A least-squares penalty method algorithm for inverse problems of steady-state aquifer models

Hailong Li *, Qichang Yang

Institute of Biomathematics, Anshan Normal College, Anshan 114005, People’s Republic of China

Abstract

Based on the generalized Gauss–Newton method, a new algorithm to minimize the objective function of the penalty method in (Bentley LR. Adv Wat Res 1993;14:137–48) for inverse problems of steady-state aquifer models is proposed. Through detailed analysis of the “built-in” but irregular weighting effects of the coefficient matrix on the residuals on the discrete governing equations, a so-called scaling matrix is introduced to improve the great irregular weighting effects of these residuals adaptively in every Gauss–Newton iteration. Numerical results demonstrate that if the scaling matrix equals the identity matrix (i.e., the irregular weighting effects of the coefficient matrix are not balanced), our algorithm does not perform well, e.g., the computation cost is higher than that of the traditional method, and what is worse is the calculations fail to converge for some initial values of the unknown parameters. This poor situation takes a favourable turn dramatically if the scaling matrix is slightly improved and a simple preconditioning technique is adopted: For naturally chosen simple diagonal forms of the scaling matrix and the preconditioner, the method performs well and gives accurate results with low computational cost just like the traditional methods, and improvements are obtained on: (1) widening the range of the initial values of the unknown parameters within which the minimizing iterations can converge, (2) reducing the computational cost in every Gauss–Newton iteration, (3) improving the irregular weighting effects of the coefficient matrix of the discrete governing equations. Consequently, the example inverse problem in Bentley (loc. cit.) is solved with the same accuracy, less computational effort and without the regularization term containing prior information on the unknown parameters. Moreover, numerical example shows that this method can solve the inverse problem of the quasilinear Boussinesq equation almost as fast as the linear one.

In every Gauss–Newton iteration of our algorithm, one needs to solve a linear least-squares system about the corrections of both the parameters and the groundwater heads on all the discrete nodes only once. In comparison, every Gauss–Newton iteration of the traditional method has to solve the discrete governing equations as many times as one plus the number of unknown parameters or head observation wells (Yeh WW-G. Wat Resour Res 1986;22:95–108).

All these facts demonstrate the potential of the algorithm to solve inverse problems of more complicated non-linear aquifer models naturally and quickly on the basis of finding suitable forms of the scaling matrix and the preconditioner. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Penalty method; Inverse problem; Scaling matrix; Preconditioner; Aquifer models; LSQR algorithm; Parameter identification; Gauss–Newton method; Finite element method

1. Introduction

The traditional least-square methods for distributed parameter identification of groundwater models minimize a quadratic objective function with respect to the parameters subject to the discrete governing equations of an elliptic or parabolic aquifer model (see e.g., [4–6,8,9,14–16,21]). This means that the discrete equations of the aquifer models are regarded as an exact description of the real aquifer. But in practice there exist not only measurement errors of heads but also errors of the conceptual models, of the discretization of the models, of the determination of boundary conditions and source terms etc. Hence the concept of measurement error must be generalized such that it includes also the other errors mentioned above. Such a generalization of the measurement error concept is particularly important in least-squares and maximum likelihood methods of groundwater inverse problems. For this reason, exact enforcement of the discrete governing equations may
Convergence, as well as selecting an optimum solution technique, remain topics for future investigation.

Hence, it is necessary to find an effective algorithm to minimize the penalty method objective function. This is the second motivation of this paper.

The algorithm proposed in this paper can be briefly introduced as follows.

First, different from [2], residuals of the discrete governing equations are scaled by multiplying them with a “scaling matrix” determined adaptively by the current coefficient matrix of the discrete governing equations. With the scaled residuals, a similar penalty method objective function to that in [2] is formed. Detailed numerical analysis shows that the discrete governing equation residuals are not distributed uniformly on the discrete nodes due to the wide value range of the discrete coefficients and their “built-in” but

not be necessarily appropriate, and may actually degrade the head and parameter estimation [2,15]. This is the first motivation of this paper.

Bentley [2] formed a penalty method least-squares objective function by adding a general positive definite quadratic form of the discrete governing equation residuals into the traditional objective function and minimized the objective function without any constraints using a “damped direction-search and linesearch algorithm”. Then he suggested at the end of his paper on p. 147 that

First, different from [2], residuals of the discrete governing equations are scaled by multiplying them with a “scaling matrix” determined adaptively by the current coefficient matrix of the discrete governing equations. With the scaled residuals, a similar penalty method objective function to that in [2] is formed. Detailed numerical analysis shows that the discrete governing equation residuals are not distributed uniformly on the discrete nodes due to the wide value range of the discrete coefficients and their “built-in” but

not be necessarily appropriate, and may actually degrade the head and parameter estimation [2,15]. This is the first motivation of this paper.

Bentley [2] formed a penalty method least-squares objective function by adding a general positive definite quadratic form of the discrete governing equation residuals into the traditional objective function and minimized the objective function without any constraints using a “damped direction-search and linesearch algorithm”. Then he suggested at the end of his paper on p. 147 that
irregular weighting effects on these residuals. This irregularity of the residuals can be efficiently improved by using the above mentioned scaling matrix.

Second, based on the generalized Gauss–Newton method [11,12,21], every iteration minimizing the penalty method objective function is formulated with a linear least-squares system whose unknowns are the corrections of both the unknown parameters and the groundwater heads on all the discrete nodes. The number of linear equations contained in this linear system has to be more than or at least equal to the number of unknowns in it. So the number of observation wells has to be more than or at least equal to the number of unknown parameters when there is no regularization term containing prior information on the unknown parameters in the objective function.

Finally, a suitable algorithm is chosen to solve this linear least-squares system which is usually ill-conditioned. There are several sources of this ill-conditioned state, e.g., the ill-posedness of the inverse aquifer problems themselves, the wide value range of the discrete governing equation coefficients mainly caused by the non-uniformity of the triangulation and the wide range of the transmissivity values, the deterioration of the condition number of the least-squares systems due to large weighting factor $\lambda$ of the penalty term used to enforce the discrete governing equations, etc. Moreover, this linear least-squares system is a large sparse system incorporating the discrete governing equations. In order to overcome these difficulties, an iterative least-squares QR factorization (LSQR) algorithm for sparse linear least-squares problems [17,18] is chosen to solve it. At the same time, a simple but effective preconditioner is implemented to improve the ill-conditioned state.

Numerical results demonstrate that if either the discrete governing equation residuals are not scaled or no preconditioning technique is adopted, then the penalty method algorithm performs poorly due to the ill-conditioned state, e.g., the computation cost is higher than that of the traditional method, and what is worse is, the calculations fail to converge for some initial values of the unknown parameters. But the situation can be improved significantly by some naturally chosen diagonal forms of the scaling matrix and the preconditioner.

Due to the sparse iterative data structure of the LSQR algorithm, the size of the discrete governing equations of our algorithm can be as large as that of the traditional methods.

In every Gauss–Newton iteration of our algorithm, one needs to solve the linear least-squares system only once. In comparison, every Gauss–Newton iteration of the traditional methods has to solve the discrete governing equations as many times as one plus the number of unknown parameters if one uses the parameter perturbation and numerical difference quotient method, or as many times as one plus the number of head observation wells if one uses the state adjoint sensitivity method (variational method) [21]. The total number of iterations required to minimize the penalty method objective function is less than or equal to the number required for the two traditional methods tested.

All these facts demonstrate the potential of this method to solve inverse problems of more complicated non-linear aquifer models naturally and quickly on the basis of finding suitable forms of the scaling matrix and the preconditioner.

The paper is organized as follows. In the next part, the penalty method objective function and its minimizing algorithm based on the Gauss–Newton method are given in detail both for a linear aquifer model and a non-linear model described by the elliptic Boussinesq equation. In Section 3, two numerical examples are given. One of the examples is taken from [2,6]. The other is the elliptic Boussinesq equation describing an unconfined aquifer [1,19]. Through these numerical examples, the effects and utility of both the scaling matrix and the preconditioner are illustrated. At last, the conclusions are drawn on the basis of the former sections and problems that remain to investigate are stated.

2. Algorithm

2.1. Objective function of penalty method

The inverse problems of typical groundwater flow equations in confined and unconfined aquifers will be used as examples to illustrate the algorithm of the least-squares penalty method. Consider two-dimensional steady flow described by

$$-\nabla \cdot (sT\nabla h) - Q + B(h - h_0) = 0, \quad (x, y) \in \Omega$$

(1) in inhomogeneous, isotropic aquifers subject to boundary conditions

$$h = h_d(x, y), \quad (x, y) \in \Gamma_d,$$

(2)

$$sT \frac{\partial h}{\partial n} = q(x, y), \quad (x, y) \in \Gamma_n,$$

(3)

where

$\Omega$: Groundwater flow region. It is a given bounded horizontal domain in $(x, y)$ plane.

$\Gamma_d$ and $\Gamma_n$: The Dirichlet and Neumann boundaries of $\Omega$, respectively. $\Gamma_d \cup \Gamma_n = \partial \Omega$.

$(x, y)$ and $\nabla$: Spatial variables and the gradient operator.

$h(x, y)$: Groundwater head of a confined aquifer or watertable of an unconfined aquifer. Unknown.

Only finite measurement values at given observation wells are available.

$s(x, y, h)$: Specified aquifer-choice function. For confined aquifer $s \equiv 1$. In this case (1)-(3) is a linear
problem. For unconfined aquifer $s = (h - h_b(x, y))/M$ \cite{[1,19]}, where $h_b$ denotes the given elevation of the aquifer bottom and it has the same datum plane as $h$, and $M$ is a positive constant describing the average thickness of the unconfined aquifer. In this, case (1) represents the elliptic Boussinesq equation and (1)–(3) becomes a non-linear problem.

$T(x,y)$: Transmissivity of the aquifer. Unknown. In this paper it is assumed that $T$ is piecewise constant in $\Omega$, i.e., there are $n_p$ given subdomains (zones) $\Omega_i$ of $\Omega (1 \leq i \leq n_p)$, such that

$$\Omega = \bigcup_{i=1}^{n_p} \Omega_i, \Omega_i \cap \Omega_j = \emptyset, \quad i \neq j \quad 1 \leq i, \quad j \leq n_p \quad (4a)$$

and

$$T(x,y) = T_i, \quad (x,y) \in \Omega_i, \quad 1 \leq i \leq n_p, \quad (4b)$$

where $T_i (1 \leq i \leq n_p)$ are unknown positive constants.

$Q(x,y)$: Recharge term determined by factors such as source, sink, precipitation and evapotranspiration etc.

$B(x,y)$ and $h_i$: Specified leakage parameters. We assume $B \leq 0$ for the sake of the well-posedness of (1)–(3).

$\partial/\partial n$: Normal derivative with respect to $\Gamma_n$.

$h_d(x,y)$: Specified head or watertable on $\Gamma_d$.

$q(x,y)$: Specified flux on $\Gamma_n$.

The discrete governing equations of (1)–(3) are the starting point of our method. One can discretize governing equations (1)–(3) using various methods such as finite-difference, finite-volume or finite-element method according to one’s preference. In this paper we use linear triangle finite element method to discretize (1)–(3). Let \{(x_i,y_i), 1 \leq i \leq n\} be the position of the nodes of the triangulation, and $\phi_i(x,y) (1 \leq i \leq n)$ be the linear basis function corresponding to the $i$th discrete node $(x_i,y_i)$. Let

$$H := (h_1, \ldots, h_n)^T \quad (5)$$

be the head or watertable vector, where $h_i$ denotes the finite element approximation to $h(x_i,y_i)$, superscript “$T$” indicates the transpose of a vector or matrix. Then $\hat{h}(x,y)$ and $\hat{s}(x,y,h)$, the respective finite element approximations of $h(x,y)$ and $s(x,y,h)$, can be written as

$$\hat{h}(x,y) = \sum_{i=1}^{n} h_i \phi_i(x,y), \quad (6)$$

$$\hat{s} = \sum_{i=1}^{n} s(x_i,y_i,h_i) \phi_i(x,y). \quad (7)$$

Discretized log-transmissivities ($Y_i := \log T_i, 1 \leq i \leq n_p$), the recharge term $Q$, the leakage parameters $B$ and $h$, are constants on each triangle element $e$, i.e.,

$$Y_{ie} = Y^e, \quad Q_{ie} = Q^e, \quad B_{ie} = B^e, \quad h_{ie} = h^e \quad (8)$$

By applying the standard Galerkin finite element procedure, the discrete governing equations of (1)–(3) can be written in matrix form as follows (see e.g., \cite{[19]}):

$$A(H,P)H = F, \quad (9)$$

where $A$ is an $n \times n$ matrix with elements

$$a_{ij} = s_{ij} + b_{ij}, \quad 1 \leq i, \quad j \leq n \quad (10a)$$

in which

$$s_{ij} = \left\{ \begin{array}{ll}
\sum_{e \in \mathcal{T}_i \cap \mathcal{T}_j} 10^n \int_{\mathcal{E}} \nabla \phi_i \cdot \nabla \phi_j \, dx \, dy, & i \in \mathcal{D}, \\
0 - \sum_{e \in \mathcal{T}_i \cap \mathcal{T}_j} B^e \int_{\mathcal{E}} \phi_i \phi_j \, dx \, dy, & i \notin \mathcal{D},
\end{array} \right. \quad (10b)$$

and

$$b_{ij} = \left\{ \begin{array}{ll}
0, & i \in \mathcal{D}, \\
- \sum_{e \in \mathcal{T}_i \cap \mathcal{T}_j} B^e \int_{\mathcal{E}} \phi_i \phi_j \, dx \, dy, & i \notin \mathcal{D},
\end{array} \right. \quad (10c)$$

where $\delta_{ij}$ equals 1 when $i = j$ and 0 when $i \neq j$, $w_d$ is a positive weighting constant that is chosen large enough compared with all the other diagonal entries of $A$ for the discrete Dirichlet boundary conditions to be enforced at least as accurately as other discrete governing equations.

$e$ is the triangle element, $\mathcal{T}_i$ is the set of all the triangle elements in the neighborhood of the $i$th node $(x_i,y_i) (1 \leq i \leq n)$, i.e., $\mathcal{T}_i = \{ e(x_i,y_i) \text{ is a vertex of } e \}$. $\mathcal{D} := \{ i : (x_i,y_i) \in \Gamma_d \}$ is the serial number set of all the discrete nodes on $\Gamma_d$. And as usual, $P$ is the “log-transmissivity” vector, i.e.,

$$P := (Y_1, \ldots, Y_{n_p})^T = (\log T_1, \ldots, \log T_{n_p})^T \quad (11)$$

$F$ is an $n$-dimensional vector determined by forcing terms such as $Q$, $B$, $h$, and the boundary conditions of (1)–(3), whose components can be written as

$$f_i = \left\{ \begin{array}{ll}
\sum_{e \in \mathcal{T}_i} (Q^e + B^e h^e) \int_{\mathcal{E}} \phi_i \, dx \, dy + \int_{\Gamma_n} q \, \phi_i \, d\Gamma_n, & i \in \mathcal{D}, \\
\sum_{e \in \mathcal{T}_i} (Q^e + B^e h^e) \int_{\mathcal{E}} \phi_i \, dx \, dy, & i \notin \mathcal{D}.
\end{array} \right. \quad (12)$$

If the aquifer is confined ($s = 1$), then either the conjugate gradient method or the Cholesky decomposition method is used to solve the discrete aquifer model (9), otherwise $s = (h - h_b)/M$, (9) is a non-linear problem and it is solved by iterative method.

As in \cite{[2]}, the observation head residual vector $R_m$ and parameter prior information residual vector $R_p$ can be defined by

$$R_m(H) = H_s - H_m, \quad (13)$$

$$R_p(P) = P - P^*, \quad (14)$$

where $H_s$ and $H_m$ are $n_m$-dimensional vectors whose respective components denote the simulated and measured head values at the observation wells; $n_q$ is the number of observation wells; $P^*$ denotes an a priori estimate of $P$.

In a departure from \cite{[2]}, we define a generalized residual vector of the discrete governing equations as
$R_g(H, P) = U(A(H, P)H - F)$, (15)

where $U$ is an $n \times n$ given non-singular “scaling” matrix used to scale the elements of $R_g$. Later we will discuss how to choose it. Then the traditional least-squares method for aquifer parameter identification can be written as [4–6,8]

\[
\begin{align*}
\min_{J_p} J_p(H, P) & := J_m + J_p \\
\text{subject to} & \quad U^{-1}R_g(H, P) = 0,
\end{align*}
\]

(16)

where

\[
J_m = R_m^T W_m R_m,
\]

(17)

\[
J_p = \lambda_p R_p^T W_p R_p
\]

(18)

and $W_m$ is a given $n_m \times n_m$ symmetric positive definite matrix determined by the measurement errors of the head, $W_p$ is a given $n_p \times n_p$ symmetric semi-positive definite matrix determined by the errors of the prior information on $P$, $\lambda_p$ is a non-negative weighting factor used to measure the exactness and reliability of the prior information on the unknown parameters compared to those of the information on the models and head measurements.

The least-squares penalty method can be written as

\[
\min_{J_p} J_m + J_p + J_g = J_g + J_g,
\]

(19)

where

\[
J_g = \lambda_g R_g^T W_g R_g
\]

(20)

is the penalty term to enforce the discrete governing equations, $W_g$ is a symmetric positive definite weighting matrix, and $\lambda_g$ is a weighting factor to control the enforcement of the discrete governing equations. In this paper we will set $W_g$ to be $I_n$, i.e., the $n$th-order identity matrix.

The least-squares penalty method objective function in [2] is the special case of (19) when $U = I_n$. Such a choice of $U$ will cause non-uniformity of the components of $R_g$ because the matrix $A$ in (15) has irregular weighting effects due to the great value range in the elements of $A$.

In order to analyse the weighting effects of $A$ when $U$ and $W_g$ equal $I_n$, we define $0 \leq \sigma_1 \leq \sigma_2 \leq \cdots \leq \sigma_n$ as the eigenvalues of the matrix $A^T A$ and $V_i$ ($1 \leq i \leq n$) as the corresponding unit orthogonal eigenvectors. Then

\[
V := (V_1, \ldots, V_n)
\]

(21)

is an orthogonal matrix. Define $R_h := H - A^{-1}F$ as the computational residual vector between the heads of least-squares method and those of direct method. Then from (15), (20) and (21) we have

\[
J_g|_{U=I_n} = \lambda_g^2 (V^T R_h)^T \text{Diag} [\sigma_1, \ldots, \sigma_n] V^T R_h.
\]

(22)

Eq. (22) implies that the weighting factor of the $i$th component of $V^T R_h$ is $\lambda_g^2 \sigma_i$, $1 \leq i \leq n$. Due to the non-uniformity of the spatial discretization and the large range of the transmissivity $T$ values, the value range in the elements of $A$ is great. This will result in large condition number of $A^T A$, namely the ratio $\sigma_n / \sigma_1$ [20]. For instance, in the simple synthetic examples in [2], our calculations show that the largest and smallest diagonal elements of $A^T A$ are respectively $4.5 \times 10^5$ and 37.5 with the latter corresponding to the node at the “northeast corner” of the domain in the example (Fig. 1 in [2] or this paper). It is well known that $\sigma_1$ is less than or equal to the smallest diagonal element of $A^T A$ and $\sigma_n$ greater than or equal to its greatest diagonal element. Hence the ratio $\sigma_n / \sigma_1 \geq 4.5 \times 10^5 / 37.5 = 1.2 \times 10^4$. This explains the interesting phenomenon in the numerical examples in [2] that the node with the largest residual is always at the northeast corner of the square domain, because its weighting factor corresponds to the smallest eigenvalue of $A^T A$.

Hence we reach the following conclusion: only by minimizing (19) with a scaling matrix $U$ that can appropriately rectify the non-uniform or irregular weighting effects of $A^T A$ on the components of the residual vector $R_g$, one can get a scaled “stable” residual vector $R_g$ independent of the wide value range of the entries of $A$ and make it possible to exploit the deeper potential information contained in $R_g$. Moreover, the scaling matrix will improve the illcondition caused by the wide value range of the entries of $A$.

Therefore, a suitable scaling matrix $U$ is necessary, especially for realistic problems with a wide range of entries in $A$. In this paper, three simple forms of the scaling matrix $U$ are given. Their effects will be shown and compared in the next section. These forms are:

(a) the identity matrix, i.e.,

\[
U = I_n,
\]

(23)

where there is no “scaling” or improvement as in [2];

(b) the infinite row norm form

\[
U = U_{\infty} := \text{Diag} [\tau_1^{-1}, \ldots, \tau_n^{-1}],
\]

(24)

where $\text{Diag} [\ldots]$ indicates a diagonal matrix, and $\tau_i$ is the $i$th diagonal element of $A$;

(c) the Euclidean row norm (2-norm) form

\[
U = U_E := \text{Diag} [\tau_1^{-1}, \ldots, \tau_n^{-1}],
\]

(25)

where $\tau_i$ is the Euclidean norm of the $i$th row of $A$, i.e.,

\[
\tau_i = \left( \sum_{j=1}^n a_{ij}^2 \right)^{1/2}.
\]

(26)

Although the above forms of $U$ given by (24) and (25) are dependent on $H$ and $P$, they will be regarded as constant during each iteration of the Gauss–Newton method when minimize (19) in the next section. The reason to do so is that we are interested in improving or
scaling the irregular weighting effects of \( A \) with the simplest calculation without changing the original purpose of the least-squares penalty method. This means that the penalty term \( J \) is corrected adaptively at every iteration according to the change of the entries of \( A \) mainly caused by changes in the parameter vector \( P \).

### 2.2. Algorithm

Let

\[ \chi = (\chi_1, \ldots, \chi_n, \ldots, \chi_{n+n_p})^T := (H^T, P^T)^T. \]  

(27)

Then the objective function \( J_{\text{mp}} \) defined by (19) is a non-linear function of the variable vector \( \chi \). The minimization must be carried out iteratively. Let \( i_i \) be the iteration index and \( \chi_i^0 \) be a given initial estimate of \( \chi \). At every iteration we wish to determine a correction \( \Delta \chi \) of the current iterative value of \( \chi \), i.e. \( \chi_i^k \), such that

\[ \chi_i^{k+1} = \chi_i^k + \Delta \chi, \quad i_i = 0, 1, 2, \ldots \]  

(28)

results in a reduction of the objective function \( J_{\text{mp}} \). The Gauss–Newton method for obtaining \( \Delta \chi \) leads to the following linear system of equations \([11,12,21]\)

\[ J^T W J \Delta \chi = -J^T W R, \]  

(29)

where

\[ W = \text{Diag}[\chi_1^2 W_1, \ldots, \chi_n^2 W_m, \chi_{n+n_p}^2 W_p] \]  

(30)

and \( R \) is the entire residual vector composed of \( R_\varepsilon, R_m \) and \( R_p \), i.e.,

\[ R = (r_1, \ldots, r_n + n_m + n_p)^T := (R_\varepsilon^T, R_m^T, R_p^T)^T, \]  

(31)

\( J \) is the Jacobian or sensitivity matrix when \( \chi = \chi_i^k \) with elements

\[ J_{ij} = \frac{\partial r_i}{\partial \chi_j} \, |_{\chi = \chi_i^k}, \quad 1 \leq i \leq n + n_m + n_p, \quad 1 \leq j \leq n + n_p. \]  

(32)

This \((n + n_m + n_p) \times (n + n_p)\) matrix \( J \) is very large because the node number \( n \) is usually very large. Hence it will be time-consuming and require substantial computer-storage to solve (29) directly. Note that (29) is equivalent to the following linear least-squares system

\[ \min_{\Delta \chi} \| W J \Delta \chi + W R \|_2 \]  

(33)

where \( \| \cdot \|_2 \) indicates the L2-norm of a vector, \( W \) is the square root or Cholesky decomposition of the positive definite matrix \( W \), i.e.,

\[ W^T W = W. \]  

(34)

Instead of solving (29) directly, we consider the linear least-squares system (33) which can be solved using the iterative LSQR method \([17,18]\) or the direct QR factorization method \([10,12]\). In our numerical tests both methods are tried. For relatively small \( n \) (e.g., \( \leq 100 \)) both work well. When \( n \) is larger, the direct QR factorization method needs more storage and its speed is comparable to the iterative method. Hence the iterative LSQR method is preferred, because it can use a sparse matrix data structure to store the non-zero elements of \( J \). Consequently, the node number \( n \) can be as large as that of the traditional method.

In order to solve (33) efficiently and accurately, a suitable preconditioning technique is needed. Let \( C \) be a suitable \((n + n_p) \times (n + n_p)\) non-singular matrix which can improve the condition number of the matrix \( W J C \). Then instead of solving (33) directly, we first solve the following system about an \((n + n_p)\)-dimensional unknown vector \( \xi \)

\[ \min_{\xi} \| W J C \xi + W R \|_2 \]  

(35)

and then calculate

\[ \Delta \chi = C \xi. \]  

(36)

The simplest form of \( C \) is a diagonal matrix. We wish to improve the condition number of \( W J C \) by choosing a diagonal matrix \( C := D_k \) such that all the diagonal elements of \( D_k J^T W J D_k \) is equal to 1. This leads to the following form of \( C \)

\[ C := D_k = \text{Diag}[\rho_1, \ldots, \rho_{n+n_p}], \]  

(37)

where

\[ \rho_j = \left( \sum_{i=1}^{n+m+n_p} J_{ij} w_{i,k} \right)^{-1/2}, \quad 1 \leq j \leq n + n_p \]  

(38)

with \( w_{i,k} \) denoting the \((i,k)\) entry of \( W \). In our numerical tests this simple procedure accelerates the convergence of the iterative LSQR algorithm significantly and it also widens the range of initial values of the unknown parameter \( P \) within which our algorithm can converge (see Section 4, Table 1).

The main work in implementing the above algorithm is to calculate \( J \). By calculating directly using (4a), (4b), (5)–(9), (10a)–(10c), (11)–(15), (27), (31) and (32), setting the matrix \( U \) according to the entries in \( A \) of the current iteration and regarding \( U \) as a constant matrix independent of \( \chi \), \( J \) can be written as

\[ J = \left( \begin{array}{cc} U(A + Z) & UP \\ M_w & 0 \\ 0 & I_{n_p} \end{array} \right) \]  

(39)

in which \( Z \) is an \( n \times n \) matrix with elements

\[ z_{ij} = \frac{\partial (AH - F)}{\partial h_j} - a_{ij} = \sum_{k \in \mathcal{V}_i} \frac{\partial s_k}{\partial h_j} h_k \]

\[ = \begin{cases} 0 & \text{if } i \in \mathcal{D} \text{ or } j \notin \mathcal{N}_i \\ \sum_{k \in \mathcal{N}_i} h_k \sum_{e \in \mathcal{F}} \int_{\mathcal{F}} 10^{iy} \phi_j \nabla \phi_i \cdot \nabla \phi_k \, dx \, dy \end{cases} \]  

(40)
where \( s' = (ds/\partial h)(x_i, y_j, h_j) \), and \( \mathcal{V}_i \) is the serial number set of the nodes in the neighborhood of the \( i \)th node \((x_i, y_i))\). For linear aquifer models \( Z = 0 \) because of \( s' = 1 \). \( M_{n,m} \) is an \( n_m \times n \) matrix whose \( i \)th row is an \( n \)-dimensional vector with its \( n \)th component being \( 1 \) and the others \( 0 \). Here \( n_j(1 \leq i \leq n_m) \) indicates the serial number of the node located at the \( i \)th head observation well. \( I_{n} \) is the \( n \times n_p \) matrix with elements

\[
p_{ij} = \sum_{k \in \mathcal{V}_i} \frac{\partial Q_k}{\partial Y_j} h_k = \begin{cases} 
0 & \text{if } i \in \mathcal{D} \sum_{k \in \mathcal{V}_i} h_k \sum_{\mathcal{F} \cap \mathcal{F}_j} \int_{\mathcal{F}_j} 10^Y e \partial \phi \partial f \partial \partial dx dy dx, \\
\ln 10 \sum_{k \in \mathcal{V}_i} h_k \sum_{\mathcal{F} \cap \mathcal{F}_j} \int_{\mathcal{F}_j} 10^Y e \partial \phi \partial f \partial \partial dx dy dx, 
\end{cases}
\]

(41)

and \( \Omega_j \) is defined by (4a) and (4b). Here it is assumed that each triangle element \( e \) is contained by one of the subdomains \( \Omega_i, \ldots, \Omega_m \).

Based on the above discussion, the penalty method algorithm can be described as follows:

1. **Initialization.** Choose a suitable \( P_0 \) as the initial value of \( P \). The initial value \( H_0 \) of \( H \) is obtained by solving (9) when \( P = P_0 \). Set \( \gamma = (H_0^T P_0^T)^T \), \( J_0 = 0 \). Choose two suitably small numbers \( \delta_1 > 0 \) and \( \delta_2 > 0 \) as the convergence criteria of the Gauss–Newton iteration. Choose two suitable \( n \times n \)-dimensional vectors \( P^{-} \) and \( P^{+} \) as the lower and upper bounds of the parameter vector \( P \).

2. For \( n_k = 0, 1, 2, \ldots \) repeat steps 3–5.

3. **Implement Gauss–Newton method.**

(a) **Calculate Jacobian matrix** \( J \). Calculate \( J_{ij} = (\partial y_j / \partial x_i) \) (1 \( \leq i \leq n + n_m + n_j, 1 \leq j \leq n + n_p \)) using (39)–(42), while the matrix \( U \) in (39) is determined by the current entries in \( A \). (In numerical examples three forms of \( U \) defined by (23)–(26) are used and their utilities are compared.)
Verify the convergence.

(c) Update $\chi$. Calculate $\hat{\chi}^{n+1}$ using (28).

4. Verify the convergence. Calculate the value of $J_{it+1} := J_{gmp}$ using (19). If
   \begin{equation}
   |J_{it+1} - J_{it}| < \delta_j \quad \text{or} \quad |\Delta \chi|_{\infty} < \delta_x \tag{43}
   \end{equation}
   holds, stop, where $|\Delta \chi|_{\infty}$ indicates the infinity norm of $\Delta \chi$, else set $i_t := i_t + 1$.

5. Implement the upper and lower bound constraints to $P$.
   Let $P_{n+1}$ be the last corresponding $n_p$ components of $\hat{\chi}^{n+1}$. Judge if $P^{-} \leq P_{n+1} \leq P^{+}$ holds, where the inequality refers to the corresponding components of the three vectors. If there are components of $P_{n+1}$ which are less than their lower bounds or greater than their upper bounds, then set them to equal their lower or upper bound values, respectively. Then set the last corresponding $n_p$ components of $\hat{\chi}^{n+1}$ to equal those of $P_{n+1}$ and go to step 3.

3. Numerical examples

3.1. Example of linear aquifer model

In order to demonstrate the utility of our algorithm, the steady-state test problem used by Carrera and Neuman in [6] and Bentley in [2] will be solved. It can be described by specifying the factors in the aquifer model (1)–(3) as follows (see [2,6]):

$\Omega$. A square domain of $6000 \text{ m} \times 6000 \text{ m}$ (see Fig. 1).
$I_d$ and $h_d$. $I_d$ is the southern side of the square domain, $h_d \equiv 100 \text{ m}$.
$I_n$ and $q$. The three sides of the domain are $I_n$.

On the western side the flux $q = 0.25 \text{ m}^2/\text{day}$, on the northern and eastern sides there is no flow, i.e., $q = 0$.

Transmissivity $T(x, y)$. The domain $\Omega$ is divided into $n_p = 9$ constant-transmissivity zones of $2000 \text{ m} \times 2000 \text{ m}$ with the corresponding transmissivity values $T_i (1 \leq i \leq 9)$ ranging from 5 to 150 $\text{m}^2/\text{day}$ (see Fig. 1).

$s(x, y, h), B(x, y)$ and $h_r$. The aquifer is assumed to be confined, i.e., $s(x, y, h) \equiv 1$. The stratum overlying and underlying the aquifer are supposed to be impervious, i.e., $B(x, y) \equiv 0$. Hence $Bh_r \equiv 0$.

$Q(x, y)$. In the upper half part of the northern three zones $Q = 0.274 \times 10^{-3} \text{ m/day}$, in the lower part of them $Q = 0.137 \times 10^{-3} \text{ m/day}$, in other six zones $Q = 0$.

Head measurements. They are available at the 18 observation wells shown as points in Fig. 1.

In computation, the flow region is discretized into $12 \times 12$ squares, each of which is divided into two triangle elements. All the head observation wells coincide with the discrete nodes. In order to obtain head measurements at observation wells, as in [2,6], the “true” head values at discrete nodes, which range from 100.0 m on the southern Dirichlet boundary to 205.88 m at the north-east corner, were numerically calculated using the true transmissivity values shown in Fig. 1. Then head values at the 18 observation wells were corrupted by uncorrelated Gaussian noise of variance 1. Consequently the head measurement weighting matrix $W_m$ in (17) is set to be the $n_m$th-order identity matrix as in [2,6].

To test the utility of our algorithm, the regulation term $J_{p}$ containing prior information on $P$ is not used in all our calculation, i.e., $\lambda_p \equiv 0$. But $P$ is restricted from below and above by choosing $P^{-}$ with uniform components log $0.01 = -2$ and $P^{+}$ with uniform components log $10^3 = 3$. The Gauss–Newton iteration stop criteria are chosen to be $\delta_1 = 10^{-7}$ and $\delta_2 = 10^{-4}$.

The numerical tests are composed of two parts.

The first part is designed to investigate the effects of different forms of the residual scaling matrix $U$ and the preconditioner $C$ on the calculation speed and numerical results. For this purpose, three different forms of $U$, i.e., $I_n$, $U_\infty$ and $U_\|C$ defined by (23)–(26), two different forms of $C$, i.e., $I_{n+n_p}$ and $D_{C2}$ defined by (37) and (38), and four different but uniform initial values of $T_i (1 \leq i \leq n_p = 9)$ as in [6], namely, 1, 10, 100 and $200 \text{ m}^2/\text{day}$, are used combinatorially. The weighting factor $\lambda_g$ is respectively fixed to be 1.0 when $U = I_n$ considering the great equivalent weighting effects of $A^T A$ and to be 10.0 for the other two forms of $U$ defined by (24)–(26) because of their counteracting the weighting effects of $A^T A$. Hence there are altogether $4 \times 2 \times 3 = 24$ different cases.
In Table 1 the relevant data on the computational costs corresponding to the 24 cases are summarized. The first eight rows corresponding to methods 1–4 show that if either the discrete governing equation residuals are not scaled (i.e., \( U = I_n \)), or no precondition technique is adopted (i.e., \( C = I_{n+n} \)), our algorithm performs poorly, e.g., the computation cost is higher than that of the traditional method, and what is worse is the Gauss–Newton iterations fail to converge for some initial values of the unknown parameters. But the situation is always improved with the gradual improvement of \( U \) and \( C \).

The successive four rows corresponding to methods 5 and 6 demonstrate the improvements due to setting \( U = U_\infty \) or \( U_E \) (Eqs. (24)–(26)) and \( C = D_3 \) (Eqs. (37) and (38)). The range of initial values of \( T \) for which the Gauss–Newton iteration converges has increased, there are no longer “Failed” calculations for all the four different initial values of \( T \); and the computational effort is also significantly reduced. For example, when the initial values of \( U \) are uniform log 100 m/day, \( S_{\text{sp}r} \), the number of LSQR iterations is reduced from 16360 for \( U = I_n \) and \( C = I_{n+n} \) to 1980 for \( U = U_\infty \) and \( C = D_3 \). And the CPU time is reduced proportionally.

In the next two rows that begin with “B & Y” we show the computational effort of the method described by Becker and Yeh [3,21], which uses a Gauss–Newton method to minimize the objective function \( J \) defined by (16) means of by the parameter perturbation and numerical difference quotient method. Comparison of the CPU times and the number of iterations required by the “B & Y” method (i.e., BY) to the results from the improved scaling matrixes demonstrates that the computational cost of our algorithm is reduced to a level similar to that of the traditional method. This trend shows potential of our algorithm to solve inverse problems more quickly than the traditional method on the basis of finding more suitable forms of \( U \) and \( C \).

In the last row, \( CN \), the number of iterations required by the “conjugate gradient algorithm coupled with Newton’s method” used by Carrera and Neuman in [6] to minimize the traditional objective function \( J_1 \) defined by (16), is quoted. One can see that \( GN \), the number of Gauss–Newton iterations required by our algorithm, is less than \( CN \). Unfortunately, the computational costs of our algorithm and that used in [6] in every iteration minimizing the objective functions are difficult to compare due to the lack of relevant data.

The five Failed cases in Table 1, which mean that the iterations fail to converge, can be improved so that the iterations converge if the Marquardt’s method [7,12] is incorporated in our algorithm. But it takes more iterations and CPU times than those of the other unfailed cases listed in Table 1 where the Marquardt’s method is not used. The detail will not be stated here due to space limitation.

In Table 2 the numerical results corresponding to the 24 cases mentioned above and to the traditional method by Becker and Yeh [3,21] are summarized. Theoretically, (i.e., if there are no round off errors) different forms of non-singular matrix \( C \) do not influence the calculated results. The numerical results support this fact: the optimization results neither depend on the initial values of \( T \), nor on the change of \( C \) up to five significant figures. Hence, only results corresponding to the three different forms of \( U \) are listed in Table 2, and they are valid for all the combinations of different initial values of \( T \) and forms of \( C \) in Table 1 which are not Failed. The last column is the results calculated by using the traditional method of Becker and Yeh.

The first four rows in Table 2 give the values of \( J_1 \), the traditional objective function that appeared in (19), of \( J_g \), the weighted square sum of the discrete governing equation residuals defined by (20), of \( J_{gm} \), the penalty method objective function defined in (19) and of the ratio \( J_g/J_{gm} \) successively. It is well-known that greater the weighting factors of residuals in a least-squares system, smaller these residuals will be. Hence the ratio \( J_g/J_{gm} \) can be regarded as a measure of the weighting effects of \( \lambda_g U \) on the discrete governing equation residuals. According to this measure the weighting effects of \( \lambda_g U = 10.0 U_E \) are the weakest, and those of \( \lambda_g U = 1.0 I_n \) the strongest.

The data listed under “Identified \( T \)” and “The Statistic Data of log \( T \)” demonstrate that the numerical results of \( T \) differ very slightly when they are calculated by using our algorithm with the three different choices of \( \lambda_g U \) and when they are obtained by the traditional method of Becker and Yeh. For example, \( \sigma(\Delta P) \), the standard deviation of the residuals between the computed and true log-transmissivity, ranges only from 2.78 \times 10^{-3} \text{ to } 3.33 \times 10^{-3} \text{, and } \text{SSE}(\Delta P), \text{ the standard square error between the computed and true log-transmissivity, from } 2.81 \times 10^{-3} \text{ to } 3.54 \times 10^{-3} \text{.}

One may conjecture that the discrete governing equations will be less accurately satisfied at the nodes coincident with the head observation wells than at other nodes due to the measurement error at the head observation wells. This conjecture can be seen and understood clearly if “The statistic data of \( R_{gm} \) and \( R_{gr} \)” are compared row by row, where \( R_{gm} \) is an \( n_{m} \)-dimensional vector composed of the components of \( R_g \) whose indices equal those of the nodes coincident with the head observation wells and \( R_{gr} \) is an \( (n-n_{m}) \)-dimensional vector composed of the rest of the components of \( R_g \) except those of \( R_{gm} \). For example, \( \text{SSE}(R_{gm}) \), the standard square error of \( R_{gm} \), is much greater than \( \text{SSE}(R_{gr}) \) for all the three choices of \( \lambda_g U \). The ratio \( \text{SSE}(R_{gm})/\text{SSE}(R_{gr}) \) ranges from 2.16 to 9.69. The other pair of statistics \( |R_{gm}|_\infty \text{ and } |R_{gr}|_\infty \), the respective infinity norms of \( R_{gm} \) and \( R_{gr} \), also show the similar phenomenon with the ratio \( |R_{gm}|_\infty/|R_{gr}|_\infty \) ranging from 1.96 to 3.06. That
Comparison of the numerical results of our algorithm and of the traditional method

<table>
<thead>
<tr>
<th>$U$</th>
<th>$I_n$</th>
<th>$U_\infty$</th>
<th>$U_E$</th>
<th>Traditional method</th>
</tr>
</thead>
<tbody>
<tr>
<td>$J_1$</td>
<td>16.0</td>
<td>14.8</td>
<td>14.5</td>
<td>16.2</td>
</tr>
<tr>
<td>$J_2$</td>
<td>0.115</td>
<td>0.674</td>
<td>0.839</td>
<td>0.0</td>
</tr>
<tr>
<td>$J_{\text{gm}}$</td>
<td>16.1</td>
<td>15.5</td>
<td>15.3</td>
<td></td>
</tr>
<tr>
<td>$J_{\text{gm}} / \hat{\gamma}$</td>
<td>0.715%</td>
<td>4.35%</td>
<td>5.47%</td>
<td>0.0%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>True $T$</th>
<th>Identified $T$ (m$^3$/day)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1(150)$</td>
<td>158.71</td>
</tr>
<tr>
<td>$T_1(150)$</td>
<td>137.20</td>
</tr>
<tr>
<td>$T_1(50)$</td>
<td>51.608</td>
</tr>
<tr>
<td>$T_1(150)$</td>
<td>123.02</td>
</tr>
<tr>
<td>$T_1(50)$</td>
<td>49.756</td>
</tr>
<tr>
<td>$T_1(15)$</td>
<td>15.516</td>
</tr>
<tr>
<td>$T_1(50)$</td>
<td>66.537</td>
</tr>
<tr>
<td>$T_1(15)$</td>
<td>15.028</td>
</tr>
<tr>
<td>$T_1(5)$</td>
<td>5.0254</td>
</tr>
</tbody>
</table>

The statistic data of $log T$

<table>
<thead>
<tr>
<th>$E(AP)$</th>
<th>$\sigma(AP)$</th>
<th>$\text{SSE}(AP)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5.89 \times 10^{-3}$</td>
<td>$3.27 \times 10^{-3}$</td>
<td>$2.81 \times 10^{-3}$</td>
</tr>
<tr>
<td>$4.90 \times 10^{-3}$</td>
<td>$3.25 \times 10^{-3}$</td>
<td>$2.81 \times 10^{-3}$</td>
</tr>
<tr>
<td>$4.24 \times 10^{-3}$</td>
<td>$3.33 \times 10^{-3}$</td>
<td>$3.34 \times 10^{-3}$</td>
</tr>
<tr>
<td>$7.05 \times 10^{-3}$</td>
<td>$2.96 \times 10^{-3}$</td>
<td>$3.01 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

The statistic data of $R_{gm}$ and $R_{gt}$

<table>
<thead>
<tr>
<th>$\hat{\gamma}<em>{SSE}(R</em>{gm})$</th>
<th>$\hat{\gamma}<em>{SSE}(R</em>{gt})$</th>
<th>$\hat{\gamma}<em>{SSE}(R</em>{gm}) / \hat{\gamma}<em>{SSE}(R</em>{gt})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1.31 \times 10^{-3}$</td>
<td>$2.01 \times 10^{-2}$</td>
<td>$2.50 \times 10^{-2}$</td>
</tr>
<tr>
<td>$2.08 \times 10^{-3}$</td>
<td>$2.58 \times 10^{-3}$</td>
<td>$2.58 \times 10^{-3}$</td>
</tr>
<tr>
<td>$4.50 \times 10^{-3}$</td>
<td>$9.66$</td>
<td>$9.69$</td>
</tr>
<tr>
<td>$6.50 \times 10^{-4}$</td>
<td>$9.09$</td>
<td>$9.09$</td>
</tr>
<tr>
<td>$0.0458$</td>
<td>$0.0190$</td>
<td>$0.0190$</td>
</tr>
<tr>
<td>$0.0458$</td>
<td>$0.0190$</td>
<td>$0.0190$</td>
</tr>
<tr>
<td>$3.06$</td>
<td>$2.33$</td>
<td>$1.96$</td>
</tr>
</tbody>
</table>

Table 2

is to say, statistically, the residuals of the discrete governing equations discretized at the head-observation-well nodes are much greater than those of the discrete governing equations at other nodes. Therefore, the above-mentioned conjecture is true in the sense of statistics.

If “The statistic data of $R_{gm}$ and $R_{gt}$” are compared column by column, the following interesting facts can be found: among the three group values listed in the three columns, the values of $\hat{\gamma}_{SSE}(R_{gm})$ and $\hat{\gamma}_{SSE}(R_{gt})$ in the first column corresponding to $\hat{\gamma}_{SSE}(U = 1.0I_n)$ are always the smallest, but the values of $|R_{gm}|_{\infty}$ and $|R_{gt}|_{\infty}$ in the first column are always the biggest. These results imply that the components of the residual vector $R_{g}$ when $U = I_n$ are irregularly distributed and they have been improved when $U$ is set to be $U_\infty$ or $U_E$.

The second part of our numerical tests are designed to investigate the effects of the weighting factor $\hat{\gamma}_{SSE}$ on the numerical results. To this end the initial values of $T$ are fixed to be uniform 100 m$^3$/day, $U = U_E$ and $C = D_{c2}$, then our algorithm is used to minimize the penalty method objective function $J_{gm}$ corresponding to different $\hat{\gamma}_{SSE}$, which is respectively set to be 10.0, 10$^2$, 10$^4$, 10$^6$ and 10$^8$. The results are listed in Table 3.

First let us look at the data listed above “The computation cost” in Table 3. If they are compared column by column, the following facts can be seen:

1. As $\hat{\gamma}_{SSE}$ increases from 10.0 to 10$^6$, the values of $J_1$ and $J_{gm}$ tend to the same constant 16.2 increasingly, and the values of $J_6$ decreases with an approximate rate of $\hat{\gamma}_{SSE}^{-2}$. So the ratio $J_6 / J_{gm}$ decreases also with the same approximate rate $\hat{\gamma}_{SSE}^{-2}$. In fact, it has been theoretically proven that $J_6$, $J_{gm}$ and $J_6$ have the asymptotic properties

$$\lim_{\hat{\gamma}_{SSE} \to +\infty} J_1 = \lim_{\hat{\gamma}_{SSE} \to +\infty} J_{gm} = C_1, \quad \lim_{\hat{\gamma}_{SSE} \to +\infty} \hat{\gamma}_{SSE} J_6 = C_2,$$

where $C_1$ is the value of $J_1$ corresponding to the traditional-method solution to problem (15), $C_2$ is a nonnegative constant that vanishes if and only if $C_1$ vanishes [13]. The numerical results in Table 3 show these properties clearly when $\hat{\gamma}_{SSE} \leq 10^6$ with the constants $C_1$ and $C_2$ approximating 16.2 and 94.1, respectively. Unfortunately, by $\hat{\gamma}_{SSE} = 10^8$, they begin to
3. The facts stated in the above paragraph are also sup-
pose part of their fidelity due to the serious deterior-
ity of the condition number of the weighting matrix \( W \) defined by (30).

2. When \( \lambda_g^2 = 10.0 \), \( J_g \) takes comparatively large value 4.06 and the ratio \( J_g/J_{mp} \) reaches 36.9\%, implying that much of the head measurement noise is “absorbed” by the discrete governing equation residuals. This leads to the large errors in the estimated values of \( T_i(1 \leq i \leq 9) \) (see the second column listed under \( \lambda_g^2 = 10.0 \)). When \( \lambda_g^2 \geq 10^2 \), accurate and stable estimates of \( T_i(1 \leq i \leq 9) \) are obtained due to the sufficient enforcement of the discrete governing equations, even in the case of \( \lambda_g^2 = 10^8 \). They also differ very slightly from each other. Moreover, if one compares our results of \( T_i(1 \leq i \leq 9) \) when \( 10^4 \leq \lambda_g^2 \leq 10^8 \) with the traditional method results of \( T_i \) listed in the last column of Table 2, one can easily see that each \( T_i(1 \leq i \leq 9) \) always has the same first three significant figures with only one exception of \( T_7 = 67.5 \) when \( \lambda_g^2 = 10^8 \).

3. The facts stated in the above paragraph are also sup-
ported by the data listed under “The statistic data of log \( T \)”.

In the last two rows the “CPU Time” and \( S_{sup} \), the sum of LSQR iterative numbers in all the Gauss–New-
ton iterations, are listed for different values of \( \lambda_g^2 \). One may see that although they differ from each other, the average CPU time is only 18.9 s, which is a little less than 19.5 s, the CPU time of the traditional method corresponding to the method 7 B & Y and initial \( T = 100 \text{ m}^2/\text{day} \) in Table 1. Particularly, when \( \lambda_g^2 = 10^4 \), the CPU time of our algorithm is only 11.1 s, 56.9\% of that of the traditional method.

In short, there is a sufficiently wide workable range of \( \lambda_g \) values within which the penalty method algorithm performs well enough to give accurate estimations of the unknown parameter \( T \) with low computational cost just like the traditional method.

### 3.2. Example of non-linear aquifer model described by Boussinesq equation

In order to demonstrate the advantage of our algo-

rithm that it can solve inverse problem of non-linear

aquifer models naturally and quickly, an unconfined

quasilinear aquifer model described by the Boussinesq

equation [1,19] is artificially designed by specifying the

factors in the aquifer model (1)–(3) as follows:

1. The aquifer-choice function \( s(x, y, h) = (1/M) \)

\((h - h_b(x, y))\), where \( h_b := (-1/400)(x + y) \), \( M := 160.0 \text{ m} \).

2. The leakage parameters \( B(x, y) := -5.0 \times 10^{-7} \)

\text{day}^{-1}, \( h_l = 0 \text{ m} \).

3. All the other factors, such as \( Q, r \) and \( h_0, r_a \) and \( q_s \),

transmissivity \( T(x, y) \) determined by the average

thickness of the unconfined aquifer, recharge \( Q(x, y) \)

and watertable measurement locations are the same

as those described in the example of Section 3.1 and

Fig. 1.

As before, the flow region \( \Omega \) is discretized into

12 \times 12 \text{ squares each of which is divided into two

Table 3
Comparison of the effects of different weighting factor \( \lambda_g \) on the numerical results *

<table>
<thead>
<tr>
<th>( \lambda_g^2 )</th>
<th>10.0</th>
<th>10^2</th>
<th>10^4</th>
<th>10^6</th>
<th>10^8</th>
</tr>
</thead>
<tbody>
<tr>
<td>( J_i )</td>
<td>6.95</td>
<td>14.5</td>
<td>16.2</td>
<td>16.2</td>
<td>16.2</td>
</tr>
<tr>
<td>( J_p )</td>
<td>4.06</td>
<td>8.38 \times 10^{-3}</td>
<td>9.39 \times 10^{-3}</td>
<td>9.41 \times 10^{-3}</td>
<td>6.64 \times 10^{-1}</td>
</tr>
<tr>
<td>( J_g/J_{mp} )</td>
<td>11.01</td>
<td>15.3</td>
<td>16.2</td>
<td>16.2</td>
<td>16.9</td>
</tr>
<tr>
<td>( J_g/J_{mp} )</td>
<td>36.9%</td>
<td>5.47%</td>
<td>0.058%</td>
<td>0.000581%</td>
<td>3.94%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>True ( T )</th>
<th>Identified ( T ) (m\text{/day})</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T_1(150) )</td>
<td>130.14</td>
</tr>
<tr>
<td>( T_2(150) )</td>
<td>104.24</td>
</tr>
<tr>
<td>( T_3(50) )</td>
<td>50.335</td>
</tr>
<tr>
<td>( T_4(15) )</td>
<td>94.447</td>
</tr>
<tr>
<td>( T_5(50) )</td>
<td>50.690</td>
</tr>
<tr>
<td>( T_6(15) )</td>
<td>15.254</td>
</tr>
<tr>
<td>( T_7(50) )</td>
<td>70.870</td>
</tr>
<tr>
<td>( T_8(15) )</td>
<td>14.779</td>
</tr>
<tr>
<td>( T_9(5) )</td>
<td>5.0383</td>
</tr>
</tbody>
</table>

The statistic data of log \( T \)

| \( \bar{E}(\bar{AP}) \) | -2.85 \times 10^{-2} | 4.24 \times 10^{-3} | 7.01 \times 10^{-3} | 7.03 \times 10^{-3} | 7.09 \times 10^{-3} |
| \( \sigma(\bar{AP}) \) | 9.44 \times 10^{-3} | 3.33 \times 10^{-3} | 2.95 \times 10^{-3} | 2.95 \times 10^{-3} | 2.99 \times 10^{-3} |
| \( \text{SSE}(\bar{AP}) \) | 1.03 \times 10^{-2} | 3.34 \times 10^{-3} | 3.00 \times 10^{-3} | 3.00 \times 10^{-3} | 3.04 \times 10^{-3} |

The computation cost

| CPU time (s) | 14.7 | 17.9 | 11.1 | 25.4 | 25.5 |
| \( \text{GN} \times (L_{ap}) = S_{ap} \) | 12 \times (138–156) | 11 \times (183–203) | 7 \times (175–205) | 14 \times (104–325) | 20 \times (71–506) |

*These results are calculated when \( U = U_\infty, C = D_\infty \), initial values of \( T_i(1 \leq i \leq 9) \) equal uniform 100 m\text{/day}. The signs of this table have the same meanings as those in Tables 1 and 2.
triangle elements. In order to obtain watertable measurements at observation wells, the true watertable values at discrete nodes were numerically calculated using the true transmissivity values shown in Fig. 1. Then the watertable values at the 18 observation wells were corrupted by uncorrelated Gaussian noise of variance 1. \( W_m, \delta_m, P^o \) and \( P^e \), \( \delta_y \) and \( \delta_z \) are also the same as the preceding example. The numerically calculated true watertable in \( \Omega \) ranges from 100.0 m on the southern side to 168.43 m at the northeast corner of \( \Omega \).

In Table 4 numerical results of \( T \) corresponding to two different methods are compared. In the first column the true transmissivities of the nine zones are listed in parentheses. The second column is the results of the penalty method algorithm when \( U = U_E, C = D_{c2,2} \) and \( \delta_m = 10.0 \). The third column is the results of the traditional method described by Yeh and Becker in [3] and [21]. One can see that the two results are almost equal and they are very close to the true values of \( T \). The data of \( E(\Delta P), \sigma(\Delta P) \) and SSE(\( \Delta P \)) show that there are no significant differences between the two sets of values of identified \( T \).

In Table 5 the computational costs of the above mentioned two methods are compared. The first two rows are the results of the penalty method algorithm, the next two rows the results of the traditional method by Becker and Yeh.

From Table 5 one can see the following facts:

1. For the non-linear Boussinesq aquifer model and the four different initial uniform values of \( T_j (1 \leq i \leq 9) \), both \( S_{\text{log}} \), the sum of LSQR iterative numbers in all the Gauss–Newton iterations, and the CPU time of our algorithm do not increase significantly compared with those when the aquifer model is linear, i.e., those corresponding to method 6 in Table 1. For one case both even decrease, but the CPU time of the traditional method by Becker and Yeh increases by 9–12. 16 times compared with that when the aquifer model is linear, i.e., that corresponding to “B & Y” in Table 1. On the other hand, the comparison of the CPU times of the two methods in Table 5 shows that our algorithm runs 6.5–8.6 times as fast as the traditional method. These facts demonstrate the potential of our algorithm to solve the inverse problems of more complicated non-linear aquifer models naturally and quickly on the basis of finding suitable forms of \( U \) and \( C \). The main reason of these facts is that the computational procedure of our algorithm is the same regardless of the linearity or non-linearity of the aquifer models with respect to \( h \).

2. Unfortunately, our algorithm Failed when the initial values of \( T \) are uniform 1 m²/day. How to improve this by looking for more suitable forms of \( U \) and \( C \) or other techniques remains interesting open problem. However, the incorporation of the Marquardt’s method [7,12] in our algorithm can improve the calculation so that the iteration converge. But it takes more iterations and CPU times than those of the other three unfailed cases listed in the first two rows in Table 5 where the Marquardt’s method is not used. The detail will not be stated here due to space limitation.

### Table 4
Comparison of the numerical results of our penalty method algorithm and of the traditional method by Becker and Yeh for an unconfined quasilinear aquifer model

<table>
<thead>
<tr>
<th>Methods</th>
<th>True ( T )</th>
<th>Identified ( T ) (m²/day)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T_i (150) )</td>
<td>154.70</td>
<td>157.71</td>
</tr>
<tr>
<td>( T_i (150) )</td>
<td>115.71</td>
<td>120.98</td>
</tr>
<tr>
<td>( T_i (50) )</td>
<td>53.099</td>
<td>53.425</td>
</tr>
<tr>
<td>( T_i (150) )</td>
<td>128.80</td>
<td>133.71</td>
</tr>
<tr>
<td>( T_i (50) )</td>
<td>54.394</td>
<td>53.967</td>
</tr>
<tr>
<td>( T_i (15) )</td>
<td>15.758</td>
<td>15.796</td>
</tr>
<tr>
<td>( T_i (50) )</td>
<td>49.628</td>
<td>58.607</td>
</tr>
<tr>
<td>( T_i (15) )</td>
<td>14.923</td>
<td>15.119</td>
</tr>
<tr>
<td>( T_i (5) )</td>
<td>5.0132</td>
<td>5.0164</td>
</tr>
</tbody>
</table>

The initial values of \( T \) can be any one of those in Table 5. The signs \( E(\Delta P), \sigma(\Delta P) \) and SSE(\( \Delta P \)) have the same meanings as those in Table 2.

### Table 5
Comparison of the computational cost of our algorithm and of the traditional method of Becker and Yeh for an unconfined quasilinear aquifer model

<table>
<thead>
<tr>
<th>Method</th>
<th>Uniform initial transmissivity ( T ) (m²/day)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>Penalty ( (\delta_m U = 10 U_E, C = D_{c2,2}) )</td>
<td>+∞</td>
</tr>
<tr>
<td>CPU time (s)</td>
<td>244.0</td>
</tr>
<tr>
<td>( GN \times L_{\text{log}} )</td>
<td>Failed</td>
</tr>
<tr>
<td>( B &amp; Y ) (traditional)</td>
<td>2657</td>
</tr>
</tbody>
</table>

All the signs in Table 5 have the same meanings as those in Table 1.
4. Conclusion

The least-squares penalty method objective function described in [2] is generalized by scaling the discrete governing equation residuals with a scaling matrix \( U \) determined by the current coefficient matrix of the discrete governing equations adaptively. The utilities of three different simple forms of \( U \) are shown and compared.

The generalized Gauss–Newton method [11,12,21] is used to minimize the generalized penalty method objective function. Every Gauss–Newton iteration is formulated by a linear least-squares system whose unknowns are the corrections of both the parameters and the groundwater heads on all the discrete nodes. An iterative LSQR algorithm for sparse linear least-squares problems [17,18] is chosen to solve this least-squares system. Due to the sparse iterative data structure of the LSQR algorithm, the size of the discrete governing equations can be as large as that of the traditional method. A simple but efficient preconditioner \( C \) is proposed to improve the condition number of the least-squares system.

Numerical results show that even simple and natural diagonal forms of \( U \) and \( C \) have significant effects on: (1) widening the range of the initial values of the unknown parameters within which the minimizing iteration can converge; (2) reducing the computational cost in every Gauss–Newton iteration; (3) improving the irregular distribution of the discrete governing equation residuals on the nodes.

The reducing trend of computational cost with the improvement of \( U \) and \( C \) and the sufficiently wide range of the workable values of \( \lambda_g \) shown in numerical tests demonstrate adequate potential of our algorithm to perform better and faster than the traditional method on the basis of finding suitable forms of \( U \) and \( C \). Moreover, because the computation procedure of our algorithm is the same regardless of the linearity or non-linearity of the aquifer models with respect to \( h \), numerical example shows that it can solve the inverse problem of a quasilinear Boussinesq aquifer model almost as fast as that of linear ones. This evidence indicates some promise of our algorithm to solve inverse problems of more complicated non-linear aquifer models naturally and quickly.

All the above mentioned potential of our algorithm are ultimately dependent upon the findings of more suitable forms of the scaling matrix \( U \), of the preconditioning matrix \( C \) and of the weighting matrix \( W_g \). How to look for them according to the respective distributions of the discrete governing equation residuals and the discrete aquifer model errors on different nodes, to the data structure of \( A \), remains interesting open problems.

Acknowledgements

Part of this work was done when the author visited the Institute of Mathematics, Computer Science Department of the Federal Armed Forces University in Munich and Chair of Applied Mathematics, Technical University of Munich. The author thanks Prof. Dr. Ulrich Hornung, Prof. Dr. K.-H. Hoffmann and Dr. T. Lohmann for their instruction, Dr. Stephan, Dr. Youcef, Miss Christa and Dr. Evelina for their enthusiastic help. The author is also grateful to the public “Meschach Library” which is available over Internet/AARnet via netlib, or at the anonymous ftp site thrain.anu.edu.au in the directory pub/meschach. This library provides us a complete set of C routines for performing calculations on matrices and vectors. This work was partly supported by the fund of “The research of the groundwater in Anshan” awarded by the science and technology committee of Anshan City. Many thanks to Prof. M.B. Allen and other anonymous referees for their instructions.

References