SPARSE MATRIX COMPUTATIONS ARISING IN DISTRIBUTED PARAMETER IDENTIFICATION

CURTIS R. VOGEL*

Abstract. A penalized least squares approach known as Tikhonov regularization is commonly used to estimate distributed parameters in partial differential equations. The application of quasi-Newton minimization methods then yields very large linear systems. While these systems are not sparse, sparse matrices play an important role in gradient evaluation and Hessian matrix-vector multiplications. Motivated by the spectral structure of the Hessian matrices, a preconditioned conjugate gradient method is introduced to efficiently solve these linear systems. Numerical results are presented.

Key words. distributed parameter identification, regularization, conjugate gradient iteration, preconditioning

1. Introduction. *Parameter identification* means the estimation of coefficients in a differential equation for observations of the solution. By a *distributed parameter*, we mean a coefficient which is not simply a constant, but is a function of position and/or time. Distributed parameter identification problems arise in a number of applications. Important examples include the estimation of elastic parameters from seismic observations and the determination of aquifer characteristics from groundwater flow observations. A simple mathematical model for groundwater flow, which will serve to illustrate numerical techniques to be presented in this paper, is the partial differential equation (PDE)

$$-\nabla \cdot (\kappa \nabla u) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega. \tag{1.1}$$

In groundwater flow applications, u represents fluid pressure, $\kappa(\mathbf{x})$ is the (spatially dependent) hydraulic conductivity, and $\vec{v} = -\kappa \nabla u$ is the fluid flow field. Provided the fluid is incompressible, $f(\mathbf{x})$ represents fluid gain or loss rate, e.g., due to injecting or pumping out fluid from wells in the aquifer. The parameter of interest, $q(\mathbf{x}) = \log(\kappa(\mathbf{x}))$, is known as the log conductivity and is to be estimated from observations of the solution $u(\mathbf{x})$ to the PDE (1.1). Adopting notation from [1], we represent the parameter dependent PDE by

$$\mathcal{A}(q)u = f, \tag{1.2}$$

and the observations of the solution by

$$u_{obs} = \mathcal{C}u. \tag{1.3}$$

The *parameter-to-observation map* is the composition

$$\mathcal{F}(q) \stackrel{\text{def}}{=} \mathcal{C}\, u(q) = \mathcal{C}\mathcal{A}(q)^{-1} f. \tag{1.4}$$

The measured data, which is inexact, is modeled by

$$z = \mathcal{F}(q) + \eta. \tag{1.5}$$

*Department of Mathematical Sciences, Montana State University, Bozeman, MT 59717-0240. e-mail: vogel@math.montana.edu. Work partially supported by the NSF under grant DMS-9622119 and by the Air Force Office of Scientific Research under grant F49620-96-1-0456.

The error term η accounts for factors like measurement errors and inadequacies of the mathematical model.

To estimate the parameter q given the data z, one must somehow "solve" the operator equation

$$\mathcal{F}(q) = z. \tag{1.6}$$

Obvious numerical difficulties are presented by the nonlinearity of the operator $\mathcal{F}$ and the large number of unknowns in the discretization of (1.6). A more subtle but very serious difficulty is the ill-posedness of (1.6), i.e., the lack of continuous dependence of the parameter q on the data z.

To overcome ill-posedness, one must apply regularization. Intuitively, this means replacing the ill-posed problem (1.6) with a "nearby" well-posed problem like the penalized least squares minimization

$$\min_q \frac{1}{2}||\mathcal{F}(q) - z||^2 + \alpha \mathcal{J}_{reg}(q). \tag{1.7}$$

This approach is known as *Tikhonov regularization* in the Inverse Problems community [15, 9]. Here $\mathcal{J}_{reg}$ is the regularization, or penalty, functional. Besides imposing stability, it serves to penalize "unreasonable", e.g., highly oscillatory, estimates for the parameter q. A popular choice in distributed parameter identification is the squared H^1 seminorm,

$$\mathcal{J}_{reg}(q) = \frac{1}{2}\int_\Omega |\nabla q|^2 = \frac{1}{2}\int_\Omega \sum_i \left(\frac{\partial q}{\partial x_i}\right)^2. \tag{1.8}$$

Recently, the Total Variation (TV) functional

$$\mathcal{J}_{reg}(q) = \int_\Omega |\nabla q| = \int_\Omega \sqrt{\sum_i \left(\frac{\partial q}{\partial x_i}\right)^2}, \tag{1.9}$$

has been applied (see [4, 8, 17]). This has the advantage of better recovering "blocky", possibly discontinuous, coefficients $q(\mathbf{x})$. The *regularization parameter* α in (1.7) is a small positive number which quantifies the trade-off between the goodness of fit to the data and stability.

We will address numerical linear algebraic issues in the solution of problems like (1.7) which arise in regularized distributed parameter identification. Quasi-Newton methods to handle the nonlinearity yield a sequence of approximates $q^{\nu+1} = q^\nu + s$, with

$$\mathcal{H}^\nu s = -g(q^\nu), \quad \nu = 0, 1, \ldots, \tag{1.10}$$

where the approximate Hessian takes the form

$$\mathcal{H}^\nu = \mathcal{H}_{ls} + \alpha\mathcal{L}. \tag{1.11}$$

The $\mathcal{L}$ in (1.11) arises from regularization functionals like (1.8) or (1.9), and is a symmetric, nonnegative diffusion operator, cf., [16]. In case (1.8), $\mathcal{L}$ is the negative Laplacian. Standard discretization techniques, e.g., finite difference or finite element methods, then yield large sparse positive semidefinite matrices. Unfortunately, when

these same discretization techniques are applied, the discretization of the least squares term $\mathcal{H}_{ls}$ is *not* sparse. It is symmetric, and with certain quasi-Newton schemes (e.g., Gauss-Newton linearization of the least squares term in (1.7)), it is positive definite. Moreover, it has eigenvalues which cluster at zero, which means that it has limited useful spectral content. In addition, using adjoint, or costate, techniques [5, 18], one can compute the action of the least-squares Hessian $\mathcal{H}_{ls}$ on a vector very quickly with sparse matrix methods. This suggests the use of iterative methods like conjugate gradients (CG) to solve (1.10). Since CG convergence tends to be slow, the issue of preconditioning is important. We will present a preconditioner based on the operator $\mathcal{L}$ which is effective unless the regularization parameter α becomes very small.

This paper is organized as follows: Section 2 contains an example which illustrates the important role played by sparse matrices in gradient evaluation and Hessian matrix-vector multiplications. While this example involves a specific 2-D boundary value problem, the structure and ideas presented here carry over to more general problems. The matrix computations outlined in this section are the discrete analogue of continuous adjoint approaches for computing derivatives in parameter identification and control theory. In section 3, we discuss the spectral properties of the Hessian (1.11), and we present a preconditioned conjugate gradient (PCG) method for its inversion. Numerical results are presented in the final section.

2. Gradient and Hessian matrix-vector evaluation. We first briefly discuss standard numerical optimization techniques used in parameter identification [1]. Assume a discretization of (1.2)-(1.7),

$$\min_{\mathbf{q}\in\mathbb{R}^{n_q}} \frac{1}{2}||\mathbf{F}(\mathbf{q})-\mathbf{z}||^2 + \alpha J_{reg}(\mathbf{q}), \tag{2.1}$$

where

$$\mathbf{F}(\mathbf{q}) = C\mathbf{u}(\mathbf{q}) = CA(\mathbf{q})^{-1}\mathbf{f}. \tag{2.2}$$

Let n, m, and n_q, respectively, denote the lengths of the vectors $\mathbf{u}$, $\mathbf{z}$, and $\mathbf{q}$. Then $A(\mathbf{q})$ is an $n \times n$ matrix, while C is $m \times n$. Standard secant methods like BFGS are ineffective for nonlinear least squares problems. Instead, one typically employs a Gauss-Newton approximation (see [7, p. 221]) to the least squares fit-to-data term in (2.1),

$$\begin{aligned} J_{ls}(\mathbf{q}+\mathbf{s}) &\stackrel{\text{def}}{=} \frac{1}{2}||\mathbf{F}(\mathbf{q}+\mathbf{s})-\mathbf{z}||^2 \\ &\approx \frac{1}{2}||\mathbf{F}'(\mathbf{q})\mathbf{s}+\mathbf{r}(\mathbf{q})||^2. \end{aligned} \tag{2.3}$$

Here the prime ($'$) denotes differentiation with respect to $\mathbf{q}$ and $\mathbf{r}(\mathbf{q}) = \mathbf{F}(\mathbf{q}) - \mathbf{z}$ denotes the least squares residual. Differentiating (2.3) with respect to $\mathbf{s}$ and evaluating at $\mathbf{s} = \mathbf{0}$, one obtains the true least squares gradient,

$$\mathbf{g}_{ls}(\mathbf{q}) = \mathbf{F}'(\mathbf{q})^T\mathbf{r}(\mathbf{q}). \tag{2.4}$$

Taking the second derivative with respect to $\mathbf{s}$ in (2.3) yields the Gauss-Newton approximation to the least squares Hessian,

$$H_{ls}(\mathbf{q}) = \mathbf{F}'(\mathbf{q})^T\mathbf{F}'(\mathbf{q}). \tag{2.5}$$

This matrix is $n_q \times n_q$ and symmetric. Unlike the true least squares Hessian, which contains the additional term $\mathbf{F}''(\mathbf{q})^T\mathbf{r}(\mathbf{q})$, it is also guaranteed to be positive semidefinite. From (2.2) and the fact that $A(\mathbf{q})^{-1}A(\mathbf{q}) = I$,

$$\mathbf{F}'(\mathbf{q}) = -CA(\mathbf{q})^{-1}A'(\mathbf{q})A(\mathbf{q})^{-1}\mathbf{f} = -CA(\mathbf{q})^{-1}A'(\mathbf{q})\mathbf{u}(\mathbf{q}). \tag{2.6}$$

Since $A(\mathbf{q})$ is an $n \times n$ matrix, its derivative $A'(\mathbf{q})$ is a *tensor* of size $n \times n_q \times n$.

To illustrate additional structure, consider a 2-D version of (1.1) on the open unit square, $\Omega = \{(x, y) : 0 < x < 1,\ 0 < y < 1\}$, with homogeneous Dirichlet boundary conditions,

$$\begin{aligned} -\frac{\partial}{\partial x}\left(\exp(q(x,y))\frac{\partial u}{\partial x}\right) - \frac{\partial}{\partial y}\left(\exp(q(x,y))\frac{\partial u}{\partial y}\right) &= f(x,y), \quad (x,y) \in \Omega, \\ u(x,y) &= 0, \quad (x,y) \in \partial\Omega. \end{aligned} \tag{2.7}$$

Applying Cell-Centered Finite Difference (CCFD) discretization [19] on a uniform $n_x \times n_x$ grid with lexicographical ordering of the $n = n_x^2$ coefficients of $\mathbf{u}$, one obtains a discrete analogue of (1.2),

$$A(\mathbf{q})\mathbf{u} = \mathbf{f}. \tag{2.8}$$

In the case of uniform CCFD discretization of (2.7), one can express

$$\begin{aligned} A(\mathbf{q}) &= B_x^T D_x(\mathbf{q}) B_x + B_y^T D_y(\mathbf{q}) B_y \\ &= B_{div} D(\mathbf{q}) B_{grad}, \end{aligned} \tag{2.9}$$

where $B_{div} = B_{grad}^T$ is block upper bidiagonal and $D(\mathbf{q})$ is diagonal with diagonal entries consisting of $\kappa = \exp(q)$ evaluated at the cell boundary midpoints. The matrix $A(\mathbf{q})$ is symmetric positive definite and block tridiagonal. For an illustration of the sparsity structure, see [13, p. 55]. With other finite difference and finite element discretizations on a uniform grid, this sparsity structure is retained. The cost of storing and inverting $A(\mathbf{q})$ using direct methods which utilize sparsity is $\mathcal{O}(n_x^3) = \mathcal{O}(n^{3/2})$. If state-of-the-art sparse iterative techniques for elliptic PDE's (e.g., multigrid methods) are used, the storage and computational requirements are typically much smaller. It should be noted that with other geometries or boundary conditions or more general diffusion operators, $A(\mathbf{q})$ need not be symmetric, but its sparsity structure is much the same.

If one assumes observations of $u(x, y)$ at the n cell centers, then the observation operator C is the $n \times n$ identity matrix. More generally, with m observations and (local) interpolation, C is $m \times n$ and sparse.

Next, we discuss the efficient evaluation of both the least squares gradient and Hessian matrix-vector products. The fact that $A(\mathbf{q})^{-1}$ is, in general, a full matrix makes it impractical to assemble the $m \times n_q$ matrix $\mathbf{F}'(\mathbf{q})$, cf., equation (2.6). Fortunately, all that is required is the *action* of $\mathbf{F}'(\mathbf{q})$ and its transpose on vectors. In particular, the i^{th} component of the least squares gradient in (2.4) is

$$\begin{aligned} [\mathbf{g}_{ls}(\mathbf{q})]_i \overset{\text{def}}{=} \frac{\partial J_{ls}}{\partial q_i} &= \left(\frac{\partial \mathbf{F}}{\partial q_i}\right)^T \mathbf{r}(\mathbf{q}) \\ &= -\left(C\,A(\mathbf{q})^{-1}\frac{\partial A}{\partial q_i}\mathbf{u}(\mathbf{q})\right)^T \mathbf{r}(\mathbf{q}) \\ &= -\left(\frac{\partial D}{\partial q_i}B_{grad}\mathbf{u}(\mathbf{q})\right)^T B_{div}^T A(\mathbf{q})^{-T} C^T \mathbf{r}(\mathbf{q}). \end{aligned} \tag{2.10}$$

The last equality follows from (2.9). If $D(\mathbf{q}) = \mathrm{diag}(\exp(q_i))$, then $\frac{\partial D}{\partial q_i}$ has only one nonzero entry, which is $\exp(q_i)$ in position i, i.

To evaluate the gradient, one first obtains the solution $\mathbf{u}(\mathbf{q})$ to the discrete linear system (2.8), and then computes the residual $\mathbf{r}(\mathbf{q}) = C\mathbf{u}(\mathbf{q}) - \mathbf{z}$. Next, one obtains the solution $\mathbf{y}(\mathbf{q})$ to the discrete adjoint system

$$A(\mathbf{q})^T \mathbf{y} = -C^T \mathbf{r}(\mathbf{q}). \tag{2.11}$$

After computing $\tilde{\mathbf{u}} = B_{grad}\mathbf{u}$ and $\tilde{\mathbf{y}} = B_{div}^T \mathbf{y}$, one computes the gradient components $\left(\frac{\partial D}{\partial q_i}\tilde{\mathbf{u}}\right)^T \tilde{\mathbf{y}}$ for $i = 1, \ldots, n+1$. Note that these are all sparse matrix computations, and that two sparse matrix inversions (one for $A(\mathbf{q})$ and one for its transpose) are required to obtain the gradient vector.

Matrix-vector multiplication with the Gauss-Newton approximation to the Hessian matrix, cf., (2.5), can be carried out in a similar manner. Given a vector $\mathbf{v} \in \mathbb{R}^{n+1}$,

$$\begin{aligned}
[H_{ls}\mathbf{v}]_i &= \left(CA(\mathbf{q})^{-1}\,\frac{\partial A}{\partial q_i}\mathbf{u}(\mathbf{q})\right)^T CA(\mathbf{q})^{-1}\,\left(\sum_{j=1}^{n+1}\frac{\partial A}{\partial q_j}v_j\right)\,\mathbf{u}(\mathbf{q}) \\
&= \left(\frac{\partial D}{\partial q_i}B_{grad}\mathbf{u}(\mathbf{q})\right)^T \times \\
&\quad B_{div}^T A(\mathbf{q})^{-T}C^T CA(\mathbf{q})^{-1}B_{div}\left(\sum_{j=1}^{n+1} v_j\frac{\partial D}{\partial q_j}\right) B_{grad}\mathbf{u}(\mathbf{q}) \\
&= \left(\frac{\partial D}{\partial q_i}\tilde{\mathbf{u}}\right)^T B_{div}^T A(\mathbf{q})^{-T}C^T CA(\mathbf{q})^{-1}B_{div}\left(\sum_{j=1}^{n+1} v_j\frac{\partial D}{\partial q_j}\right)\tilde{\mathbf{u}}. \qquad (2.12)
\end{aligned}$$

Given that one has $\tilde{\mathbf{u}}$ from the gradient computation, this requires inversion of sparse matrices $A(\mathbf{q})$ and $A(\mathbf{q})^T$, plus some additional sparse matrix and dot product computations. Similar adjoint techniques may be used to apply the true Hessian. See for example [14].

Finally, we examine the structure of the matrices arising in the discretization of the regularization operator in (1.7). In general the gradient of $\mathcal{J}_{reg}$ takes the form

$$\mathcal{J}'_{reg}(q) = \mathcal{L}(q)q, \tag{2.13}$$

where $\mathcal{L}(q)$ is a positive semidefinite diffusion operator with a diffusion coefficient which may depend on q. The associated boundary conditions are "natural", or homogeneous Neumann. The true Hessian of the regularization term is

$$\begin{aligned}
\mathcal{J}''_{reg}(q) &= \mathcal{L}(q) + (\mathcal{L}''(q))q \\
&\approx \mathcal{L}(q) \qquad (2.14)
\end{aligned}$$

The above approximate Hessian is symmetric and positive semidefinite, and corresponds to the "lagged diffusivity" fixed point iteration introduced in [16]. In the case of H^1 regularization (1.8), $\mathcal{L}$ is the negative Laplacian, which is independent of q. In the case of Total Variation (1.9),

$$\mathcal{L}(q)v = -\nabla \cdot \left(\frac{1}{|\nabla q|}\nabla v\right). \tag{2.15}$$

Provided that the same discretization methods are applied, the sparsity structure of the resulting $n_q \times n_q$ matrix $L(\mathbf{q})$ will be much the same as $A(\mathbf{q})$, e.g., block tridiagonal. The spectral properties of $L(\mathbf{q})$ and $A(\mathbf{q})$ should also be quite similar, except that $L(\mathbf{q})$ has a nontrivial null space consisting of constant vectors. Note however, that $H_{ls}(\mathbf{q})$ and $L(\mathbf{q})$ have drastically different structure. Due to the presence of $A(\mathbf{q})^{-1}$ in (2.6), $H_{ls}(\mathbf{q})$ is a full matrix with eigenvalues which cluster at zero.

3. Solution of Linear systems. In many parameter identification applications where minimization problems like (1.7) are solved, quasi-Newton schemes to handle the nonlinearity are rapidly convergent. The computational bottleneck is the solution of the linear systems (1.10), with (approximate) Hessian matrices of the form

$$H = H_{ls} + \alpha L, \tag{3.1}$$

where α is a small positive parameter. L is a discretization of a diffusion operator, e.g., the negative Laplacian, with natural boundary conditions. With standard discretization methods, it is symmetric, positive semidefinite, and sparse, and its null space, null(L), consists of the constant vectors.

The least squares Hessian matrix H_{ls} is harder to characterize. From a spectral standpoint, it has much in common with "blurring" matrices arising in image reconstruction (see [3]). It is the discretization of a compact operator, and hence has eigenvalues which cluster at zero. In 2-D, the blurring matrix is block Toeplitz and can be applied quickly using fast Fourier transforms (FFT's). FFT's can also be used to construct effective preconditioners for systems (3.1) arising in image deblurring [3, 17]. In parameter identification H_{ls} has no such regular structure, but the adjoint approaches sketched in the previous section can be used to compute Hessian matrix-vector products. The construction of effective preconditioners is more difficult in this case, since the explicit computation of components of H_{ls} is impractical.

Given the compactness of the continuous least squares Hessian, perhaps the most obvious approach is to precondition with L, converting (3.1) to *standard form* [11]

$$\tilde{H} = \tilde{H}_{ls} + \alpha I, \quad \tilde{H}_{ls} \quad \text{a discretization of a compact operator.} \tag{3.2}$$

In this case, the rate of convergence of PCG is known to be asymptotically superlinear [6] as the discretization level $h \to 0$. Since L has a nontrivial null space, a preliminary transformation is required. Consider the decomposition

$$\mathbf{s} = a\mathbf{v}_0 + \mathbf{s}^-, \tag{3.3}$$

where $\mathbf{s}^- \in null(L)^-$ and $\mathbf{v}_0 \in null(L)$ with $||\mathbf{v}_0|| = 1$. From the linearizations (2.3) and (2.14), one obtains the quadratic penalized least squares functional

$$Q_\alpha(a, \mathbf{s}^-) = ||a\mathbf{F}'(\mathbf{q})\mathbf{v}_0 + \mathbf{F}'(\mathbf{q})\mathbf{s}^- + \mathbf{r}||^2 + \alpha(\mathbf{s}^-)^T L\mathbf{s}^-, \tag{3.4}$$

whose minimizer satisfies the block system

$$\begin{bmatrix} \mathbf{v}_0^T\mathbf{w}_0 & \mathbf{w}_0^T \\ \mathbf{w}_0 & \mathbf{F}'(\mathbf{q})^T\mathbf{F}'(\mathbf{q}) + \alpha L \end{bmatrix} \begin{bmatrix} a \\ \mathbf{s}^- \end{bmatrix} = \begin{bmatrix} -\mathbf{v}_0^T\mathbf{g} \\ -\mathbf{g} \end{bmatrix} \tag{3.5}$$

with $\mathbf{w}_0 = \mathbf{F}'(\mathbf{q})^T\mathbf{F}'(\mathbf{q})\mathbf{v}_0$. Taking the Schur complement, one solves the reduced system

$$\overline{H}\,\mathbf{s}^- = -\overline{\mathbf{g}} \tag{3.6}$$

where

$$\overline{H} = \mathbf{F}'(\mathbf{q})^T \mathbf{F}'(\mathbf{q}) - \frac{\mathbf{w}_0 \mathbf{w}_0^T}{\mathbf{v}_0^T \mathbf{w}_0} + \alpha L \stackrel{\text{def}}{=} \overline{H}_{ls} + \alpha L,$$

$$\overline{\mathbf{g}} = \mathbf{g} - \frac{\mathbf{v}_0^T \mathbf{g}}{\mathbf{v}_0^T \mathbf{w}_0} \mathbf{w}_0.$$

One then sets

$$a = \frac{-\mathbf{v}_0^T \mathbf{g} - \mathbf{w}_0^T \mathbf{s}^-}{\mathbf{v}_0^T \mathbf{w}_0}$$

and computes $\mathbf{s}$ using (3.3).

One can show that $null(\overline{H}) = null(L)$ and that $\overline{\mathbf{g}} \in null(L)^-$. Hence, the reduced system (3.6) is consistent and has a unique pseudoinverse solution in $null(L)^-$. This pseudoinverse solution can be obtained using CG, given an initial guess in $null(L)^-$. One can precondition the reduced system (3.6) with L^-, the restriction of L to $null(L)^-$, obtaining the desired transformation of the reduced Hessian to the form (3.2).

It should be noted that the above approach is in principle very similar to that of Hanke and Hansen [11]. However, in terms of practical implementation it is quite different in that it does not require a (not necessarily square) matrix B for which

$$\mathbf{s}^T L \mathbf{s} = ||B\mathbf{s}||^2. \tag{3.7}$$

Without a factorization $L = B^T B$, it is not possible to transform the problem to allow the direct application of CGLS or LSQR or some other variant of CG specially designed for least squares problems.

4. Numerical experiments. Consider the 2-D diffusion equation (2.7) with slightly different boundary conditions, $u = 0$ on the left ($x = 0$) and right ($x = 1$) boundary edges, and no flux ($u_y = 0$) boundary conditions on the top and bottom edges. Corresponding to a point source at $(x_0, y_0) = (1/2, 1/2)$, we took $f(x, y) = \delta(x - x_0, y - y_0)$, the Dirac distribution. This yields a solution u with a logarithmic singularity at (x_0, y_0). The diffusion coefficient $\kappa(x, y)$ was taken to be piecewise smooth with a curved interface between regions of low diffusivity on the left and higher diffusivity on the right, and is shown in the upper left subplots of Figures 1 and 2. The parameter of interest, $q(x, y) = \log(\kappa(x, y))$, is also piecewise smooth. For a discussion of an alternative numerical approach to a very similar model problem, see [10].

All computations were performed using MATLAB [12]. We applied uniform CCFD discretization to the PDE (2.7). The solution $u(x, y)$ and the parameter $q(x, y)$ are defined by their values at the cell centers. With $n_x = 64$ cells on a side, the vectors $\mathbf{u}$ and $\mathbf{q}$ are both of length $n = 4096$. The resulting discretization error is insignificant compared to the simulated measurement error. Linear interpolation was used to obtain values of q at the cell interface midpoints. Due to the logarithmic singularity in u, simulated data was generated only at cell centers *outside* a neighborhood of radius of .05 of the source point (x_0, y_0). Hence, the matrix C, which is induced by the piecewise constant CCFD representation of the solution u, is diagonal with 1's at diagonal entries corresponding to nodes outside this neighborhood, and 0's elsewhere. Most of the diagonal entries are 1's. We generated simulated data according to the

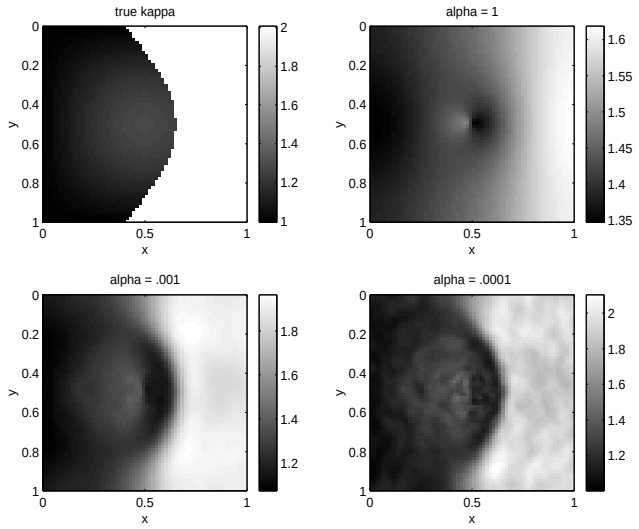

FIG. 1. *Grayscale plots showing true diffusion coefficient* $\kappa = \exp(q(x,y))$ *(upper left corner) and reconstructions obtained for* $\alpha = 1$ *(upper right),* $\alpha = 10^{-3}$ *(lower left), and* $\alpha = 10^{-4}$ *(lower right).*

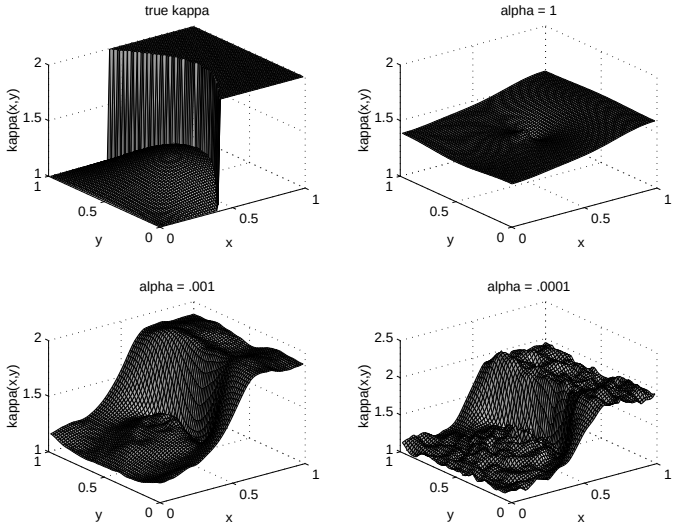

FIG. 2. *Surface plots showing true diffusion coefficient* $\kappa = \exp(q(x,y))$ *(upper left corner) and reconstructions obtained for* $\alpha = 1$ *(upper right),* $\alpha = 10^{-3}$ *(lower left), and* $\alpha = 10^{-4}$ *(lower right).*

discrete version of (1.5), with additive Gaussian error whose mean was zero and whose standard deviation was selected so that the noise-to-signal ratio $||\eta||/||C\mathbf{u}|| = 0.01$.

Figures 1 and 2 show the reconstructions obtained by solving the penalized least squares minimization problem (1.7) with TV penalty, cf., (1.9), for various values of the regularization parameter α. Note the increasing amount of detail as α decreases. Note also for small α the onset of spurious oscillations due to the inversion of noise in the data.

To estimate the parameter q, we employed a crude "continuation in α" ap-

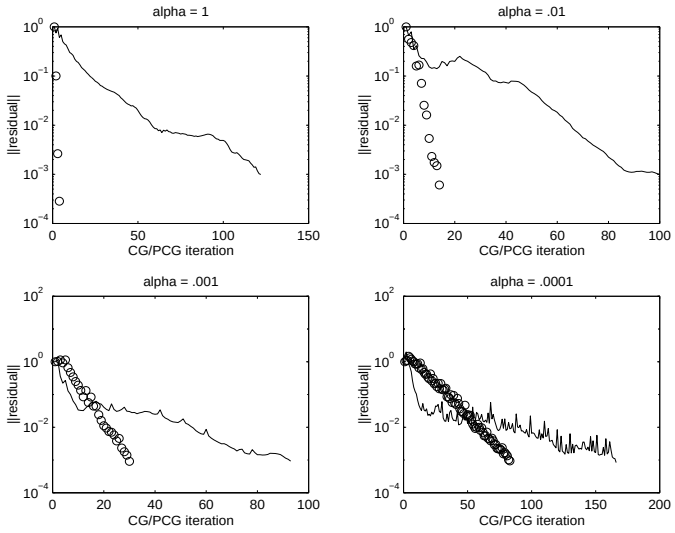

FIG. 3. *Convergence history for CG iterations (solid lines) and PCG iterations (circles), showing norm of the residual,* $||H\mathbf{s}^k + \mathbf{g}||$ *vs. iteration count* k*. The upper left subplot corresponds to* $\alpha = 1$*; upper right subplot corresponds to* $\alpha = 10^{-2}$*; lower left subplot corresponds to* $\alpha = 10^{-3}$*; and lower right subplot corresponds to* $\alpha = 10^{-4}$*.*

proach, solving minimization problems (2.1) for a decreasing sequence, $\alpha = 10^{-k}, k = 0, 1, 2, 3, 4$. For each fixed value of α, we applied a quasi-Newton iteration with the value of $\mathbf{q}$ from the previous α as the initial guess. At least for relatively large α, this eliminated the need for "globalization" (e.g., line search or trust region) to guarantee convergence, and it required relatively few total quasi-Newton iterates. An *a posteriori* scheme (see [11]) can then be applied to determine the "best" value of α. Given an optimality criterion for α, one may instead estimate α "on the fly" rather than using a predetermined sequence of values for the regularization parameter. See for example [2]. Note that the parameterization of the diffusivity, $\kappa = \exp(q)$, eliminated the need for a nonnegativity constraint on κ.

The quasi-Newton scheme consists of Gauss-Newton linearization of the least squares functional, cf., (2.5), and "lagged diffusivity" linearization of the TV penalty term, cf., (2.14). The convergence rate for each α is linear, but quite rapid. For each value of α, we took 5 quasi-Newton iterations. Each quasi-Newton iteration required the solution of a linear system of the form (3.1). To solve these systems, we applied CG with and without preconditioning. Figure 3 provides a comparison between plain CG and PCG and shows the effects of decreasing the regularization parameter α. The results shown in this figure correspond to the fifth quasi-Newton iteration in each case, but are essentially independent of quasi-Newton iteration count. The preconditioner is as described in the previous section. The performance of plain CG varied little with α. On the other hand, PCG performed very well for larger values of α, and the performance deteriorated as α decreased. For $\alpha = 1$, to attain a three order of magnitude decrease in the residual norm, PCG required less than one 30^{th} as many iterations as plain CG. This ratio decreased to about $1/7$ for $\alpha = 10^{-2}$ and $1/3$ for $\alpha = 10^{-3}$. For $\alpha = 10^{-4}$, PCG required about half as many iterations as plain CG.

The cost of each CG iteration is dominated by the Hessian matrix-vector multiplication. From (2.12), this is dominated by the cost of solving one linear system

involving the block tridiagonal matrix $A(\mathbf{q})$ and one linear system involving its transpose. Each PCG iteration requires an additional inversion of L^-, as described at the end of section 3. Assuming the cost of inverting L^- is comparable to that of inverting $A(\mathbf{q})$ and assuming that the dominant costs arise in solving linear systems, one plain CG iteration should be about two thirds as expensive as one PCG iteration. This rough cost analysis ignores a substantial amount of overhead in the Hessian matrix-vector multiplications, cf., (2.12). Consequently, from Figure 3 one observes a computational cost advantage for PCG for all values of $\alpha \leq 10^{-4}$, but the advantage becomes less pronounced as α decreases. Obviously, when applying a "continuation in α" approach, this particular preconditioner can drastically reduce total computational cost, since a several systems (3.1) with relatively large α must be solved.

We close this section with an explanation for the deterioration in PCG performance as α decreases. The analysis in [6] shows that CG is superlinearly convergent for Hilbert space operators equations in which the operator is a compact perturbation of the identity. Unfortunately, this analysis is qualitative and gives no indication of the effects of varying parameters like α. Looking at the continuous version of (3.1), we see that the Hessian is the sum of a symmetric compact operator, with eigenvalues clustering at zero, and a positive scalar multiple of a diffusion operator, with eigenvalues clustering at positive infinity. Assuming that boundary conditions are imposed so that L is invertible (it is not, but this technical difficulty is overcome by the decomposition (3.3)), our PCG scheme would yield a transformed operator of the form

$$\tilde{H} = L^{-1/2} H_{ls} L^{-1/2} + \alpha I. \tag{4.1}$$

Since $L^{-1/2}$ is now compact, $L^{-1/2} H_{ls} L^{-1/2}$ is again compact, and has eigenvalues which decay to zero more rapidly than those of H_{ls}. Consequently, the eigenvalues of $\tilde{H}$ cluster at α, cf., (3.2). This is clearly seen in the bottom two subplots in Figure 4. Unfortunately, as can be seen in the lower right subplot, there are a relatively large number of eigenvalues which are significantly greater than α and are not clustered. Note that the eigenvalue computations were performed on much smaller systems, of order $32^2 = 1024$. While the results for the order 64^2 system may be somewhat different, the qualitative behavior is the same.

REFERENCES

[1] H. T. Banks and K. Kunisch, *Estimation Techniques for Distributed Parameter Systems*, Birkhauser, 1989.

[2] P. Blomgren and T. F. Chan, *Modular Solvers for Constrained Image Restoration Problems*, UCLA CAM Report 97-52, Department of Mathematics, UCLA.

[3] R. H. Chan, T. F. Chan, and C. K. Wong, *Cosine transform based preconditioners for total variation minimization problems in image processing*, UCLA Math. Dept. CAM Report 95-23 (1995).

[4] T. F. Chan, G. H. Golub, and P. Mulet, *A nonlinear primal-dual method for total variation-based image restoration*, in ICAOS '96, 12^{th} Int'l Conf. on Analysis and Optimization of Systems: Images, Wavelets, and PDE's, Paris, June 26-28, 1996, M. Berger, R. Deriche, I. Herlin, J. Jaffre, and J. Morel, eds., no. 219 in Lecture Notes in Control and Information Sciences, 1996, pp. 241-252.

[5] G. Chavent and P. Lemonnier, *Identification de la Non-Linearite' D'Une Equation Parabolique Quasilineare*, Applied Math. and Optim., Vol. 1 (1974), pp. 121-162.

[6] J. W. Daniel, *The conjugate gradient method for linear and nonlinear operator equations*, SIAM J. Numer. Anal., Vol. 4 (1967), pp. 10-26.

[7] J. E. Dennis, Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, 1983.

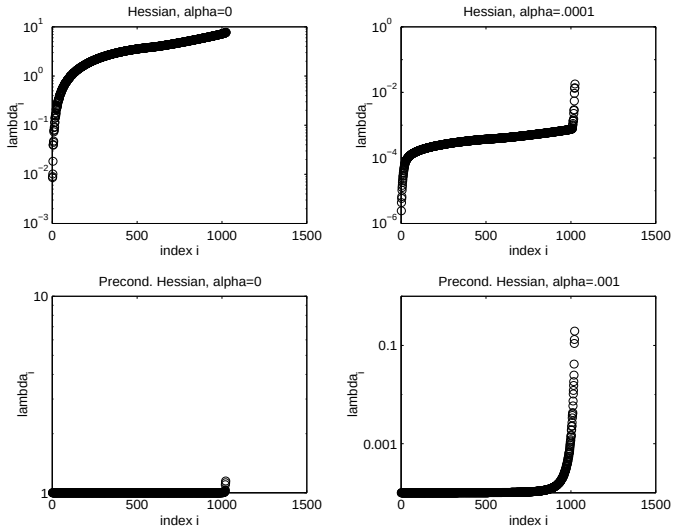

FIG. 4. *Eigenvalue distributions for the Hessian (top row) and preconditioned Hessian (bottom row of subplots) for* $\alpha = 1$ *(left subplots) and* $\alpha = 10^{-3}$ *(right subplots).*

[8] D. Dobson and F. Santosa, *Recovery of blocky images from noisy and blurred data*, SIAM J. Applied Math, Vol. 56 (1996), pp. 1181-1198.

[9] C. W. Groetsch, *Inverse Problems in the Mathematical Sciences*, Vieweg, 1993.

[10] M. Hanke, *A regularizing Levenberg-Marquardt scheme, with applications to inverse groundwater filtration problems*, Inverse Problems, Vol. 13 (1997), pp. 79-95.

[11] M. Hanke and P. C. Hansen, *Regularization methods for large-scale problems*, Surv. Math. Ind., Vol. 3 (1993), pp. 253-315.

[12] MATLAB, The MathWorks, Inc., 24 Prime Park Way, South Natick, Massachusetts 01760-1500.

[13] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Co., 1996.

[14] F. Santosa and W. W. Symes, *Computation of the Hessian for least-squares solutions of inverse problems in reflection seismology*, Inverse Problems, Vol. 4 (1988), pp. 211-233.

[15] A. N. Tikhonov, *Regularization of incorrectly posed problems*, Soviet Mathematics Doklady, Vol. 4 (1963), pp. 1624-1627.

[16] C. R. Vogel and M. E. Oman, *Iterative Methods for Total Variation Denoising*, SIAM J. Sci. Comput., Vol. 17 (1996), pp. 227-238.

[17] C. R. Vogel and M. E. Oman, *Fast, robust total variation–based reconstruction of noisy, blurred images*, to appear in IEEE Transactions on Image Processing, available at web site `http://www.math.montana.edu/~vogel`.

[18] C. R. Vogel and J. G. Wade, *Analysis of costate discretizations in parameter estimation for linear evolution equations*, SIAM J. Control and Optimization, vol. 33 (1995), pp. 227-254.

[19] A. Weiser and M. F. Wheeler, *On Convergence of Block-Centered Finite Differences for Elliptic Problems*, SIAM J. Num. Analysis, Vol. 25, 1988, pp. 351-375.