–6 is well justified for larger remediation projects that require long periods of pumping. As workstations become more powerful, they can also be used to solve these problems.

However, for cases in which the plume is not expected to move very far during the period required for cleanup, a time invariant analysis may be adequate. Hence a decision about whether to use the algorithm discussed here versus a static nonlinear programming algorithm depends upon the characteristics of the problem. Ahlfeldt et al. [1988b] consider only 3 years of pumping in the Woburn case they analyzed. The water quality constraints considered in their analysis were such that they could be satisfied within 3 years of pumping during which time the plume may not have moved far enough to require adjustments in pumping rates. For cases in which cleanup is feasible during a short enough time period that the plume does not move very far, then the approach suggested by Gorelick et al. [1984] and Ahlfeldt et al. [1988a, b] for using a nonlinear programming algorithm like MINOS is adequate.

However, many large groundwater remediation programs are expected to require many years to reach satisfactory water quality standards. The length of pumping period is in part determined by the sorption and degradation properties of the contaminant. Chemicals with low degradation rates and high adsorption/desorption rates will generally require longer periods of pumping to decontaminate an aquifer. Hence the additional computational effort required for computing time-varying pumping rates with our control theory

---

**TABLE 7. Summary of Nominal and Optimal Policies for Case 5**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective with penalty $Z$</td>
<td>163.72</td>
<td>116.16</td>
<td>80.85</td>
<td>44</td>
</tr>
<tr>
<td>Objective without penalty $\Sigma g(x_t, u_t)$</td>
<td>74.25</td>
<td>116.13</td>
<td>79.68</td>
<td>46</td>
</tr>
<tr>
<td>Penalty function ($\sum y_{t,i}$)</td>
<td>88.47</td>
<td>0.035</td>
<td>0.17</td>
<td>...</td>
</tr>
</tbody>
</table>
TABLE 8. Computation Summary of Optimal Policies for Three Cases

<table>
<thead>
<tr>
<th>Computation Summary</th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of iterations</td>
<td>368</td>
<td>711</td>
<td>883</td>
</tr>
<tr>
<td>Total CPU time, hours</td>
<td>1.86</td>
<td>3.01</td>
<td>3.45</td>
</tr>
</tbody>
</table>

The approach is worthwhile for cases where the distribution of the contaminant, its speed of travel and its high sorption and/or low degradation rates indicate that pumping rates at the wells should vary over the time period required for remediation.

As indicated in Table 2 and equations (1)–(10), the numerical results are based on a model of a dissolved contaminant in a homogeneous isotropic confined aquifer with sorption described by a linear isotherm. At many remediation sites one or more of these assumptions is violated.

However, the methodology we have described here is applicable to more complex situations. Heterogeneity, for example, is incorporated by simply changing conductivities in space. Some of the other complexities, such as a nonlinear isotherm or unconfined aquifer, change the basic equations (1)–(10), and hence will result in changes in the derivatives used in the optimization model (given in the appendices). Other factors such as in situ bioremediation require the incorporation of additional variables in the finite element model and hence increase the state variable dimension, \( n \), and the associated computational effort. These are topics for future research projects.

The actual increase in computational times associated with larger grids is expected to be dependent on the extent to which sparsity in the derivative matrices can be incorporated into the methodology. Future topics for research in this area include the exploration of sparsity and computation of numerical results for a range of grid sizes to evaluate the actual increase in computation time associated with larger grids.

7. Conclusion

We have shown that the constrained optimal control algorithm presented here is a computationally feasible method for calculating optimal time-varying pumping rates for groundwater remediation. Computation of optimal time-varying pumping rates is more computationally demanding than that of time invariant pumping rates. The numerical results in Tables 3–6 indicate that time-varying pumping policies can be much more economical than time invariant pumping policy for situations in which the plume boundaries move a significant distance relative to the radius of influence of the pumping wells.

TABLE 9. CPU Time for Case 4

<table>
<thead>
<tr>
<th>Overall</th>
<th>Forward Sweep</th>
<th>Backward Sweep</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total CPU time, hours</td>
<td>1.90</td>
<td>0.71</td>
</tr>
<tr>
<td>Percent of total CPU</td>
<td>100</td>
<td>37</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Overall</th>
<th>Forward Sweep</th>
<th>Backward Sweep</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total CPU time, hours</td>
<td>5.17</td>
<td>2.44</td>
</tr>
<tr>
<td>Percent of total CPU</td>
<td>100</td>
<td>47</td>
</tr>
</tbody>
</table>

APPENDIX A: DEFINITION OF COEFFICIENT MATRICES FOR TRANSITION FUNCTION

The definitions of all the coefficient matrices for (22) and (23) at each element are as follows:

\[
[A]_j^e = \left( \int \int \left( \hat{T}_{xx} \frac{\partial w_i}{\partial x} \frac{\partial w_j}{\partial x} + \hat{T}_{yy} \frac{\partial w_i}{\partial y} \frac{\partial w_j}{\partial y} \right) dx \, dy \right)^e
\]

\[
[B]_j^e = \left( \int \int S w_i w_j dx \, dy \right)^e
\]

\[
\{F_h\}_i^e = \left( - \int \int w_i f_a dx \, dy - \int \Gamma \right) \\
\cdot \sum_{j=1}^{n_d} \left( \hat{T}_{xx} \frac{\partial w_j}{\partial x} n_x + \hat{T}_{yy} \frac{\partial w_j}{\partial y} n_y \right) h_j \cdot dl \right)^e
\]

\[
[N]_{ij}^e = \left( \int \int \left( \hat{D}_{xx}(\hat{v}_t+1) \frac{\partial w_i}{\partial x} \frac{\partial w_j}{\partial x} + 2\hat{D}_{xy}(\hat{v}_t+1) \frac{\partial w_i}{\partial y} \frac{\partial w_j}{\partial y} \right. \right. \\
+ \left. \left. \hat{D}_{yy}(\hat{v}_t+1) \frac{\partial w_i}{\partial y} \frac{\partial w_j}{\partial y} \right) \right)^e
\]

\[
[M]_{ij}^e = \left( \int \int (b \theta + \rho_n b K_d) w_i w_j dx \, dy \right)^e
\]

\[
\{F_c\}_i^e = \left( - \int \Gamma (b \theta \hat{D} \cdot \Delta c) \cdot n w_i \, dl - \int \int f_a c \rho_n w_i dx \, dy \right)^e
\]

where

\( \hat{T} \) transmissivity, equal to \( bK \);
\( n_d \) number of nodes at element \( e \);
\( n \) unit normal vector to the boundary;
\( n_x, n_y \) directional cosines between \( n \) and \( x, y \) coordinate axes;
\( \hat{D}(\hat{v}_t+1) = b\hat{D}(\hat{v}) \).

The superscript \( e \) indicates that the quantity is defined at element \( e \) and the global coefficient matrix in (22) and (23) is assembled from those local element matrices. Define \( M(e, k) \) as the mapping of the local node number \( k \) in element \( e \) to its global node number, e.g., \( M(e, k) = q \) represents that the global node number \( q \) is corresponding to the local node \( k \) in...
element \( e \). The global matrix \( N \) is then assembled from each element matrix \( N_e \), such as

\[
[N]_{ij} = \sum_{e=1}^{n_e} \sum_{r=1}^{n_\varphi} \sum_{s=1}^{n_d} [N]^e_{rs} \delta_{\varphi(r,e),i} \delta_{\varphi(s,e),j} \tag{48}
\]

where \( n_e \) is the total number of elements and \( n_d \) is the total number of nodes in element \( e \).

**Appendix B: Derivatives of the Penalty Function**

**First Derivatives of the Penalty Function**

Successful implementation of the optimal control algorithm requires computation of derivatives of the objective function, penalty function and transition equations. We have used the approach of computing the derivatives analytically directly from the equations rather than numerically as suggested by Gorelick et al. [1984]. These derivatives are presented in Appendices B and C.

Recall that the definition of the penalty function is

\[
y_i = \xi_i \quad \xi_i \leq \xi_0 \tag{49}
\]

\[
y_i = a \xi_i^n + b \xi_i^m + c \quad \xi_i \geq \xi_0
\]

with

\[
\xi_i = (w_i f_i^2 + t^2)^{1/2} + w_i f_i
\]

\[
a = \frac{m - 1}{n(m - n) \xi_0} \tag{50a}
\]

\[
b = \frac{n - 1}{m(m - n) \xi_0} \tag{50b}
\]

\[
c = \left( 1 - \frac{m - 1}{n(m - n)} - \frac{n - 1}{m(m - n)} \right) \xi_0 \tag{50c}
\]

where \( a \), \( b \), and \( c \) are chosen so that \( y_i \), \( \partial y_i / \partial \xi_i \), and \( \partial^2 y_i / \partial \xi_i^2 \) are continuous at \( \xi_i = \xi_0 \). The \( f_i \) is the \( i \)th constraint of the constraints vector \( (f \leq 0) \). In this paper, \( \xi_0 = 1 \), \( n = 2 \), and \( m = 0.5 \) leads to \( a = 1/3 \), \( b = 4/3 \), \( c = -2/3 \). Function \( y_i \), then equals

\[
y_i = \xi_i \quad \xi_i \leq 1
\]

\[
y_i = \frac{1}{3} \xi_i^2 + \frac{4}{3} \xi_i^{1/2} - \frac{2}{3} \xi_i \quad \xi_i \geq 1
\]

In (50a), if \( w_if_i \leq 0 \), the following equation for \( \xi_i \), instead of (50a), is suggested

\[
\xi_i = \frac{t^2}{(w_i f_i^2 + t^2)^{1/2} - w_i f_i} \tag{52}
\]

Mathematically, (50) is equivalent to (52), but (52) can avoid machine rounding error when \( w_if_i \leq 0 \).

Using first derivatives of the penalty function, equation (49), the first derivatives of the penalty function can be computed as follows: When \( \xi_i \leq \xi_0 \),

\[
\frac{\partial y_i}{\partial f_i} = \frac{\partial y_i}{\partial \xi_i} = \frac{w_i \xi_i}{(w_i^2 f_i^2 + t^2)^{1/2}} \tag{53}
\]

When \( \xi_i > \xi_0 \),

\[
\frac{\partial y_i}{\partial f_i} = \frac{\partial y_i}{\partial \xi_i} = \frac{w_i \xi_i}{(w_i^2 f_i^2 + t^2)^{1/2}} \tag{54}
\]

\[
\frac{\partial y_i}{\partial f_i} = \frac{\partial y_i}{\partial \xi_i} = \frac{\partial y_i}{\partial \xi_i} (\frac{w_i \xi_i}{(w_i^2 f_i^2 + t^2)^{1/2}}) \tag{55}
\]

\[
\frac{\partial y_i}{\partial f_i} = \frac{\partial y_i}{\partial \xi_i} = \frac{\partial y_i}{\partial \xi_i} (\frac{w_i \xi_i}{(w_i^2 f_i^2 + t^2)^{1/2}}) \tag{56}
\]

Substituting the values of \( a, b, m, n \) into (56), we obtain

\[
\frac{\partial y_i}{\partial f_i} = 2a \xi_i + 0.5b \xi_i^{1/2} \frac{w_i \xi_i}{(w_i^2 f_i^2 + t^2)^{1/2}} \tag{57}
\]

**Second Derivatives of the Penalty Function**

When \( \xi_i \leq \xi_0 \),

\[
\frac{\partial^2 y_i}{\partial f_i^2} = \frac{\partial^2 y_i}{\partial \xi_i^2} = \frac{w_i^2 t^2}{(w_i^2 f_i^2 + t^2)^{3/2}} \tag{58}
\]

When \( \xi_i > \xi_0 \),

\[
\frac{\partial^2 y_i}{\partial f_i^2} = \frac{\partial^2 y_i}{\partial \xi_i^2} = \frac{\partial^2 y_i}{\partial \xi_i^2} + \frac{\partial^2 y_i}{\partial \xi_i^2} \frac{(\partial y_i)^2}{\partial f_i^2} \tag{59}
\]

\[
\frac{\partial^2 y_i}{\partial f_i^2} = (an \xi_i^{n-1} + bm \xi_i^{m-1}) \frac{w_i^2 t^2}{(w_i^2 f_i^2 + t^2)^{3/2}} + (an(n - 1) \xi_i^{n-2} + bm(m - 1) \xi_i^{m-2}) \left( \frac{w_i \xi_i}{(w_i^2 f_i^2 + t^2)^{1/2}} \right)^2 \tag{60}
\]

and it equals

\[
\frac{\partial^2 y_i}{\partial f_i^2} = (2a \xi_i + 0.5b \xi_i^{1/2}) \frac{\partial^2 y_i}{\partial \xi_i^2} - \left( \frac{0.25b}{\xi_i^{3/2}} \right) \left( \frac{(\partial y_i)^2}{\partial f_i^2} \right)^2 \tag{61}
\]

**Appendix C: Derivatives of the Transition Function**

**First Derivatives of the Transition Function to State Variables** \((\partial T(x_i, u_i)/\partial x_i)\)

Recall that the transition function is composed by the simulation model (in what follows, \([L] = [P]_\alpha \) as used in (25)):

\[
([A] + [B](\Delta t)(h_{t+1}) = \frac{[B]}{\Delta t} \{h_t\} - \{F\} + [L]_\alpha [u]
\]

\[
([N(h_{t+1}, u_i)] + [M]/\Delta t)(c_{t+1}) = \frac{M}{\Delta t} \{c_t\} - \{F_c\}
\]

\[
- \sum_{i=1}^{m} u_{i}(c_{t+1} - c_i)[P_i] \tag{63}
\]
For the convenience of deriving the derivatives of the transition function, we define

\[ [E_1] = [A] + \begin{bmatrix} B \end{bmatrix}/\Delta t \] (64)

\[ [E_2(h_i, u_i)] = [N(h_i, u_i)] + [M]/\Delta t \] (65)

\[ [E_3(h_i, u_i)] = [E_2(h_i, u_i)] + \sum_{i=1}^{m} u_{i,i[P^i]} \] (66)

and get

\[ [E_1][h_{t+1}] = \frac{[B]}{\Delta t} \{ h_i \} - \{ F_c \} + [L_k]{u_i} \] (67)

\[ [E_2(h_i, u_i)]\{c_{t+1}\} = \frac{M}{\Delta t} \{ c_i \} - \{ F_c \} \]

\[ - \sum_{i=1}^{m} u_{i,i}[c_{t+1,i} - c_i][P^i] \] (68)

Equation (68) can be rewritten as

\[
\left( N(h_i, u_i) + \frac{M}{\Delta t} + \sum_{i=1}^{m} u_{i,i}[P^i] \right) c_{t+1} = \left( M/\Delta t \right) c_t + \sum_{i=1}^{m} u_{i,i}[P^i] c' - F_c
\]

which can be expressed as

\[ (E_3(h_i, u_i))c_{t+1} = (M/\Delta t)c_t + \sum_{i=1}^{m} u_{i,i}[P^i] c' - F_c \] (69)

By definition,

\[
\frac{\partial T}{\partial x_t} = \frac{\partial x_{t+1}}{\partial x_t}
\]

\[
\frac{\partial h_{t+1}}{\partial h_t} = \frac{\partial h_{t+1}}{\partial h_t}
\]

\[
\frac{\partial c_{t+1}}{\partial c_t} = \frac{\partial c_{t+1}}{\partial c_t}
\] (70)

By (62), one can easily get

\[
\frac{\partial h_{t+1}}{\partial h_t} = \left[ A + \frac{B}{\Delta t} \right]^{-1} \left[ B \right]
\]

\[
\frac{\partial c_{t+1}}{\partial c_t} = 0
\] (71)

By (63),

\[
\frac{\partial c_{t+1}}{\partial c_t} = \left[ E_3(h_t, u_t) \right]^{-1} \left[ M/\Delta t \right]
\]

The computation of \( \left[ \partial c_{t+1}/\partial h_t \right]_{m,n} \) is more complicated than the others, and is derived by a series of steps starting with (71), where \( m \) is the number of control variables (potential pumping wells). Take the derivatives of (71) to get

\[ \frac{\partial N(h_i, u_i)c_{t+1}}{\partial h_{t,j}} + \frac{\partial \left( N(h_{t+1}) \right)_{ij}}{\partial h_{t,j}} \left( \sum_{k=1}^{n_n} N_{ik} c_{t+1,k} \right) = 0 \] (72)

In (75), \( \partial N(h_i, u_i)c_{t+1}/\partial h_{t,j} \) is computed by

\[ \frac{\partial N_{ik} c_{t+1,k}}{\partial h_{t,j}} = \sum_{k=1}^{n_n} \left( N_{ik} c_{t+1,k} \right) \frac{\partial N_{ik}}{\partial h_{t,j}} + \sum_{k=1}^{n_n} N_{ik} \frac{\partial c_{t+1,k}}{\partial h_{t,j}} \] (73)

Equation (68) can be rewritten as

\[
\left( N(h_i, u_i) + \frac{M}{\Delta t} + \sum_{i=1}^{m} u_{i,i}[P^i] \right) c_{t+1} = \left( M/\Delta t \right) c_t + \sum_{i=1}^{m} u_{i,i}[P^i] c' - F_c
\]

which can be expressed as

\[ (E_3(h_i, u_i))c_{t+1} = (M/\Delta t)c_t + \sum_{i=1}^{m} u_{i,i}[P^i] c' - F_c \] (69)

By definition,

\[ \frac{\partial T}{\partial x_t} = \frac{\partial x_{t+1}}{\partial x_t} \]

\[
\frac{\partial h_{t+1}}{\partial h_t} = \frac{\partial h_{t+1}}{\partial h_t}
\]

\[
\frac{\partial c_{t+1}}{\partial c_t} = \frac{\partial c_{t+1}}{\partial c_t}
\] (70)

By (62), one can easily get

\[
\frac{\partial h_{t+1}}{\partial h_t} = \left[ A + \frac{B}{\Delta t} \right]^{-1} \left[ B \right]
\]

\[
\frac{\partial c_{t+1}}{\partial c_t} = 0
\] (71)

By (63),

\[
\frac{\partial c_{t+1}}{\partial c_t} = \left[ E_3(h_t, u_t) \right]^{-1} \left[ M/\Delta t \right]
\]

Hence \( [R] \) is computed by

\[ [R]_{ij} = \sum_{k=1}^{n_n} \frac{\partial N_{ik}}{\partial h_{t,j}} c_{t+1,k} \] (78)

By (77), (78), and (79),

\[ \left[ \frac{\partial N(h_i, u_i)c_{t+1}}{\partial h_t} \right]_{m,n_1} = R_{m,n_1} + N_{m,n_2} \left[ \frac{\partial c_{t+1}}{\partial h_t} \right]_{n_2,n_1} \] (80)

In (80), \( R \) is computed by

\[ [R]_{ij} = \sum_{k=1}^{n_n} \frac{\partial N_{ik}}{\partial h_{t,j}} c_{t+1,k} \]
$$[R] = \left[ \sum_{i=1}^{n^1} \frac{\partial \tilde{N}}{\partial h_{t+1,i,k}} \right] e \left[ \frac{\partial h_{t+1,i,k}}{\partial h_t} \right]$$

$$= \left[ \sum_{i=1}^{n^1} \frac{\partial \tilde{N}}{\partial h_{t+1,i,k}} \right] e \left[ \frac{[E_1]^{-1} [B]}{\Delta t} \right]$$

(82)

where $\partial N/\partial h_{t+1,i,k}$ will be defined later. Substituting (80) into (75), we get

$$R + \left( N(h_t, u_t) + M + \sum_{i=1}^{m} u_{i,[P^1]} \right) \frac{\partial c_{i+1}}{\partial h_t} = 0$$

(83)

$$\frac{\partial c_{i+1}}{\partial h_t} = -\left( N(h_t, u_t) + M + \sum_{i=1}^{m} u_{i,[P^1]} \right)^{-1} R$$

$$= -[E_3(h_t, u_t)]^{-1} R$$

(84)

where $R$ is defined by (82). Equations (72), (73), (74), and (84) complete the computation of $\partial T(x_t, u_t)/\partial x_t$.

**Computation of $\partial \tilde{N}/\partial h_{t+1,i,k}$**

In order to complete the computation of $R$ as given in (82) the following derivation is required. Recall that $\tilde{N}(h_{t+1})$ is rewritten from $\tilde{N}(\tilde{D}(\tilde{s}))$ and it is

$$\left[ \tilde{N} \right]_{ij} = \left( \int \int \left( D_{xx} \frac{\partial w_i}{\partial x} \frac{\partial w_j}{\partial y} + 2D_{xy} \frac{\partial w_i}{\partial x} \frac{\partial w_j}{\partial y} + \sum_{k=1}^{n^2} \frac{\partial w_i}{\partial h_{t+1,k}} \frac{\partial w_j}{\partial h_{t+1,k}} \right) w_i \frac{\partial w_j}{\partial x} \right) \left( \sum_{k=1}^{n^2} \frac{\partial w_i}{\partial h_{t+1,k}} \frac{\partial w_j}{\partial h_{t+1,k}} \right)$$

$$+ \left( \sum_{k=1}^{n^2} \frac{\partial w_i}{\partial h_{t+1,k}} \frac{\partial w_j}{\partial h_{t+1,k}} \right) \left( \sum_{k=1}^{n^2} \frac{\partial w_i}{\partial h_{t+1,k}} \frac{\partial w_j}{\partial h_{t+1,k}} \right)$$

(85)

and $D(h_{t+1})$ is defined as $\tilde{D}(\tilde{c}(h_{t+1}))$ where $\tilde{D}$ and $\tilde{s}$ are given in (21) and (8)-(10). Taking the derivative of (85) with respect to $h_{t+1,k}$, we get

$$\frac{\partial \tilde{N}_{ij}}{\partial h_{t+1,k}} = \left( \int \int \left( \frac{\partial D_{xx}}{\partial h_{t+1,k}} \frac{\partial w_i}{\partial x} \frac{\partial w_j}{\partial y} + 2\frac{\partial D_{xy}}{\partial h_{t+1,k}} \frac{\partial w_i}{\partial x} \frac{\partial w_j}{\partial y} + \frac{\partial D_{yy}}{\partial h_{t+1,k}} \frac{\partial w_i}{\partial x} \frac{\partial w_j}{\partial y} \right) \left( \sum_{k=1}^{n^2} \frac{\partial w_i}{\partial h_{t+1,k}} \frac{\partial w_j}{\partial h_{t+1,k}} \right)$$

$$+ \left( \sum_{k=1}^{n^2} \frac{\partial w_i}{\partial h_{t+1,k}} \frac{\partial w_j}{\partial h_{t+1,k}} \right) \left( \sum_{k=1}^{n^2} \frac{\partial w_i}{\partial h_{t+1,k}} \frac{\partial w_j}{\partial h_{t+1,k}} \right)$$

(86)

where

$$\frac{\partial D_{xx}}{\partial h_{t+1,k}} = \left( \frac{\partial \tilde{D}_{xx}}{\partial h_{t+1,k}} + \frac{\partial \tilde{D}_{xy}}{\partial h_{t+1,k}} \frac{\partial \tilde{s}_y}{\partial h_{t+1,k}} \right)$$

$$\frac{\partial D_{xy}}{\partial h_{t+1,k}} = \left( \frac{\partial \tilde{D}_{xx}}{\partial h_{t+1,k}} \frac{\partial \tilde{s}_x}{\partial h_{t+1,k}} + \frac{\partial \tilde{D}_{xy}}{\partial h_{t+1,k}} \frac{\partial \tilde{s}_y}{\partial h_{t+1,k}} \right)$$

(87)

**First Derivatives of Transition Function About Control Variables ($\partial T(x_t, u_t)/\partial u_t$)**

By definition

$$\left[ \frac{\partial T}{\partial u_t} \right] = \left[ \begin{array}{c} \frac{\partial h_{t+1}}{\partial u_t} \\ \vdots \\ \frac{\partial c_{i+1}}{\partial u_t} \\ \frac{\partial \tilde{u}_t}{\partial u_t} \end{array} \right]$$

(88)

The derivation of $[\partial c_{i+1}/\partial u_t]$ is similar to the derivation of $[\partial c_{i+1}/\partial h_t]$. Taking the derivative of (70) with respect to $u_r$, we get

$$\frac{\partial (N(h_t, u_t)c_{i+1})}{\partial u_t} + \frac{M}{\Delta t} \frac{\partial c_{i+1}}{\partial u_t} + \frac{\partial}{\partial u_t} \left( \sum_{i=1}^{m} u_{i,[P^1]} c_{i+1} \right) = \sum_{i=1}^{m} [P^1] c_{i+1}$$

(89)
\[
\frac{\partial N(h_{i}, u_{i})c_{i+1}}{\partial u_{i}} = V + N \frac{\partial c_{i+1}}{\partial u_{i}} \quad (90)
\]

\[
V = \left[ \sum_{k=1}^{n_2} \frac{\partial N_{ik}}{\partial u_{i}^2} c_{k+1}^{T} \right] = \sum_{i=1}^{n_1} \left( \sum_{l=1}^{n_1} \frac{\partial h_{l+1}^i}{\partial u_{i}} \right) c_{i+1}^{T} \quad (91)
\]

Similar to the derivation of (82) \( V \) can be computed by

\[
[V] = \left[ \sum_{i=1}^{n_1} \frac{\partial N}{\partial h_{l+1}^i} c_{l+1}^T \right] \frac{\partial h_{l+1}^i}{\partial u_{i}}
\]

\[
= \left[ \sum_{i=1}^{n_1} \frac{\partial N}{\partial h_{l+1}^i} c_{l+1}^T \right] [E_3]^{-1} L_i
\]

where \( \partial h_{l+1}^i / \partial u_{i} \) is computed by (87) and \( \partial N_{ik} / \partial h_{l+1}^i \) is defined in the computation of \( \partial T / \partial x_i \). Substituting (89) and (90) into (88) and rearranging the results, we obtain

\[
\left( N + \frac{M}{\Delta t} + \sum_{i=1}^{m} u_{i} \left[P_i \right] \right) \frac{\partial c_{i+1}}{\partial u_{i}} = -V - \sum_{i=1}^{m} \left[P_i \right] c_{i+1} e_{i}^T + \sum_{i=1}^{m} \left[P_i \right] c_{i}
\]

\[
\frac{\partial c_{i+1}}{\partial u_{i}} = \left( N + \frac{M}{\Delta t} + \sum_{i=1}^{m} u_{i} \left[P_i \right] \right)^{-1} \left( -V - \sum_{i=1}^{m} \left[P_i \right] (c_{i+1} - c_i) e_{i}^T \right)
\]

\[
\frac{\partial c_{i+1}}{\partial u_{i}} = [E_3(h_i, u_i)]^{-1} \left( -V - \sum_{i=1}^{m} \left[P_i \right] (c_{i+1} - c_i) e_{i}^T \right)
\]

Equations (87) and (95) define the computation of \( \partial T(x, u_{i}) / \partial u_{i} \). Dealing with the case where \( u_{i} \) is time invariant, Sykes et al. [1985] and Ahfeld et al. [1988a, b] obtained similar results by adjoint sensitivity theory.

Acknowledgments. This research was funded primarily by a grant from the U.S. Geological Survey to C.A.S. and P.L.-F.L. and a grant from NSF to C.A.S. The optimization code used in the analysis was modified from FORTRAN code provided by S. Yakowitz, T. Culver and G. Whiffin provided computational assistance. Computer time and the assistance of systems analyst Hugh Caffey were provided by the Cornell National Supercomputer Facility (CNSF) through a Strategic User Award to C.A.S. C. Pelkie of CNSF staff produced animated computer graphics describing optimal pumping rates and associated contaminant movement over the 60 time periods in Figure 7. A narrated video tape, “Optimization of Groundwater Remediation With Time-Varying Pumping,” including these graphics is available for $15 by writing to C.A.S. The CNSF is funded by NSF, IBM, New York State and the Corporate Institute.

References


L.-C. Chang, Department of Civil Engineering, National Chiao Tung University, Hsinchu, Taiwan, Republic of China.

(Received July 23, 1990; revised July 8, 1992; accepted July 15, 1992.)