Spring 2006

Hessian Matrix-Free Lagrange-Newton-Krylov-Schur-Schwarz Methods for Elliptic Inverse Problems

Widodo Samyono
Old Dominion University

HESSIAN MATRIX-FREE LAGRANGE-NEWTON-KRYLOV-SCHUR-SCHWARZ METHODS FOR ELLIPTIC INVERSE PROBLEMS

by

Widodo Samyono
B.S. March 1986, Sepuluh Nopember Institute of Technology, Surabaya, Indonesia.
M.S. August 1995, Hampton University, Hampton, Virginia, USA.

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirement for the Degree of

DOCTOR OF PHILOSOPHY

COMPUTATIONAL AND APPLIED MATHEMATICS

OLD DOMINION UNIVERSITY
May 2006

Approved by:

David E. Keves (Director)

Fang Q. Hu (Member)

Hideaki Kaneko (Member)

Wu Li (Member)

John J. Swetits (Member)

ABSTRACT

HESSIAN MATRIX-FREE LAGRANGE-NEWTON-KRYLOV-SCHUR-SCHWARZ METHODS FOR ELLIPTIC INVERSE PROBLEMS

Widodo Samyono
Old Dominion University, 2006
Director: Dr. David E. Keyes

This study focuses on the solution of inverse problems for elliptic systems. The inverse problem is constructed as a PDE-constrained optimization, where the cost function is the L^2 norm of the difference between the measured data and the predicted state variable, and the constraint is an elliptic PDE. Particular examples of the system considered in this study are groundwater flow and radiation transport. The inverse problems are typically ill-posed due to error in measurements of the data. Regularization methods are employed to partially alleviate this problem. The PDE-constrained optimization is formulated as the minimization of a Lagrangian functional, formed from the regularized cost function and the discretized PDE, with respect to the parameters, the state variables, and the Lagrange multipliers. Our approach is known as an "all at once method." An algorithm is proposed for an inverse problem that is capable of being extended to large scales. To overcome storage limitations, we develop a parallel preconditioned Newton-Krylov method employed in a Hessian-free manner. The preconditioners have an inner-outer structure, taking the form of a Schur complement (block factorization) at the outer level and Schwarz projections at the inner level. However, building an exact Schur complement is prohibitively expensive. Thus, we use Schur complement approximations, including the identity, probing, the Laplacian, the J operator, and a BFGS operator. For exact data the exact Schur complements are superior to the inexact approximations. However, for data with noise the inexact methods are competitive to or even better than the exact in every computational aspect. We also find that nonsymmetric forms of the Karush-Kuhn-Tucker matrices and preconditioners are competitive to or better than the symmetric forms that are commonly used in the optimization community. In this study, iterative Tikhonov and Total Variation regularizations are proposed and compared to the standard regularizations and each other. For exact data with jump

discontinuities the standard and iterative Total Variation regulations are superior to the standard and iterative Tikhonov regularizations. However, in the case of noisy data the proposed iterative Tikhonov regularizations are superior to the standard and iterative Total Variation methods. We also show that in some cases the iterative regularizations are better than the noniterative. To demonstrate the performance of the algorithm, including the effectiveness of the preconditioners and regularizations, synthetic one- and two-dimensional elliptic inverse problems are solved, and we also compare with other methodologies that are available in the literature. The proposed algorithm performs well with regard to robustness, reconstructs the parameter models effectively, and is easily implemented in the framework of the available parallel PDE software PETSc and the automatic differentiation software ADIC. The algorithm is also extendable to three-dimensional problems.

ACKNOWLEDGMENTS

First of all, I would like to convey my gratitude to Professor David E. Keyes for his valuable time, his patience, his guidance and his support in researching this dissertation and in my study at Old Dominion University.

I am grateful to the other committee members, Professor Fang Q. Hu, Professor Hideaki Kaneko, Professor Wu Li, and Professor John J. Swetits, for providing their valuable time and supporting me to finish this dissertation and my study. I would especially like to give my special appreciation to Professor Wu Li for his very valuable comments on a preliminary draft of my dissertation.

My appreciation goes to Omar Ghattas and Volkan Akcelik from Carnegie Mellon University, and to George Biros from the University of Pennsylvania for their collaboration.

Many thanks are due to the PETSc and ADIC teams from Argonne National Laboratory for providing software and assistance in its use.

I also would like to thank Arun Verma from Cornell University for providing the ADMAT (Automatic Differentiation for MATLAB) and his useful hints in using it.

Financial support for my education and research came from the Indonesian Government through the Directorate General of Higher Education, Nangroe Aceh Darussalam, the Indonesian Cultural Foundation, the U.S. National Science Foundation, and Old Dominion University.

Finally, I am really grateful for my wife, my sons, and my daughter for giving me their deepest love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The twin developments of advanced algorithms and reusable, extensible, portable scientific software make the solution of partial differential equations (PDEs) increasingly routine. This opens the door to PDE-constrained optimization. Many problems in science and engineering are ultimately expressed as PDE-constrained optimization. This interdisciplinary thesis integrates recent developments in algorithms and software, and focuses them on a particularly important application: inverse problems for elliptic PDEs. These problems arise in groundwater flow, diffusive radiation transport, nonlinear heat conduction and many other fields; see [6, 80, 85] and references therein.

PDE-constrained optimization typically appears in one of three particular forms: design optimization, optimal control, and parameter identification/data assimilation/inverse problems; see [10] and references therein.

Software to handle PDE-constrained optimization is less mature than what exists for the underlying PDEs. Software for PDE problems in spatial dimensions two or three, with billions of variables, can easily be found. One such package is PETSc (the Portable, Extensible Toolkit for Scientific Computation) from Argonne National Laboratory [5]. PETSc's nonlinear solvers include Newton-like methods. As we demonstrate in this research, PDE-constrained optimization solvers can be extended algebraically from PDE solvers through Lagrangian formulations of equality-constrained optimizations. Algorithmically, we can use Newton-like methods to solve the nonlinear optimality conditions, the so-called KKT systems. By exploiting commonalities between the solutions of PDEs and PDE-constrained optimization problems, we may build on PDE solvers to solve the PDE-constrained optimization problems. The basic mechanics of the nonlinear root finding remain the same. Data structures and preconditioners need to be extended while optimization machinery needs to be added. In addition, the associated linear algebraic problems have special, exploitable structure.

The focus of this research is to solve inverse problems for a certain class of elliptic PDEs. The mathematical problem is usually formulated as PDE-constrained optimization in which the cost function is minimized with respect to the parameters

This dissertation follows the style of *SIAM Journal on Scientific Computing*

and the state variables, subject to an elliptic PDE as the constraint. Representative instances of the problems that we study are found in groundwater flow and diffusive radiation transport.

We use Newton methods to solve the first-order optimality conditions. The outer solver is a Newton method employed in a Hessian-free manner. The inner solver is a preconditioned Krylov solver. The preconditioners have an inner-outer structure, taking the form of a Schur complement (block factorization) at the outer level, which divides the variables by type, and Schwarz at the inner level, which divides the variables by domain, for parallelism. Unfortunately, the Schur complement is very expensive to form and store, even if we are fortunate enough to have an automatic differentiation toolkit for evaluation of its elements. In creating the Schur preconditioner, probing and other techniques, including automatic differentiation, that are inexpensive in the sense of being proportional in cost to the action of the Jacobian-vector product itself, are considered. Preconditioning is one of the issues that we study in this thesis.

The Schwarz and Schur components are based on domain decomposition methods, which are commonly used in solving PDE problems. The KKT residual vector can be decomposed by blocking according to the parameter vector, the state variable vector, and the Lagrange multiplier vector. To carry out the Newton iteration due to the potentially extreme large storage in using this methods we compute the Hessian-vector product by a matrix-free method. This leads to a parallel solver for the PDE-constrained problem. The efficiency and the scalability of the algorithm are discussed.

Unlike forward problems in the solution of PDEs, inverse problems are frequently ill-posed. According to Hadamard [43], a problem is "ill-posed", if a solution is not unique, or the unique solution is unstable, or there exists no solution for the problem. To overcome ill-posedness we can use a regularization. In this study we combine iterative regularization with Tikhonov and Total Variation regularization. We simply add the regularizing term to the cost function. The effectiveness of the regularizations is also discussed.

1.1 FORWARD PROBLEMS IN ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

The forward problem in elliptic PDEs can be formulated as follows:

Find $u(x)$ that satisfies partial differential equation

$$h(u, p, s, b, i; \mathbf{x}) = 0, \tag{1}$$

where u is the state variable, p is the parameter, s is the source, b is the boundary condition, and i is the initial condition.

The solutions for the forward problem can be formally written as

$$u = H(p, s, b, i; \mathbf{x}), \tag{2}$$

where p, s, b, and i are provided.

The general form of elliptic PDE we consider can be formulated as follows (See Banks *et al.* [6, 47]):

$$\nabla \cdot (-a\nabla u) + b \cdot \nabla u + cu = f \text{ in } \Omega, \tag{3}$$

with the Dirichlet boundary data

$$u = g \text{ on } \partial\Omega. \tag{4}$$

where

$$f \in H^0(\Omega).$$

Define $A = \nabla \cdot (-a\nabla u) + b \cdot \nabla u + cu$, with bounded and measurable coefficients a (symmetric real-valued $n \times n$ matrix) and complex-valued b and c in $L^\infty(\Omega)$. A is an elliptic operator if there is $\varepsilon > 0$ such that $a(\mathbf{x})\xi \cdot \xi \geq \varepsilon|\xi|^2$ for any vector $\xi \in \Re^n$ and any $\mathbf{x} \in \Omega$.

We also consider nonlinear case in which a depends on the solution u.

A particular mathematical model of an elliptic PDE that models two-dimensional steady flow in a confined aquifer takes the form:

$$\nabla \cdot (T(x, y)\nabla\phi(x, y)) = S(x, y), \tag{5}$$

subject to boundary conditions

$$\phi(x, y) = f_1(x, y) \text{ on } \Gamma_1 \subset \partial\Omega, \tag{6}$$

$$T(x, y)\nabla\phi(x, y) \cdot \mathbf{n} = f_2(x, y) \text{ on } \Gamma_2 \subset \partial\Omega, \tag{7}$$

where $\phi(x, y)$ is the piezometric head, $T(x, y)$ is the transmissivity, and $S(x, y)$ is the sink/source (See Sun [80, p. 22]).

This study is motivated by successes in solving such PDEs with Newton-Krylov-Schur, Newton-Krylov-Schwarz, and Newton-Krylov-Schur-Schwarz methods. These methods are based on domain decomposition. Hence, they are suitable for solving large-scale PDEs on distributed-memory parallel machines. For more detail about domain decomposition methods for PDEs, see the books by Smith, Bjorstad, and Gropp [78], Quateroni and Valli [72], and Widlund and Toselli [87], as well as the 15 proceedings volumes of the International Conferences on Domain Decomposition [36, 20, 21, 37, 53, 71, 54, 38, 15, 63, 61, 22, 30, 45, 59].

1.2 INTRODUCTION TO ELLIPTIC INVERSE PROBLEMS

1.2.1 Formulation of Inverse Problems

The inverse problems can be formulated as Output Least Squares as follows:
Find $p \in P_{ad}$, such that

$$\min_p E(p) = \min_p \|u(p) - d\|, \tag{8}$$

where P_{ad} is an admissible set of the unknown parameters, d is the data, and $u(p)$ satisfies

$$h(u, p, s, b, i; \mathbf{x}, t) = 0, \ \mathbf{x} \in \Omega. \tag{9}$$

Equation (9) is a partial differential equation (PDE).

If the inverse problems are ill-posed, we need to add a regularization term, $\gamma R(p)$, where $R(p)$ is a regularization functional and γ is a nonnegative regularization parameter, to the cost function.

If Equation (9) is $h(u, p, s, b; \mathbf{x}) = 0$, that is an elliptic PDE (independent of t), the inverse problem is called an inverse problem for *an elliptic system* or *an elliptic inverse problem.* If $b(\mathbf{x}) = 0$, and we know u only on the boundary, it is called *inverse conductivity.* In the groundwater modeling case, besides u on the boundary being known, where $b(\mathbf{x}) = 0$, u in the interior is also known. For examples of the elliptic inverse problems we refer to [6, 48, 49, 80, 47].

For other type of PDEs subject to output least squares, we mention hyperbolic PDEs as described in [3, 24, 47] and parabolic PDEs as described in [6, 47, 80, 40].

1.2.2 Types of Solution Methods: All At Once and Multidisciplinary Feasible Methods

Three types of methods for PDE-constrained optimization may be distinguished by the treatment of the state variables and the parameters. The first class of methods are often designated as "all at once" (AAO). In this method the state variables and the parameters are independent of each other. This class is also known as "simultaneous analysis and design" (SAND) or "one-shot" methods. The second class of methods are called "multidisciplinary feasible" (MDF). In this method the state variables are considered to be functions of the parameters. Thus, this is type of a reduced basis or generalized reduced gradient method. A third class of methods is proposed by Lewis and Schnabel [32]. They designate it as an "in-between" method. This method is a hybrid of the two previous methods.

1.2.3 Types of Solution Methods in Groundwater Modeling

In the groundwater modeling (see [80, pp. 30-34]) three methods are commonly used to solve the inverse problems: trial and error (black box) method, indirect method, and direct method.

The trial and error method is the simplest method. It only needs a forward (PDE) solver, observation data on state variables, and an expert hydro-geologist. Iteratively, starting with an initial guess, forward problem is solved, then the hydro-geologist compares the result with observation data. If the result is satisfactory, then the iterative process is stopped. If not, the hydro-geologist studies the output, then modifies the parameter to get better fit between the observation data and the model output. The method is repeated until the satisfactory result is achieved. With this method, there is no need to write a new program. Furthermore, it can be used to solve any kind of inverse problem. This is a primitive method that is still extensively used in practice. However, this method has some drawbacks. It is slowly convergent and time consuming. Furthermore, different results may be obtained by different users.

The indirect method is the output least squares method. Compared to the trial and error method, we need only to replace the procedure for modifying the parameter to get better matched solutions by a nonlinear programming procedure. In this procedure the PDE solver is called many times. Thus, this way indirectly we solve

the inverse problem through the PDE solver. This method can reduce human effort, and begin to solve the problem quickly. However, use of the direct (forward) solver inside the iterative process leads to long computational times if the data of the PDE is large.

The direct method can be explained as follows:

First, discretize the PDE, then rearrange the discrete equations such that we obtain overdetermined equations

$$H\mathbf{p} - \mathbf{r} = \mathbf{0}, \tag{10}$$

where H is an $L \times M$ matrix, $\mathbf{r}$ an L row-vector, L the number of observations, and M the number of unknown parameters, where $L > M$.

Second, let the residual of lth equation be

$$\mu_l(\mathbf{p}) = H_{l1}p_1 + H_{l2}p_2 + \cdots + H_{lm}p_m - r_l, \ (l = 1, 2, \ldots, L). \tag{11}$$

Third, find $\bar{\mathbf{p}}$ that is the solution of the optimization problem

$$\min D(\mathbf{p}) = \sum_{l=1}^{L} [\mu_l(\mathbf{p})]^2, \ \mathbf{p} \in P_{ad}, \tag{12}$$

where

$$P_{ad} = \{p_m, \underline{p}_m \leq p_m \leq \overline{p}_m\}, \tag{13}$$

where $\underline{p}_m$ is the lower bound, and $\overline{p}_m$ is the upper bound of the parameter p_m.

This method does not require a PDE solver. However, in order to use this method the interpolation process is needed in order to get all nodal values of the distributed state variables. This can result extra interpolation error (noise). Thus, this method is very sensitive to any noise associated with the observation data.

These methods originally are proposed by Neuman in 1973 [68]. Our approach is the hybrid of the last two methods, enhanced with some new sophisticated techniques that are suitable for solving large-scale problems.

1.2.4 Analytical and Numerical Solutions

In general there are two ways of solving inverse problems: analytical approaches and numerical approaches. The advantages of the analytical approaches are that we can have accurate closed form solutions. However, we usually can solve only very simple problems and well-posed problems. In contrast, the numerical approach can

potentially solve extremely large problems and ill-posed problems. Although the solutions are only approximate, they can be used to inform many realistic situations.

Analytical approaches to inverse problems for elliptic PDEs can be found in [58] for groundwater modeling, and in [29, 4] for inverse conductivity.

1.3 SOLUTION METHODS AND RELATED WORK

Our study focuses on solving elliptic inverse problems by means of numerical methods based on output least squares formulations. We describe some numerical methods to solve inverse problems in elliptic systems using the output least squares formulations and their variants.

First, if we assume the forward problem is well-posed, then we can obtain an unconstrained regularized output least square minimization problem,

$$\min_p \|F(p) - d\| + \alpha R(p) = \min_p \|C\mathcal{A}(p)^{-1}f - d\| + \alpha R(p), \tag{14}$$

where $\mathcal{A}(p)u = f$ is the forward PDE problem, and C is called the state-to-observation map.

This approach can be found in Vogel [86, 85]. For large-scale problem, this approach may be costly, since, in order to solve the minimization problem, we need to solve the forward PDE problems for different parameters p in the iterative process. Also this method can fail to converge for finer grids. This method is known as the "following feasible path" approach or Multidisciplinary Feasible Method (MFD). For comparison of this method with the "All at Once" method, see Haber and Ascher [42].

Second, for iterative regularization methods of parameter identification and their variants, we mention Burger and Muhlhuber [16, 17] and the references therein. Here, the problem is

$$\min_{(u,p)} \frac{1}{2}\|Eu - z\|^2 \tag{15}$$

subject to

$$e(u,p) = f, \tag{16}$$

where $e(u,p) = f$ is the partial differential equation (PDE).

The first method is proposed as follows:

Method 1 (Iteratively regularized sequential quadratic programming (IRSQP) method).

Let (u_0, p_0, λ_0) be a given initial value and let $(\beta_k)_{k\in\mathcal{N}}$, where $\mathcal{N}$ is the set of natural numbers, be a bounded sequence of positive real numbers. The IRSQP method consists of the iteration procedure

$$(u_{k+1}, p_{k+1}, \lambda_{k+1}) = (\bar{u}_k, \bar{p}_k, \bar{\lambda}_k), \tag{17}$$

where $(\bar{u}_k, \bar{p}_k)$ is the minimizer of the quadratic programming problem

$$\min_{(u,p)} \frac{1}{2}\|Eu - z^\delta\|^2 + \frac{\beta_k}{2}\|p - p_k\|^2 + \langle \frac{1}{2}\lambda_k, e''(u_k, p_k)(u - u_k, p - p_k)^2\rangle \tag{18}$$

subject to the linear constraint

$$e(u_k, p_k) + e'(u_k, p_k)(u - u_k, p - p_k) = f, \tag{19}$$

where λ_k is the corresponding Lagrange multiplier, z^δ is the data with noise level δ, and prime denotes the differentiation with respect to each argument.

The implementation of this method can be costly and difficult due to the second derivative in the functional.

A second method is proposed as follows:

Method 2 (Levenberg-Marquardt sequential quadratic programming (LMSQP) method).

Let (u_0, p_0) be a given initial value and let $(\beta_k)_{k\in\mathcal{N}}$ be a bounded sequence of positive real numbers. The LMSQP method consists of the iteration procedure

$$(u_{k+1}, p_{k+1}) = (\bar{u}_k, \bar{p}_k), \tag{20}$$

where $(\bar{u}_k, \bar{p}_k)$ is the minimizer of the quadratic programming problem

$$\min_{(u,p)} \frac{1}{2}\|Eu - z^\delta\|^2 + \frac{\beta_k}{2}\|p - p_k\|^2 \tag{21}$$

subject to the linear constraint

$$e(u_k, p_k) + e'(u_k, p_k)(u - u_k, p - p_k) = f. \tag{22}$$

Theoretically this method is effective for certain conditions. As discussed in [17], the numerical examples show that the method outperforms the ordinary Levenberg-Marquardt method and Broyden's method. As a Newton method, the Hessian matrices may be ill-conditioned, hence it needs more investigation with respect to preconditioners. As an "all at once method" the usage of the storage for large-scale problems should also be considered.

Third, for Augmented Lagrangian methods and their variants, we may check [48, 49], and the references therein.

The inverse problem is stated as follows:

$$\min_{u,p} F(p,u) = \frac{1}{2}\|u-z\|^2 + \frac{\beta}{2}N(p), \tag{23}$$

subject to

$$-\nabla\cdot(p\nabla u) = f, \tag{24}$$

$$|p|_H^2 \le \gamma, \tag{25}$$

$$\alpha \le p \tag{26}$$

on Ω, in the two dependent variables p and u.

The algorithm to solve the inverse problem is as follows:

Step 1. Choose $\lambda^1 = \mu^1 = 0$, $\{c_k\}_{k=1}^{\infty}$ monotonically increasing $c_k > c_0$.

Step 2. Put $k = 1$, $u_0 = z$.

Step 3. Determine p_k from

$$\begin{aligned}\min_p \Big\{ &\frac{\beta}{2}N(p) + \langle \lambda^k, e(p,u_{k-1})\rangle + \frac{c_k}{2}|e(p,u_{k-1})|^2 + \mu^k \hat{g}(p,\mu^k,c_k) \\ &+ \frac{c_k}{2}\hat{g}(p,\mu^k,c_k)^2\Big\}\end{aligned} \tag{27}$$

over $p \in H^2$ subject to $p \ge \alpha$.

Step 4. Determine u_k from

$$\min_u \left\{ \frac{1}{2}|u-z|^2 + \langle \lambda^k, e(p_k,u)\rangle + \frac{c_k}{2}|e(p,u_{k-1})|^2 \right\}. \tag{28}$$

Step 5. $\lambda^{k+1} = \lambda^k + c_k e(p_k,u_k)$ and $\mu^{k+1} = \mu^k + c_k\hat{g}(p_k,\mu^k,c_k)$.

Step 6. If convergence is achieved, stop; otherwise put $k = k+1$ and go to Step 3.

This method performs very well for many of the numerical test cases. Globally this algorithm has a good convergence behavior but converges locally at rather slow rate. Furthermore, the decouple optimizations may cause a problem, if we realize the methods in the larger scale problems.

Fourth, we consider Reduced SQP Methods from [60] as follows:

For given observation data $z \in H_0^1(\Omega)$ find p such that

$$-\nabla\cdot(p\nabla u) = g \text{ in } \Omega \tag{29}$$

and

$$u = 0 \text{ on } \partial\Omega. \tag{30}$$

The inverse problems are formulated by the least squares formulation as follows:

$$\min_{(u,p)\epsilon H^1 \times H_0^1} \frac{1}{2}\|u - z\|_{H_0^1}^2 + \frac{\beta}{2}\|p\|_{H^1}^2, \tag{31}$$

such that

$$e(u,p) = (\Delta)^{-1}(\nabla \cdot (p\nabla u) + g) = 0, \tag{32}$$

where Δ is the Laplace operator with Dirichlet boundary conditions.

The problem is solved by using a Lagrangian method with the Lagrangian functional

$$L(u,p,\lambda) = \frac{1}{2}\|u - z\|_{H_0^1}^2 + \frac{\beta}{2}\|p\|_{\tilde{H}}^2 + \langle \lambda, e(p,u)\rangle_{H_0^1}, \tag{33}$$

where $\tilde{H} \subset L^\infty$.

With changes in the notation, this reduced SQP algorithm for the inverse problems can be described as follows:

For every $h \in \tilde{H}$ is defined the operator

$$A(h) : H_0^1 \to H^{-1} \tag{34}$$

by

$$A(h)u = -\nabla \cdot (h\nabla u). \tag{35}$$

The Fréchet derivative of e is given by

$$e'(p,u)(h,v) = \Delta^{-1}(A(h)u + A(p)v). \tag{36}$$

The Fréchet derivative of the Lagrangian is

$$L_x(q,u,\lambda)(h,v) = \langle u - z, v\rangle_{H_0^1} + \beta\langle p, h\rangle_{\tilde{H}} + \langle \lambda, \Delta^{-1}(A(h)u + A(p)v)\rangle_{H_0^1}. \tag{37}$$

ALGORITHM

(1) Choose $p_0 > \nu$, $\beta \geq 0$. Set $\lambda_0 = 0$, $u_0 = z$, and

$$B_0 = \beta I + J[\nabla u_0 \cdot (\nabla A(p_0)^{-1}\Delta A(p_0)^{-1}\nabla)((\cdot)\nabla u_0)]. \tag{38}$$

(2) Evaluate right-hand side and solve for $\omega_k \in \tilde{H}$:

$$B_k\omega_k = -\beta p_k - J[\nabla u_k \cdot \nabla A(p_k)^{-1}\Delta(u_k - z))]. \tag{39}$$

(3) Set

$$(p_{k+}, u_{k+}) = (p_k, u_k) + (\omega_k, -A(p_k)^{-1}A(\omega_k)u_k) \tag{40}$$

$$(p_{k+1}, u_{k+1}) = (p_{k+}, u_{k+}) + (0, -A(p_k)^{-1}\Delta e(p_k, u_k)). \tag{41}$$

(4) Set

$$\lambda_k = -A(p_{k+1})^{-1}\Delta(u_{k+1} - z). \tag{42}$$

(5) Set

$$\begin{aligned} y_k = \beta(p_{k+} - p_k) + J[&-\nabla\lambda_k \cdot \nabla u_{k+} + \nabla u_k \cdot \nabla A(p_k)^{-1}\Delta(u_{k+} - u_k) \\ &- \nabla u_k \cdot \nabla A(p_k)^{-1}\nabla(p_{k+}\nabla\lambda_k)]. \end{aligned} \tag{43}$$

(6) If $\omega_k = 0$ set

$$B_{k+1} = B_k, \tag{44}$$

else

$$B_{k+1} = B_k + \frac{\langle y_k, \cdot\rangle_{\tilde{H}}}{\langle y_k, \omega_k\rangle_{\tilde{H}}} y_k - \frac{\langle B_k\omega_k, \cdot\rangle_{\tilde{H}}}{\langle \omega_k, B_k\omega_k\rangle_{\tilde{H}}} B_k\omega_k. \tag{45}$$

(7) Set $k = k + 1$ and return to (2)

The method looks better than the augmented Lagrangian method with respect to local convergence. However, for certain cases it does not converge. Thus, it needs a globalization strategy that goes beyond the previous method.

For methods used in inverse problems other than parametric identification, we refer to the book by Vogel [86], and Isakov [47]. We do not discuss them here, because the methods cannot be used for elliptic inverse problems.

1.4 THE PROPOSED ELLIPTIC INVERSE PROBLEM ALGORITHM

Although the methods we mentioned above are effective, they are usually not readily applicable for large-scale problems and for parallel computations. Another feature that is different from our proposed method is the order in solving the inverse problem. Whereas the methods above use the "optimize then discretize" approach, our approach is based on the "discretize then optimize". We use this approach because it is easily implemented in an available PDE solver.

Our proposed methods have the following characteristics:

- simple and easily implemented in an available PDE solver,

- computational cost comparable to solving the forward problems,
- applicable for large-scale problems,
- applicable for parallel machines.

This work is actually the extension of our previous work [55]. We owe considerable debts to Biros and Akcelik. From Biros [12, 13, 14, 11] we borrow the ideas of using full space solvers and Schur preconditioners. His solver uses Newton Full Sequential Quadratic Programming (SQP), while ours is Newton Full Hessian matrix-free; see Chapter 2 for the details. His Schur preconditioner approximation uses Quasi-Newton SQP, while our preconditioner has an outer-inner structure in which the outer part is a Schur complement, which is approximated by probing techniques and other operators, and the inside are Schwarz methods. Our problem domains are also different. Biros applies his algorithm to control optimization, while ours is applied to an inverse problem that is even more challenging due to ill-posedness of the problem. From Akcelik [2] we borrow some regularization strategies. He uses Tikhonov and Total Variation regularizations, while we combine iterative regularizations with Tikhonov and Total Variation regularizations. Our problem domains are also different. Akcelik applies his algorithm to inverse problems in wave propagations which are hyperbolic, while ours is an elliptic inverse problem. In their preconditioners and solvers they prefer to preserve symmetry in the KKT matrices. On the other hand, we prefer to use nonsymmetric permutations of the KKT matrices in our first numerical test case (a linear elliptic PDE). Therefore, they use conjugate gradient (CG) methods with the LU decomposition or symmetric block preconditioners. Meanwhile, we use upper triangular block (by using the U part of the LU decomposition), nonsymmetric block (by dropping the blocks $(1,3)$ and $(2,3)$), and symmetric block preconditioners (by dropping further the block $(1,2)$); see Equations (101), (102), and (103), respectively. We also propose a new preconditioner that we call a *nonlinear preconditioner*.

As with Biros and Ghattas [12, 13, 14, 11], Haber and Ascher [42] report on their preconditioners and KKT systems on parameter estimation problems. Although they discussed nonsymmetric KKT systems, they do not use it. They prefer to use symmetric KKT formulations. Hence, they use preconditioned conjugate gradient (PCG) and preconditioned symmetric quasi minimum residual (PSQMR). Furthermore, they also use approximations for the reduced Hessians or the Schur complements that are different from ours. They use Gauss-Newton approximations. Although we also use

this approach for our second numerical test case (a nonlinear elliptic PDE), we use it only in the preconditioner. Our second numerical example shows that for this class of problems it may be better to use a nonlinear preconditioner; see 3.6.

Our iterative regularizations are inspired by the Landweber iterative regularization and its variants; see Burger and Muhlhuber [16, 17] and the references therein. However, they use the version of Tikhonov regularization functional without derivative. Hence, the second derivative block of the KKT systems with respect to the parameter will be an identity matrix multiply by the regularization parameter. This looks like a Levenberg-Marquadt method. Therefore, they call their method as Levenberg-Marquardt sequential quadratic programming (LMSQP). Our methods use a Tikhonov functional with derivative and a Total Variation functional. In so, we expect to have the advantages of both iterative and functional regularizations in our algorithm. One of the advantages of the functional regularizations is that they converge quickly. Meanwhile, the iterative regularizations are very stable. Our numerical test cases show that we can get these two advantages by keeping the regularization parameters large enough or not rapidly decreasing. Burger and Muhlhuber also use symmetric KKT systems.

Our formulation of numerical Example 1 (linear elliptic PDE) uses $\kappa(x) = \beta(x)$ for the diffusivity parameter, as in Burger and Muhlhuber [16, 17]. Meanwhile, Haber and Ascher [42], also Vogel [85], use the diffusivity parameter of the form $\kappa(x) = e^{\beta(x)}$. These different formulations can create different KKT systems. The KKT matrices of the former are sparser than those of the later. In the former, the first-order derivatives of the Lagrangian functionals with respect to the parameters may also result in singular or closely singular Jacobian matrices. Hence, the singular Jacobian matrices may rule out of using Gauss-Newton methods. To regularize, we keep the regularization term in the preconditioners, as in Haber and Ascher [42]. Preserving symmetry in the KKT matrices can also can lead to highly indefinite matrices, which is complementary disadvantage. We discuss this issue more details in 3.8.

1.4.1 Formulation of the Inverse Problem

Consider the domain Ω, where the state variables u are known (as collected data) inside the domain Ω and on the boundary $\partial\Omega$. We assume that in the domain Ω there are known sources/sinks S. The inverse problem seeks to discover the properties of

the medium from the known data. If we assume that the mathematical model of the discretized partial differential equation is $h(p, u, \mathbf{x}) = 0$, where u is the state variable, p is the parameter, and $\mathbf{x} \in \Omega$, we need to find p from the known data.

Assume that the medium is inhomogeneous. Because analytical solutions usually exist only for highly structured problems, assume the problem may not be solved analytically. Furthermore, the discretized forward problem is very large. Hence, the use of the classical output least squares method must be avoided. We pursue a parallel numerical algorithm that is efficient and scalable.

The mathematical formulation of the inverse problem in discrete form is as follows:

$$\min_{u,p} \frac{1}{2}\|Cu - u_d\|^2, \tag{46}$$

subject to

$$h(u, p) = 0, \tag{47}$$

where C is the state-to-observation map, u is the computed state variable, u_d is the state data, p is the parameter that we would recover, and $h(p, u) = 0$ represents the discretized elliptic PDE.

We assume that the observed data can be expressed as

$$u_d = Cu + \delta, \tag{48}$$

where δ is the noise in the data.

1.4.2 Ill-posedness and Regularization

A problem that we most likely encounter is ill-posedness. This is due, at least, to errors in the collected data. Thus, the solution to the inverse problem may be impossible to approximate, as given. We accommodate this problem by using regularization. This approach shifts the ill-posed problem to a nearby well-posed problem. Specifically, we add the regularization term to the cost function.

In this study we first describe Tikhonov regularization. This regularization is simpler and less expensive to implement than the Total Variation regularization. There are some variants of this regularization. However, this regularization can not handle a sharp discontinuity on the estimated parameter very well. We observe the Total Variation regularization that can overcome such a problem. However, this regularization is more expensive than Tikhonov regularization in the sense of iteration count

or computation time. We can see this from the work of Akcelik *et al.* [2, 3]. Another regularization that we consider is an iterative regularization. This regularization is different than the two previous regularizations in that it evolves the regularization parameter during the iterative process, whereas the two previous regularizations are fixed. We study the combination of these two type of regularization, iterative regularization and Tikhonov regularization, or Total Variation regularizations, in search of a faster and more effective algorithm. We will discuss these topics more detail in chapter 4.

1.4.3 Lagrange-Newton Method and Globalization Strategy

If we include the regularization in the formulation, the inverse problem can be defined as follows:

$$\min_{u,p} \left[c(u,p) + \gamma R(p)\right], \tag{49}$$

subject to

$$h(u,p) = 0, \tag{50}$$

where $c(u,p) = \frac{1}{2}\|Cu - u_d\|^2$, γ is the regularization parameter, and $R(p)$ is the regularization functional.

Algebraically, by introducing Lagrangian multipliers λ, we can reformulate the constrained optimization as an unconstrained optimization as follows:

$$\min_{u,p} \mathcal{L}(u,p,\lambda) \equiv \min_{u,p} \left[c(u,p) + \gamma R(p) + \lambda h(u,p)\right], \tag{51}$$

where u is the state variables, p is the parameters, and λ is the Lagrangian multipliers.

To solve this problem, we formulate the first- and second-order optimality conditions [69].

As shown in Chapter 2 and in [55] (without the regularization term), the first-order optimality conditions are:

$$\frac{\partial \mathcal{L}}{\partial u} \equiv \frac{\partial c}{\partial u} + \lambda^T \frac{\partial h}{\partial u} = 0, \tag{52}$$

$$\frac{\partial \mathcal{L}}{\partial p} \equiv \frac{\partial c}{\partial p} + \gamma \frac{\partial R}{\partial p} + \lambda^T \frac{\partial h}{\partial p} = 0, \tag{53}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} \equiv h = 0. \tag{54}$$

These are known as the Karush-Kuhn-Tucker (KKT) first-order optimality conditions. This is a nonlinear system. One of the methods to solve this nonlinear system

is Newton's method. It is known that this method can converge quadratically and be scalable. In this study we base our method on the "all at once method", where the system is solved simultaneously for all unknowns. One of the prominent disadvantages is the requirement of large storage. In order to avoid this problem, we solve the system in Hessian matrix-free manner. We explain this in more detail in Chapter 2.

If we use Newton's method to solve the first order optimality conditions above, we repeatedly encounter the following linear system of equations:

$$\begin{bmatrix} (c_{uu}+\lambda^T h_{uu}) & (c_{up}+\lambda^T h_{up}) & h_u^T \\ (c_{pu}+\lambda^T h_{pu}) & (c_{pp}+\gamma R_{pp}+\lambda^T h_{pp}) & h_p^T \\ h_u & h_p & 0 \end{bmatrix} \begin{pmatrix} \delta u \\ \delta p \\ \delta\lambda \end{pmatrix} = - \begin{pmatrix} c_u+\lambda^T h_u \\ c_p+\gamma R_p+\lambda^T h_p \\ h \end{pmatrix}$$

or

$$\begin{bmatrix} H_{uu} & H_{up}^T & J_u^T \\ H_{pu} & H_{pp}+\gamma R_{pp} & J_p^T \\ J_u & J_p & 0 \end{bmatrix} \begin{pmatrix} \delta u \\ \delta p \\ \lambda_+ \end{pmatrix} = - \begin{pmatrix} g_u \\ g_p+\gamma R_p \\ h \end{pmatrix}, \tag{55}$$

where $H_{ab} \equiv \frac{\partial^2 c}{\partial a \partial b} + \lambda^T \frac{\partial^2 h}{\partial a \partial b}$,$J_a \equiv \frac{\partial h}{\partial a}$,$R_b \equiv \frac{\partial h}{\partial b}$,$R_{bb} \equiv \frac{\partial^2 h}{\partial^2 b}$, and $g_a = \frac{\partial c}{\partial a}$,for $a, b \in \{u, p\}$, and where $\lambda_+ = \lambda + \delta\lambda$.

or

$$M_K \delta x = g, \tag{56}$$

where M_K is the KKT matrix (Hessian matrix), δx is the Newton step, and g is the gradient vector.

Hence, iteratively we will compute $x_{k+1} = x_k + \alpha_k \delta x$, for $k = 0, 1, \ldots, k_{max}$, where α_k is a robustification parameter such as is set by a line search method. We stop the iteration process when the gradient reaches a certain maximum tolerance or the iteration reaches a maximum value with a failure.

The Newton method is locally convergent but requires a globalization strategy. Two popular globalization strategies are line search methods and trust region methods, with many variants. We use a basic line search that works very well for the 2-D numerical test cases. However, we did not require globalization strategies in our 1-D examples, even without continuation methods as in [85], where the previous computations are used as the initial guesses for the next computations with decreasing sequential regularization parameters $\gamma = 1,\ 0.1,\ 0.01, \ldots$.

1.4.4 Ill-conditioned Problems and Preconditioning

The KKT matrix in (55) is frequently indefinite and ill-conditioned. This can cause the iteration process to be very slow or even divergent. Hence, we need a good preconditioner. The preconditioners that we use are based on domain decomposition preconditioners, commonly used in solving PDEs. In this study we discuss Schwarz preconditioners, Schur preconditioners, and their combinations. We propose a Schur-Schwarz combination preconditioner. In outer level we use Schur preconditioner based on decomposition of the state variables, the design variables, and the Lagrangian multiplier. Meanwhile, in the inner level we use Schwarz method based on the domain decomposition method. For more details about Schur and Schwarz preconditioners, we refer to [78, 87]. We will discuss our proposed preconditioners in more details in Chapter 3.

1.5 TEST CASES

We use 1-D and 2-D synthetic numerical models as test cases. These test cases are found in important applications, such as in groundwater flow and radiation transport.

1.6 OUTLINE OF THE DISSERTATION

The dissertation is organized as follows:

- We discuss in Chapter 2 methods of solving PDE-constrained optimization problems of Lagrange-Newton-Krylov-Schur-Schwarz (LNKSS) type, first by components of the algorithm, and then integrated.
- Linear and nonlinear preconditioning strategies are discussed in Chapter 3. We describe experiments that show the effectiveness of the preconditioners, such as the condition numbers and the spectra of the matrices. We also discuss the nonlinear preconditioners and the comparison with the linear ones for the nonlinear elliptic inverse problem (Numerical Example 2). We also discuss the preferences in selecting the nonsymmetric KKT matrices.
- The regularization strategies are the focus of Chapter 4. We observe some regularization strategies, and propose iterative Tikhonov and Total Variation regularization. We also discuss the proposed iterative regularization and compare

to the regular regularization. We also discuss a possible new stopping criteria for using the proposed iterative regularization in the proposed algorithm.

- In Chapter 5 we discuss how to implement the algorithm in the parallel library PETSc (a software package for solving algebraic systems that commonly arise in PDE problems) and ADIC (an automatic differential tool for generating derivative code). We discuss how PETSc and ADIC can be used together, and what the benefits are. We also observe how the data structure between the PDE and the optimization can be matched.

- In Chapter 6 the numerical experiments of the LNKSS are presented. We discuss the robustness of the algorithm, and the effects of the regularizations in recovering the parameters.

- In Chapter 7 we summarize our research and note some future directions.

CHAPTER 2

LAGRANGE-NEWTON-KRYLOV-SCHUR-SCHWARZ

In this section, we describe the differences and the similarities in solving PDE and PDE-constrained optimization problems. As in [12, 13, 14, 11] and [55], let $h(p,u) = 0$ be the discretized systems of (generally nonlinear) PDEs, and $c(p,u)$ be the objective function. Then, by the Lagrangian method we seek the stationary point of the regularized Lagrangian functional

$$\mathcal{L}(p,u,\lambda) \equiv c(p,u) + \gamma R(p) + \lambda^T h(p,u), \tag{57}$$

where p is the design variable, u is the state variable, γ is the regularization parameter, $R(p)$ is the regularization functional, and λ is the vector of Lagrange multipliers.

As described in [55] (with the addition of a regularization term), the first-order optimality conditions are

$$\frac{\partial \mathcal{L}}{\partial u} \equiv \frac{\partial c}{\partial u} + \lambda^T \frac{\partial h}{\partial u} = 0, \tag{58}$$

$$\frac{\partial \mathcal{L}}{\partial p} \equiv \frac{\partial c}{\partial p} + \lambda^T \frac{\partial h}{\partial p} + \gamma \frac{\partial R}{\partial p} = 0, \tag{59}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} \equiv h = 0. \tag{60}$$

We seek a correction

$$\begin{pmatrix} \delta u \\ \delta p \\ \delta \lambda \end{pmatrix}$$

to the iterate

$$\begin{pmatrix} u \\ p \\ \lambda \end{pmatrix}.$$

With subscript notation for the partial derivatives, we can write the Newton correction (KKT) equations as

$$\begin{bmatrix} (c_{uu} + \lambda^T h_{uu}) & (c_{up} + \lambda^T h_{up}) & h_u^T \\ (c_{pu} + \lambda^T h_{pu}) & (c_{pp} + \gamma R_{pp} + \lambda^T h_{pp}) & h_p^T \\ h_u & h_p & 0 \end{bmatrix} \begin{pmatrix} \delta p \\ \delta u \\ \delta \lambda \end{pmatrix} = - \begin{pmatrix} c_u + \lambda^T h_u \\ c_p + \gamma R_p + \lambda^T h_p \\ h \end{pmatrix}$$

or

$$\begin{bmatrix} H_{uu} & H_{up} & J_u^T \\ H_{pu} & H_{pp}+\gamma R_{pp} & J_p^T \\ J_u & J_p & 0 \end{bmatrix} \begin{pmatrix} \delta u \\ \delta p \\ \lambda_+ \end{pmatrix} = - \begin{pmatrix} g_u \\ g_p+\gamma R_p \\ h \end{pmatrix}, \tag{61}$$

where $H_{ab} \equiv \frac{\partial^2 c}{\partial a \partial b} + \lambda^T \frac{\partial^2 h}{\partial a \partial b}$,$J_a \equiv \frac{\partial h}{\partial a}$,$R_b \equiv \frac{\partial h}{\partial b}$,$R_{bb} \equiv \frac{\partial^2 h}{\partial^2 b}$, and $g_a = \frac{\partial c}{\partial a}$, for $a, b \in \{u, p\}$, and where $\lambda_+ = \lambda + \delta\lambda$.

In the Newton framework, it immediately can be seen that solving $h(p, u) = 0$ with respect to u in the PDE problem requires operations that are a subset of those involve in the KKT equations. Fortunately, by exploiting the similarities in the two problems, we may solve the PDE-constrained optimization problems efficiently.

As mentioned in [55, 12, 13, 14, 11], we may solve the KKT equations by Newton Reduced SQP or Full-space Lagrange-NKS method. The full-space method may be preferred due to the opportunity to reuse the available PDE solver. However, good preconditioners are needed. To form the preconditioners, not to mention the matrix for the solver, is expensive, even if we can build all the KKT matrix components by an automatic differentiation toolkit. Hence, we propose Hessian Matrix-free Lagrange-Newton-Krylov-Schur-Schwarz methods.

First, we discuss the components of the methods in the following sections and subsections.

2.1 NEWTON METHODS

Let $h(x) = 0$ be a function representing the residuals of a discretized PDE. To solve this problem using a Newton method, we need the Jacobian (first partial derivatives) of the function, $J(x)$. Iteratively with an initial guess $x^{(0)}$, the solutions is obtained by correcting to the current $x^{(k)}$:

$$x^{(k+1)} = x^{(k)} + \delta x, \tag{62}$$

where δx is the step. For a good initial guess $x^{(0)}$ the method converges rapidly [50, 51].

2.1.1 Inexact Newton

The class of *inexact Newton methods* is an alternative of the Newton methods if the number of unknowns in the nonlinear equation, $h(x) = 0$, is large as discussed in

[31]. The method can be described as follows: For initial guess $x^{(0)}$, the solution is obtained by correcting the current $x^{(k)}$:

$$x^{(k+1)} = x^{(k)} + \lambda_k \delta x, \tag{63}$$

where δx is the step and λ_k is the line search parameters for iteration k.
The iteration is stopped when

$$\|J(x^k)\delta x + h(x^k)\| < \eta_k h(x^k), \tag{64}$$

where η_k is the forcing term, is satisfied.

For robustness and efficiency, it is necessary to choose λ_k and η_k properly. This is discussed in many practical venues [50, 51, 31].

2.1.2 Newton Reduced SQP

As in [55, 12, 13, 14, 11], solving the KKT system involves the following three steps:

- **Design Step** (Schur complement for middle block-row):

$$H\delta p = f, \tag{65}$$

  where H and f are the reduced Hessian and gradient, respectively:

$$\begin{aligned} H &\equiv H_{pp} - J_p^T J_u^{-T} H_{up} + \left(J_p^T J_u^{-T} H_{uu} - H_{pu}\right) J_u^{-1} J_p + \gamma R_{pp}, \text{ and} \\ f &\equiv -g_p + J_p^T J_u^{-T} g_u - \left(J_p^T J_u^{-T} H_{uu} - H_{pu}\right) J_u^{-1} h + \gamma R_p. \end{aligned} \tag{66}$$

- **State Step** (last block-row):

$$J_u \delta u = -h - J_p \delta p \tag{67}$$

- **Adjoint Step** (first block-row):

$$J_u^T \lambda_+ = -g_u - H_{uu}\delta u - H_{up}\delta p \tag{68}$$

In the literature, e.g., [69], this method is called Reduced Sequential Quadratic Programming (RSQP). By approximating the exact reduced Hessians H with a approximate reduced Hessian Q (e.g., Broyden-Fletcher-Goldfarb-Shanno (BFGS) or Davidon-Fletcher-Powell (DFP))[69, p. 197], this three-step method is called quasi-Newton Reduced SQP. It is applied in the following steps:

- **Design Step** (severe approximation to middle block-row):

$$Q\delta p = -g_p + J_p^T J_u^{-T} g_u, \tag{69}$$

where Q is a quasi-Newton approximation to the reduced Hessian.

- **State Step** (last block-row):

$$J_u \delta u = -h - J_u \delta u \tag{70}$$

- **Adjoint Step** (approximate first block-row):

$$J_u^T \lambda_+ = -g_u \tag{71}$$

We can use this sequence of operations as a preconditioner, with or without a regularization term, as in Chapter 3 and [12, 13, 14, 11].

2.2 KRYLOV METHODS

For any linear equation $Ax = b$, we can develop the iteration $x_{k+1} = x_k + (b - Ax_k)$, for $k = 0, 1, 2, \ldots$, to approximate the solution with the initial solution x_0. If we assume the initial solution as the multiple of the right hand side b, then the first approximate solution is

$$x_1 \in \text{span}\{b\}.$$

A second approximate solution can be found from the combination of b and Ab as follows:

$$x_2 \in \text{span}\{b, Ab\}.$$

Inductively, the kth approximate solution satisfies

$$x_k \in \text{span}\{b, Ab, \ldots, A^{k-1}b\}, \; k = 1, 2, \ldots$$

The space on the right of the above equation is known as a *Krylov subspace* for the matrix A and initial vector b.

The methods based on these spaces are called as *Krylov methods.* Examples are MINimal RESidual (MINRES), Conjugate Gradient (CG), Generalized MINimal RESdiual (GMRES) [74], Quasi-minimal residual (QMR) [35], etc. For some applications and explanations of these methods, see in [7, 41, 73]. For an historical overview of these Krylov methods, we may refer to Saad and van der Vorst [75]. For preconditioners and Krylov solvers, see the recent review by Benzi, Golub, and Liesen [9].

2.2.1 Preconditioned Krylov Methods

We may left precondition the systems $Ax = b$ by P, to obtain $PAx = Pb$, in which the spectral properties of their coefficient matrix PA is supposed to be more favorable than those of A, making an iterative method more efficient. If preconditioning can be done in a matrix-free manner, then we only need a subroutine that performs a preconditioner-vector product.

In this case, the preconditioned Krylov methods will approximate the kth solution in the span of a modified Krylov space

$$x_k \in \text{span}\{Pb, (PA)Pb, (PA)^2Pb, \ldots, (PA)^{k-1}Pb)\},\ k = 1, 2, \ldots$$

Borrowing from Benzi, Golub, and Liesen paper [9], the relationship of the Krylov methods, the types of A matrices, and the types of the P matrix preconditioners can be summarized as in the Table 1.

TABLE 1
Summary of Krylov subspace methods discussed in 2.2.1.

Method	*Required A*	*Type*	*Recurrence*	*Required P*
CG	symm. def.	optimal	three-term	symm. def.
MINRES, SYMMLQ	symm.	optimal	three-term	symm. def.
SQMR	symm.	non-optimal	three-term	symm.
GMRES	general	optimal	full	general
QMR, BiCGStab, TFQMR	general	non-optimal	three-term	general

2.2.2 Some Error Bounds for Krylov Methods

The spesification of Krylov methods is completed with various criteria for termination. Since we will use and compare the GMRES and the QMR methods we will discuss the error bounds for these methods.

First we will discuss the error bound for GMRES method. As in Greenbaum [41, p. 54] and Saad [73, p. 194, PROPOSITION 6.15] the error bound for the GMRES is as follows:

Assume that A is a diagonalizable matrix and let $A = V\Lambda V^{-1}$, where $\Lambda = diag\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ is the diagonal matrix of eigenvalues. Then,

$$\|r_k\| = \min_{p_k} \|p_k(A)r_0\| = \min_{p_k} \|Vp_k(\Lambda)V^{-1}r_0\| \leq \kappa(V) \min_{p_k} \|p_k(\Lambda)\| \cdot \|r_0\|, \quad (72)$$

where $\kappa(V) = \|V\| \cdot \|V^{-1}\|$ is the condition number of the eigenvector matrix V.

If A is a normal matrix (a diagonalizable matrix with a complete set of orthonormal eigenvectors, for which $\kappa(V) = 1$), then having eigenvalues tighly clustered around a single point (away from the origin) is good. Meanwhile, having the eigenvalues clustered about the origin is bad, because according to the maximum principle it is impossible to have a polynomial that is 1 at the origin and less than 1 everywhere on some closed curve around the origin.

If the matrix A is nonnormal, even having the eigenvalues tightly clustered around 1 may not tell much about the behavior of GMRES. GMRES can still have a good behavior if A has widely-distributed eigenvalues provided there are not too many distinct clusters. This can be shown by applying the GMRES method to a problem that has any nonincreasing curve representing a plot of residual norm versus iteration number. This phenomenon can be seen in the Subsection in Chapter 3 discussing about the eigenvalue spectra of the exact versus the inexact Schur complements for noisy data case.

In the GMRES method, any vector x in $x_0 + \mathcal{K}_m$ can be written as

$$x = x_0 + V_m y, \quad (73)$$

where y is an m-vector that minimizes $\|\beta e_1 - H_m y\|$, where H_m is a Hessenberg matrix obtained from incomplete orthogonalization process, and e_1 is the first column of the $n \times n$ identity matrix.

As distinct from the GMRES method, which is based on the Arnoldi algorithm [73], the QMR method is based on the Lanczos algorithm. The approximate solution x_k is of the form

$$x_k = x_0 + V_k y_k, \quad (74)$$

where y_k is chosen to minimize a quantity that is closely related to the 2-norm of the residual. The residual is

$$r_k = V_{k+1}(\beta\xi_1 - T_{k+1,k}y_k). \quad (75)$$

Hence, y_k can be found by solving the least squares problem

$$\min_y \|\beta\xi_1 - T_{k+1,k}y\|. \quad (76)$$

As in [41, p. 81, THEOREM 5.3.1] the relation of the QMR and GMRES norm is as follows.

If r_k^G is the GMRES residual at step k and r_k^Q is the QMR residual at step k, then

$$\|r_k^Q\| \leq \kappa(V_{k+1})\|r_k^G\|, \tag{77}$$

where V_{k+1} is the matrix of basis vectors for the space $\mathcal{K}_{k+1}(A, r_0)$ constructed by the Lanczos algorithm and $\kappa(\cdot)$ is the condition number.

From here, if one could create a short recurrence that could generate well-conditioned basis vectors, then the QMR would be a method of choice. Unfortunately, sometimes this is not the case. There is a problem that can make the underlying Lanczos recurrence break down. We observed this phenomenon in the Section 3.8 (Symmetric versus Nonsymmetric).

In this study, we prefer to use GMRES on our preconditioned non-symmetric KKT matrices in order to obtain robustness of solving the KKT system regardless of the indefiniteness and the nonsymmetry of the system.

2.3 SCHWARZ METHODS

2.3.1 Additive Schwarz Method

In this subsection we discuss algebraic projection methods known as *Additive* Schwarz methods. We adopt the notations as in [7, p. 77].

Consider Figure 1 which depicts the decomposition of discrete state vector for a PDE decomposed by domain. Consider that A_H is the discretization of the whole domain Ω into the coarse subdomains Ω_i with the mesh size H and A_i' is the discretization on the subdomains Ω_i' the extension of the subdomains Ω_i with the mesh size h within a distance δ, where δ is the amount of overlap. In addition, consider that $B_i = R_i^T A_i'^{-1} R_i$, $i = 1, 2, \ldots, p$, where R_i is a local restriction operator and R_i^T is a local interpolation operator from the grids of the extended subdomains Ω_i' to the grids of the subdomains Ω_i. Similarly, R_H and R_H^T are a global restriction operator and a global interpolation operator respectively from the coarse grids with mesh size H to the fine grids with mesh size h. Assuming that we want to solve a system of linear equations $Au = f$, the *Additive Schwarz* preconditioner can be described as:

$$u = \left(R_H^T A_H^{-1} R_H + \sum_{i=1}^{p} B_i \right) r, \tag{78}$$

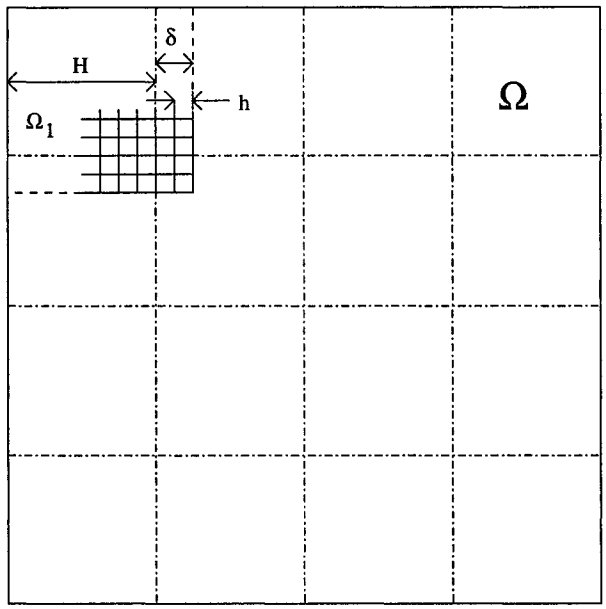

FIG. 1. *Domain decomposition for matching grid Schwarz methods.*

where r is the residual.

This is known as *two level additive Schwarz.* If we drop the first term, it is a *one level additive Schwarz.* One variant of these methods is *restricted additive Schwarz* as the default Schwarz method in software PETSc from Argonne National Laboratory [5]. We can see more details about this variant in [19].

As discussed in [55], if we apply these methods as preconditioners of the Krylov methods we can summarize in Table 2 for the forward elliptic problem in terms of theoretical condition numbers and iteration counts.. Here, for convergence scalability estimates, we assume that one subdomain per processor in a d-dimensional isotropic problem, where $N = h^{-d}$ is the problem size and $P = H^{-d}$ is the number of processors in isotropic decomposition using all spatial dimensions. This estimate is for self-adjoint positive-definite elliptic problems. It does not apply to KKT conditions. See [78] for more details about the convergence behavior of these methods.

2.4 SCHUR METHODS

In this section, we consider nonoverlapping domain decomposition methods known as *substructuring* or *Schur complement* methods [7, p.78].

TABLE 2
Summary of the Schwarz methods discussed in 2.3.

Preconditioning	$\kappa(B^{-1}A)$	$2D$ Iter.	$3D$ Iter.
Point Jacobi	$\mathcal{O}(h^{-2})$	$\mathcal{O}(N^{1/2})$	$\mathcal{O}(N^{1/3})$
Domain Jacobi	$\mathcal{O}((hH)^{-1})$	$\mathcal{O}((NP^{1/4})$	$\mathcal{O}((NP)^{1/6})$
1-level Additive Schwarz	$\mathcal{O}(H^{-2})$	$\mathcal{O}(P^{1/2})$	$\mathcal{O}(P^{1/3})$
2-level Additive Schwarz	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$

For simplicity of exposition, let us consider 2-D problems. Define I as the set of all interior points in all of the subdomains Ω_i and B as the set of all points on the boundaries of the subdomains $\partial\Omega_i$. In order to solve the system $Au = f$, we reorder u and f as $u \equiv (u_I, u_B)^T$ and $f \equiv (f_I, f_B)^T$ according to the partition. Hence, the system can be rewritten as:

$$\begin{pmatrix} A_I & A_{IB} \\ A_{IB}^T & A_B \end{pmatrix} \begin{pmatrix} u_I \\ u_B \end{pmatrix} = \begin{pmatrix} f_I \\ f_B \end{pmatrix} \tag{79}$$

This system can be rewritten again as a block LU-factorization as follows:

$$\begin{pmatrix} I & 0 \\ A_{IB}^T A_I^{-1} & I \end{pmatrix} \begin{pmatrix} A_I & A_{IB} \\ 0 & S_B \end{pmatrix} \begin{pmatrix} u_I \\ u_B \end{pmatrix} = \begin{pmatrix} f_I \\ f_B \end{pmatrix}, \tag{80}$$

where

$$S_B \equiv A_B - A_{IB}^T A_I^{-1} A_{IB}$$

is the Schur complement of A_B in A. Hence, by eliminating u_I, we have the equation

$$S_B u_B = g_B \equiv f_B - A_{IB} A_I^{-1} f_I. \tag{81}$$

To solve the overall system, we solve the Schur complement system first, then substitute back in to the overall system.

2.5 JACOBIAN MATRIX-FREE NEWTON-KRYLOV

Let $h(x) = 0$ be a system of nonlinear equations arising from a discretized PDE. To estimate the Jacobian-vector product with an arbitrary vector v at the point x, we

use a matrix-free (Fréchet) derivative as follows:

$$J(x)v \equiv \lim_{\epsilon \to 0} \frac{h(x+\epsilon v) - h(x)}{\epsilon}. \tag{82}$$

or

$$Jv \approx \frac{h(x+\epsilon v) - h(x)}{\epsilon}, \tag{83}$$

where ϵ is a suitably chosen small perturbation.

For an initial guess, δx_0, an initial residual, r_0, is defined as:

$$r_0 = -h(x) - J\delta x_0. \tag{84}$$

Hence, iteratively, the ith residual can be computed as:

$$r_i = -h(x) - J\delta x_i, \tag{85}$$

where δx_i is the Newton step (correction).

Then, to compute the correction δx_i, we can use a GMRES method as follows:

$$\delta x_i = \delta x_0 + \sum_{j=0}^{i-1} \beta_j (J)^j r_0, \tag{86}$$

where the scalars β_j are chosen in the standard way to minimize the residual [74].

From here, we see that GMRES only requires the action of the Jacobian in the form of matrix-vector products, which may be approximated by matrix-free (Fréchet) derivative.

For some examples of using Jacobian-free methods in solving PDEs, see Knoll and Keyes [57], and Kelley [51, p. 57].

2.6 HESSIAN MATRIX-FREE LAGRANGE-NEWTON-KRYLOV-SCHUR-SCHWARZ METHOD

In one of their preconditioners, Biros and Ghattas [12, 13, 14, 11] use an approximate state Jacobian. In this work, we use an exact state Jacobian generated by ADIC, an automatic differentiation toolkit in C from Argonne National Laboratory [46], and ADMAT (automatic differentiation for MATLAB [64]) from A. Verma, Cornell University [27, 28]. For the Lagrangian reduced Hessian, we use an identity matrix, probing techniques, and other approximations that are spectrally equivalent. Meanwhile, Biros and Ghattas used limited memory BFGS. Our design Jacobian is also provided by ADIC.

To solve the full system, we use a Newton framework with Hessian matrix-free manner on the KKT conditions, then we solve the linearized systems of equations by using Krylov methods.

As we mention in [55], for building the preconditioners on the outer levels we solve the three steps of the KKT systems as Schur complements, in the inner levels we use Schwarz methods in solving each equation for each step. Thus, we may describe our methods as *Hessian Matrix-free Lagrange-Newton-Krylov-Schur-Schwarz* methods.

2.7 GLOBALIZATION AND ROBUSTIFICATION

Newton methods usually converge locally. In order to get global solutions we may use trust region or line search methods. In this study we choose basic line search methods, applied to inexact Newton methods, which are shown to be robust enough. For the detail of inexact Newton methods with line search, see [50, p. 135].

CHAPTER 3

PRECONDITIONING STRATEGIES

Preconditioning strategies are very important for iteratively solving the systems of linear equations that arise in large-scale optimization. These systems of equations may be indefinite and/or ill-conditioned. Without good preconditioners, the iterative processes converge very slowly or not at all.

As mentioned earlier, our preconditioners consist of inner and outer preconditioners, where the inner preconditioner is of Schwarz type, and the outer preconditioner is of Schur type. In this chapter we discuss some preconditioning strategies, especially some approximations for the Schur complements. These approximations – probing, the J operator, and the Laplace operator – are known in solving PDEs. For cases in what the exact data is furnished, as in one of our numerical test cases, the exact Schur complements are superior to the approximations. Interestingly, for noisy data there are no preconditioners that are superior in all aspects of computations among the preconditioners, including the exact ones. In some cases the exact Schur complement preconditioner is worse than the inexact. Hence, we choose the least expensive approximation that provides acceptable results. This is important in practice, because usually we have noisy data.

In this study we also describe two kinds of preconditioners, linear and nonlinear. Our numerical experiments (e.g. 1-D Numerical Example 2) show that nonlinear preconditioners can be superior to the linear, even for the case of noisy data. We describe also two formulations of KKT matrices, symmetric and nonsymmetric. Our numerical experiments (e.g. 2-D Numerical Example 1) show that although the nonsymmetric matrices may result complex eigenvalues, when using the Krylov method GMRES, these preconditioned nonsymmetric matrices can make the iterative process converge about 50% faster than the symmetric ones, since the preconditioned KKT matrix is clustered away from the origin and only mildly indefinite. We examine this phenomenon from the convergence history of the residual norms and the spectrum of the preconditioned KKT matrices, as in Figure 16.

3.1 KKT MATRIX ANALYSIS, INDEFINITENESS, AND SOME MATRIX THEORY

3.1.1 KKT Matrix Analysis

As mentioned earlier, the KKT systems of the full-space approaches may have matrices that are indefinite and ill-conditioned. Hence, good preconditioners are required. The preconditioners that we consider in this work are Schur complement preconditioners with respect to the design variables (parameters).

First, we will discuss the case of symmetric KKT matrices. This is generally the formulation of choice for the PDE-constrained community. Biros and Ghattas in [12, 13, 14, 11] proposed a preconditioner that is a block-factorization of LU form. Haber and Ascher [42] also use this kind of preconditioner. Biros and Ghattas approximate the Schur complement by using Limited-memory BFGS, while Haber and Ascher use Gauss-Newton methods. Another preconditioner that is proposed by Battermann and Sachs [8] consist of dropping all the second derivatives except the second derivative in the block $(2,2)$, in our notation below.

To more fully describe this approach, we consider the symmetric KKT matrix of Equation (55). If the block $(1,1)$ is positive definite, then the symmetric KKT systems of the forms (55) may have positive, negative, and zero eigenvalues. From (55) we can find the Schur complement of the matrix by following 2.1.2. Hence, the KKT system can be written as follows:

$$\begin{bmatrix} H_{uu} & H_{up} & J_u^T \\ 0 & S & 0 \\ J_u & J_p & 0 \end{bmatrix} \begin{pmatrix} \delta u \\ \delta p \\ \lambda_+ \end{pmatrix} = - \begin{pmatrix} g_u \\ g_S \\ h \end{pmatrix}, \tag{87}$$

where S and g_S are the reduced Hessian (the Schur complement) and gradient, respectively:

$$S \equiv H_{pp} - J_p^T J_u^{-T} H_{up} + \left(J_p^T J_u^{-T} H_{uu} - H_{pu} \right) J_u^{-1} J_p + \gamma R_{pp} \tag{88}$$

and

$$g_S \equiv -g_p + J_p^T J_u^{-T} g_u - \left(J_p^T J_u^{-T} H_{uu} - H_{pu} \right) J_u^{-1} h + \gamma R_p. \tag{89}$$

In order to use this matrix as a preconditioner and to get a symmetric preconditioned KKT matrix we have to use the whole matrix by factoring into an LU form. Hence, we will have the LU matrix form as follows:

$$\begin{bmatrix} H_{uu}J_u^{-1} & 0 & I \\ H_{pu}J_u^{-1} & I & J_p^T J_u^{-T} \\ I & 0 & 0 \end{bmatrix} \begin{bmatrix} J_u & J_p & 0 \\ 0 & S & 0 \\ 0 & H_{up} - H_{uu}J_u^{-1}J_p & J_u^T \end{bmatrix} \tag{90}$$

As mentioned earlier, this preconditioner was used by Haber and Ascher [42], Biros and Ghattas [12, 13, 14, 11], and also Burger and Muhlhuber [16, 17]. Meanwhile, Battermann and Sachs [8] used

$$\begin{bmatrix} 0 & 0 & J_u^T \\ 0 & S & J_p^T \\ J_u & J_p & 0 \end{bmatrix} \tag{91}$$

In this dissertation we focus our attention on the nonsymmetric form of KKT systems, since the nonsymmetric cases are experimentally competitive to the symmetric ones and less explored in the literature. Thus, we may rewrite KKT systems in the following form:

$$\begin{bmatrix} J_u^T & H_{uu} & H_{up} \\ 0 & J_u & J_p \\ J_p^T & H_{pu} & H_{pp} + \gamma R_{pp} \end{bmatrix} \begin{pmatrix} \lambda_+ \\ \delta u \\ \delta p \end{pmatrix} = - \begin{pmatrix} g_u \\ h \\ g_p + \gamma R_p \end{pmatrix}, \tag{92}$$

where $H_{ab} \equiv \frac{\partial^2 c}{\partial a \partial b} + \lambda^T \frac{\partial^2 h}{\partial a \partial b}$, $J_a \equiv \frac{\partial h}{\partial a}$, and $g_a = \frac{\partial c}{\partial a}$, for $a, b \in \{p, u\}$, and where $\lambda_+ = \lambda + \delta\lambda$.

We find the Schur complement by eliminating the (3,1) and (3,2) blocks with the first block row and the second block row. Thus, we have

$$\begin{bmatrix} J_u^T & H_{uu} & H_{up} \\ 0 & J_u & J_p \\ 0 & 0 & S \end{bmatrix} \begin{pmatrix} \lambda_+ \\ \delta u \\ \delta p \end{pmatrix} = - \begin{pmatrix} g_u \\ h \\ g_S \end{pmatrix}, \tag{93}$$

where S and g_S are the reduced Hessian (the Schur complement) and gradient, respectively:

$$S \equiv H_{pp} + \gamma R_{pp} - (J_u^{-1}J_p)^T H_{up} - H_{pu}J_u^{-1}J_p + (J_u^{-1}J_p)^T H_{uu}(J_u^{-1}J_p)$$

and

$$g_S \equiv g_p + \gamma R_p - (J_u^{-1}J_p)^T g_u - H_{pu}J_u^{-1}h + (J_u^{-1}J_p)^T H_{uu}J_u^{-1}h.$$

The LU factors of the Hessian H are

$$L = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ J_p^T J_u^{-T} & H_{pu} J_u^{-1} - J_p^T J_u^{-T} H_{uu} J_u^{-1} & I \end{bmatrix} \tag{94}$$

and

$$U = \begin{bmatrix} J_u^T & H_{uu} & H_{up} \\ 0 & J_u & J_p \\ 0 & 0 & S \end{bmatrix}. \tag{95}$$

The inverse of the Hessian H is $H^{-1} = U^{-1}L^{-1}$, as follows:

$$U^{-1} = \begin{bmatrix} J_u^{-T} & -J_u^{-T} H_{uu} J_u & -J_u^{-T}(H_{up} - H_{uu} J_u^{-1} J_p) S^{-1} \\ 0 & J_u^{-1} & -J_u^{-1} J_p S^{-1} \\ 0 & 0 & S^{-1} \end{bmatrix} \tag{96}$$

and

$$L^{-1} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -J_p^T J_u^{-T} & -(H_{pu} - J_p^T J_u^{-T} H_{uu}) J_u^{-1} & I \end{bmatrix}. \tag{97}$$

3.1.2 Indefiniteness and Some Matrix Theory

KKT systems are classic saddle point problems. In practice, whether symmetric or not, the KKT matrix is generally indefinite (i.e., it has eigenvalues with both positive and negative real parts). For a comprehensive overview of these kinds of problems, including choosing the linear solvers and preconditioners, see the recent review by Benzi, Golub and Liesen [9] and the references therein.

To overcome the indefiniteness of a nonsymmetric matrix, Murphy, Golub and Wathen [66] noted the following Proposition:

Proposition 3.1.2.1 (Murphy, Golub, Wathen, 2000) *If*

$$\mathcal{A} = \begin{bmatrix} A & B^T \\ C & 0 \end{bmatrix} \tag{98}$$

is preconditioned by

$$\mathcal{P} = \begin{bmatrix} A & 0 \\ 0 & CA^{-1}B^T \end{bmatrix}, \tag{99}$$

then the preconditioned matrix $\mathcal{T} = \mathcal{P}^{-1}\mathcal{A}$ *satisfies*

$$\mathcal{T}(\mathcal{T} - \mathcal{I})(\mathcal{T}^2 - \mathcal{T} - \mathcal{I}) = 0. \tag{100}$$

From (100) this preconditioner has an attractive property, that is, the preconditioned matrix $\mathcal{T} = \mathcal{P}^{-1}\mathcal{A}$ has at most four distinct eigenvalues: $0, 1, \frac{1}{2} \pm \frac{\sqrt{5}}{2}$. Thus Krylov methods will converge within four iterations. For the same block preconditioner if $C = B$ and $S = BA^{-1}B^T$, we will have three distinct eigenvalues: $1, \frac{1}{2} \pm \frac{\sqrt{5}}{2}$. Hence, from this theoretical point of view we can see that the preconditioned symmetric KKT matrix is preferable than the nonsymmetric one. However, in practice this is not always the case, as further discussed.

The $(1, 1)$ block of the matrix $\mathcal{A}$ may be singular. Hence some modification can be made by changing the matrix and the preconditioner from A to $A + \gamma BB^T$. This is suggested by Golub and Greif [39]. They also mentioned that the construction of the preconditioner is costly for large problems.

If we assume that the $(1, 1)$ block is positive definite, another approach to overcome indefiniteness is by changing the sign of the $(2, 1)$ block. However, this method can not be applied if the $(1, 1)$ block is indefinite. To overcome this, changing from A to $A + \gamma BB^T$ can also be tried.

Unfortunately, as observed in our numerical test cases, the matrix or block B can be nearly singular. The block B consists of the Jacobians with respect to the state variable and the parameter, where the Jacobian with respect to the parameter is singular. Hence, if A is singular, then $A + \gamma BB^T$ can be singular as well. Thus, we can not rely on the approaches that are proposed by Golub *et al.* above.

More recently for block preconditioning of general nonsymmetric matrices, de Sturler and Liesen [79] have proposed a splitting of the $(1, 1)$ block of the matrix, which results more favorable spectrum that causes faster convergence for Krylov subspace methods.

3.2 PROPOSED LINEAR SCHUR-SCHWARZ PRECONDITIONER

Our proposed linear preconditioners are intended for solving large-scale problems. As in Equation (55) our KKT matrix is a 3×3 block matrix. Thus, we may construct the preconditioners based on smaller size blocks in the matrices compared to Murphy, Golub and Wathen [66].

We construct our proposed nonsymmetric preconditioner by following Eq. (92). We built our first preconditioner (PC1) as in Eq. (93). Hence, the preconditioned matrix is as follows:

$$\begin{bmatrix} J_u^T & H_{uu} & H_{up} \\ 0 & J_u & J_p \\ J_p & H_{pu} & H_{pp}+\gamma R_{pp} \end{bmatrix} \begin{bmatrix} J_u^{-T} & -J_u^{-T}H_{uu}J_u & -J_u^{-T}(H_{up}-H_{uu}J_u^{-1}J_p)S^{-1} \\ 0 & J_u^{-1} & -J_u^{-1}J_pS^{-1} \\ 0 & 0 & S^{-1} \end{bmatrix},$$

which is equal to

$$\begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ P & Q & I \end{bmatrix}, \tag{101}$$

where $P = J_p^T J_u^{-T}$, and $Q = -J_p^T J_u^{-T} H_{uu} J_u^{-1} + H_{pu} J_u^{-1}$.

We can see that the spectrum of the preconditioned matrix is clustered.

Our first proposed preconditioner leads to the following algorithm:

Algorithm of Triangular Preconditioner

Write the vector resulting from application of the preconditioner as $v = [v_\lambda, v_u, v_p]$. *Applying a left preconditioner our preconditioning algorithm can be executed as follows, where* t_1, t_2, *and* t_3 *denote temporaries*

$$\begin{aligned}
t_1 &= J_u^{-1} * h \\
t_2 &= H_{pu} * t_1 \\
t_3 &= J_p^T * ((J_u^T)^{-1} * (H_{uu} * t_1)) \\
g_S &= -g_p - \gamma R_p + J_p^T * ((J_u^T)^{-1} * g_u) - t_3 + t_2 \\
t_1 &= J_u^{-1} * J_p \\
t_2 &= H_{pu} * t_1 \\
t_3 &= J_p^T * ((J_u^T)^{-1} * (H_{uu} * t_1)) \\
S &= H_{pp} + \gamma R_{pp} - J_p^T * ((J_u^T)^{-1} * H_{pu}) + t_3 - t_2 \\
v_p &= S^{-1} * g_S \\
v_u &= J_u^{-1} * (-h - J_p * v_p) \\
v_\lambda &= (J_u^T)^{-1} * (-g_u - H_{uu} * v_u - H_{up} * v_p)
\end{aligned}$$

We find the inverses inside the algorithm by solving the related linear equations with Schwarz methods.

From (93) if we drop the $(1,3)$ and $(2,3)$ blocks, we get our second preconditioner

(PC2), and the preconditioned matrix is follows:

$$\begin{bmatrix} J_u^T & H_{uu} & H_{up} \\ 0 & J_u & J_p \\ J_p & H_{pu} & H_{pp}+\gamma R_{pp} \end{bmatrix} \begin{bmatrix} J_u^{-T} & -J_u^{-T}H_{uu}J_u & 0 \\ 0 & J_u^{-1} & 0 \\ 0 & 0 & S^{-1} \end{bmatrix},$$

which is equal to

$$\begin{bmatrix} I & 0 & H_{up}S^{-1} \\ 0 & I & J_pS^{-1} \\ P & Q & (H_{pp}+\gamma R_{pp})S^{-1} \end{bmatrix}, \tag{102}$$

where $P = J_p^T J_u^{-T}$, and $Q = -J_p^T J_u^{-T} H_{uu} J_u^{-1} + H_{pu} J_u^{-1}$.

The second preconditioner is a result of dropping the blocks (1, 3) and (2,3), which is the same as dropping the terms H_{up} and J_p. Therefore, for applying the second preconditioner we simply drop all the related terms from the first algorithm above.

We apply our second proposed preconditioner as in the following algorithm:

Algorithm of Reduced Triangular Preconditioner

Write the vector resulting from application of the preconditioner as $v = [v_\lambda, v_u, v_p]$. *Applying a left preconditioner our preconditioning algorithm can be executed as follows, where* t_1, t_2, *and* t_3 *denote temporaries*

$$\begin{aligned}
t_1 &= J_u^{-1} * h \\
t_2 &= H_{pu} * t_1 \\
t_3 &= J_p^T * ((J_u^T)^{-1} * (H_{uu} * t_1)) \\
g_S &= -g_p - \gamma R_p + J_p^T * ((J_u^T)^{-1} * g_u) + t_2 - t_3 \\
t_1 &= J_u^{-1} * J_p \\
t_2 &= H_{pu} * t_1 \\
t_3 &= J_p^T * ((J_u^T)^{-1} * (H_{uu} * t_1)) \\
S &= H_{pp} + \gamma R_{pp} - J_p^T * ((J_u^T)^{-1} * H_{pu}) - t_2 + t_3 \\
v_p &= S^{-1} * g_S \\
v_u &= J_u^{-1} * (-h) \\
v_\lambda &= (J_u^T)^{-1} * (-g_u - H_{uu} * v_u)
\end{aligned}$$

We find the inverses inside the algorithm by solving the related linear equations with Schwarz methods.

If from (93) we drop further the (1, 2) block, we get our third preconditioner

(PC3), and the preconditioned matrix will be as follows:

$$\begin{bmatrix} J_u^T & H_{uu} & H_{up} \\ 0 & J_u & J_p \\ J_p & H_{pu} & H_{pp}+\gamma R_{pp} \end{bmatrix} \begin{bmatrix} J_u^{-T} & 0 & 0 \\ 0 & J_u^{-1} & 0 \\ 0 & 0 & S^{-1} \end{bmatrix},$$

which is equal to

$$\begin{bmatrix} I & H_{uu}J_u^{-1} & H_{up}S^{-1} \\ 0 & I & J_pS^{-1} \\ J_p^TJ_u^{-T} & H_{pu}J_u^{-1} & (H_{pp}+\gamma R_{pp})S^{-1} \end{bmatrix}. \tag{103}$$

We apply this preconditioner as follows:

Algorithm of Block Diagonal Preconditioner

Write the vector resulting from application of the preconditioner as $v = [v_\lambda, v_u, v_p]$. *Applying a left preconditioner our preconditioning algorithm can be executed as follows, where* t_1, t_2, *and* t_3 *denote temporaries*

$$\begin{aligned} t_1 &= J_u^{-1} * h \\ t_2 &= H_{pu} * t_1 \\ t_3 &= J_p^T * ((J_u^T)^{-1} * (H_{uu} * t_1)) \\ g_S &= -g_p - \gamma R_p + J_p^T * ((J_u^T)^{-1} * g_u) + t_2 - t_3 \\ t_1 &= J_u^{-1} * J_p \\ t_2 &= H_{pu} * t_1 \\ t_3 &= J_p^T * ((J_u^T)^{-1} * (H_{uu} * t_1)) \\ S &= H_{pp} + \gamma R_{pp} - J_p^T * ((J_u^T)^{-1} * H_{pu}) - t_2 + t_3 \\ v_p &= S^{-1} * g_S \\ v_u &= J_u^{-1} * (-h) \\ v_\lambda &= (J_u^T)^{-1} * (-g_u) \end{aligned}$$

We find the inverses inside the algorithm by solving the related linear equations with Schwarz methods.

Our numerical experiments show that the eigenvalues of the preconditioned matrices using these three proposed preconditioners are clustered or well-distributed. Hence, for acceleration by using Krylov methods, such as GMRES, these are efficient linear preconditioners.

3.3 PROPOSED NONLINEAR PRECONDITIONER

The preconditioners of this section are used to precondition the PDE function $h(x) = 0$ before we set up the Lagrangian and solve the overall system, so that if we use Newton methods, we will have more stable linear systems. See [51, p. 30] and [18]. In the elliptic inverse problems that we study in this dissertation, we precondition the elliptic PDE $h(x) = 0$ by left multiplying with inverse Laplacian with homogeneous Dirichlet boundaries before we solve the problems by means of Lagrangian methods. Hence, we have more stable KKT systems.

The first-order optimality conditions will be as follows:

$$\frac{\partial \mathcal{L}}{\partial u} \equiv \frac{\partial c}{\partial u} + \lambda^T \frac{\partial \triangle^{-1} h}{\partial u} = 0, \tag{104}$$

$$\frac{\partial \mathcal{L}}{\partial p} \equiv \frac{\partial c}{\partial p} + \lambda^T \frac{\partial \triangle^{-1} h}{\partial p} = 0, \tag{105}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} \equiv \triangle^{-1} h = 0. \tag{106}$$

In this work we demonstrate by a numerical example the effectiveness of the nonlinear preconditioners compared to the linear preconditioners by implementing them on a nonlinear elliptic PDE.

3.4 NUMERICAL EXAMPLES

In this study we implement our method for one-dimensional cases in the framework of MATLAB [64] and ADMAT [27, 28] on the Sun workstation of the Office of Computing and Communications Services (OCCS) at Old Dominion University. Meanwhile, for the two-dimensional cases we implement the method in the framework of PETSc [5] and ADIC [46] on the SUN Enterprise 10000 Starfire (E10K) with 64 processors and Linux Cluster with 32 processors at OCCS. In one-dimensional cases, our Hessians and Jacobians are computed by using ADMAT. Meanwhile, in the two-dimensional cases our Jacobians are computed by using ADIC. For efficient computation of Jacobians and Hessians, see [26, 25].

3.4.1 Numerical Example 1 (1-D)

The problem is to recover the linear diffusivity function the parameter from the flow measurement in a PDE model equation. These types of linear elliptic PDEs can be found in the groundwater modeling problems.

In this numerical example we solve an inverse problem for a PDE of the form:

$$\nabla \cdot (\beta(x)\nabla u(x)) = f(x), \tag{107}$$

on $0 < x < 1$, where

$$\beta(x) = 1 - 0.25e^{-\frac{(x-0.45)^2}{0.01}} \tag{108}$$

is the parameter, $u(x)$ is the flow, and $f(x) = \delta(\frac{1}{3}) = 1$ is the source/sink, then subject to boundary conditions $u(0) = 0.0$ and $u(1) = 0.0$.

This example is similar to one in Vogel's book [86, p. 90]. However, Vogel uses two measurements. We use only one measurement.

3.4.2 Numerical Example 2 (1-D)

The problem is to recover the nonlinear diffusivity constant as the parameter from the flow measurement in PDE model equation. These kinds of PDEs can be found in the radiation transport problems, where the PDEs are nonlinear elliptic PDEs.

In this numerical example we solve an inverse problem for a PDE of the form:

$$\nabla \cdot (\beta(x)u(x)^{\alpha}\nabla u(x)) = f(x), \tag{109}$$

on $0 < x < 1$, where $\beta(x)$ and α are the parameters, $u(x)$ is the flow, and $f(x) = 0.001$ is the source distributed in the whole domain, subject to boundary conditions $u(0) = 0.1$ and $u(1) = 1.0$.

In this example we choose

$$\beta(x) = \begin{cases} 1 & \text{if } x <= 0.5 \\ 10 & \text{if } x > 0.5 \end{cases} \tag{110}$$

and $\alpha = 2.5$.

3.5 NUMERICAL EXPERIMENTS FOR LINEAR PRECONDITIONERS

In this section we will do some experiments for the linear preconditioners. We compare the preconditioners with respect to the real or exact data and the noisy data, and the exact and inexact Schur complements. We use the two numerical examples above.

3.5.1 Numerical Experiments with Example 1

Exact Data

In this example we assume heat conduction in metal rod. We place a heat source at one-third of the rod, then measure the temperature on certain points of the rod. The inverse problem is to recover the conductivity constant of the rod.

In Figure 2 we show the spectra of the preconditioned KKT matrices with exact Schur complements for the three preconditioners.

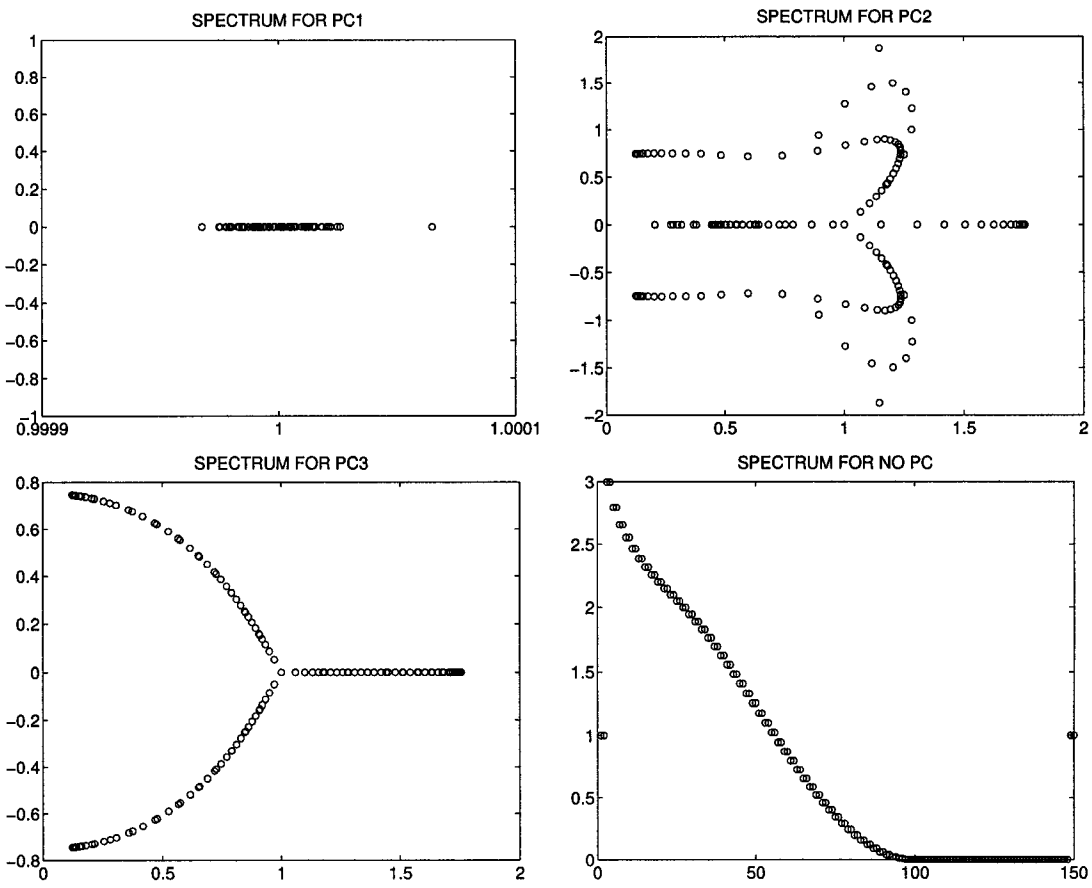

FIG. 2. *Spectrum of the KKT matrices preconditioned by the three linear preconditioners with the exact Schur complements on the complex plane (axes label by real and imaginary parts) (top row and bottom left) and unpreconditioned on the real line (real values plotted against index) (bottom right) with* 0% *noise.*

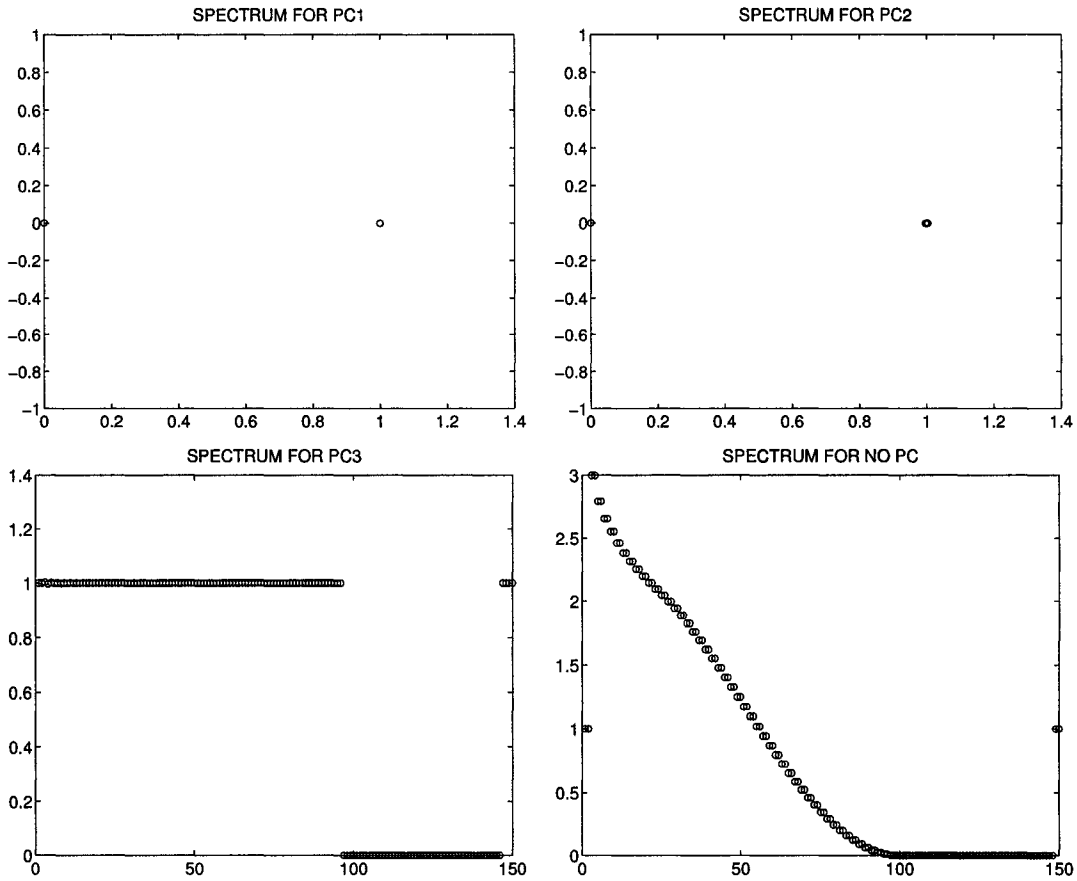

FIG. 3. *Spectrum of the KKT matrices preconditioned by the three linear preconditioners with the inexact Schur complements approximate by identity matrix on the complex plane (axes label by real and imaginary parts) (top row) and on the real line (real values plotted against index) (bottom left), and unpreconditioned on the real line (bottom right) with* 0% *noise.*

In Figure 3 the inexact Schur complements are approximated by identity matrices. We see that all the spectra, where the regularization parameters are equal to $\gamma = 1.0 \times 10^{-8}$, are more clustered compared to the unpreconditioned case as we can

see in the bottom right of the Figure 2 and Figure 3. We will discuss the effects of the regularization parameters on the convergence histories, the spectra, and the matching between the computed or recovered parameters and the exact data in the next chapter.

We can see the convergence histories of the residual norms of the right-hand sides (gradients) of the preconditioned KKT systems with the exact Schur complements in the top row (with PC1 (left) and PC2 (right)) and the left bottom (with PC3) figures in Figure 4.

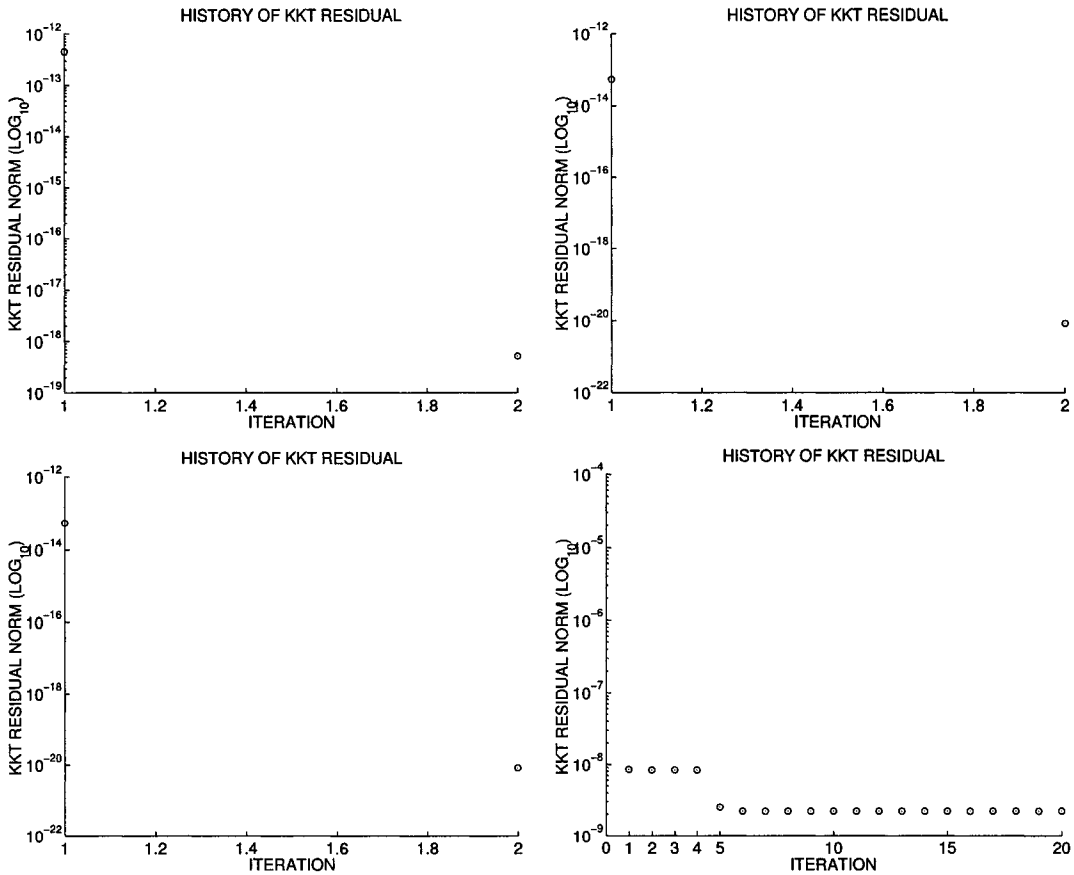

FIG. 4. *Convergence history of the residuals (the right-hand side of the KKT systems preconditioned by the three linear preconditioners with the exact Schur complements) (top left (PC1), top right (PC2), and bottom left (PC3)) and unpreconditioned (bottom right) with* 0% *noise.*

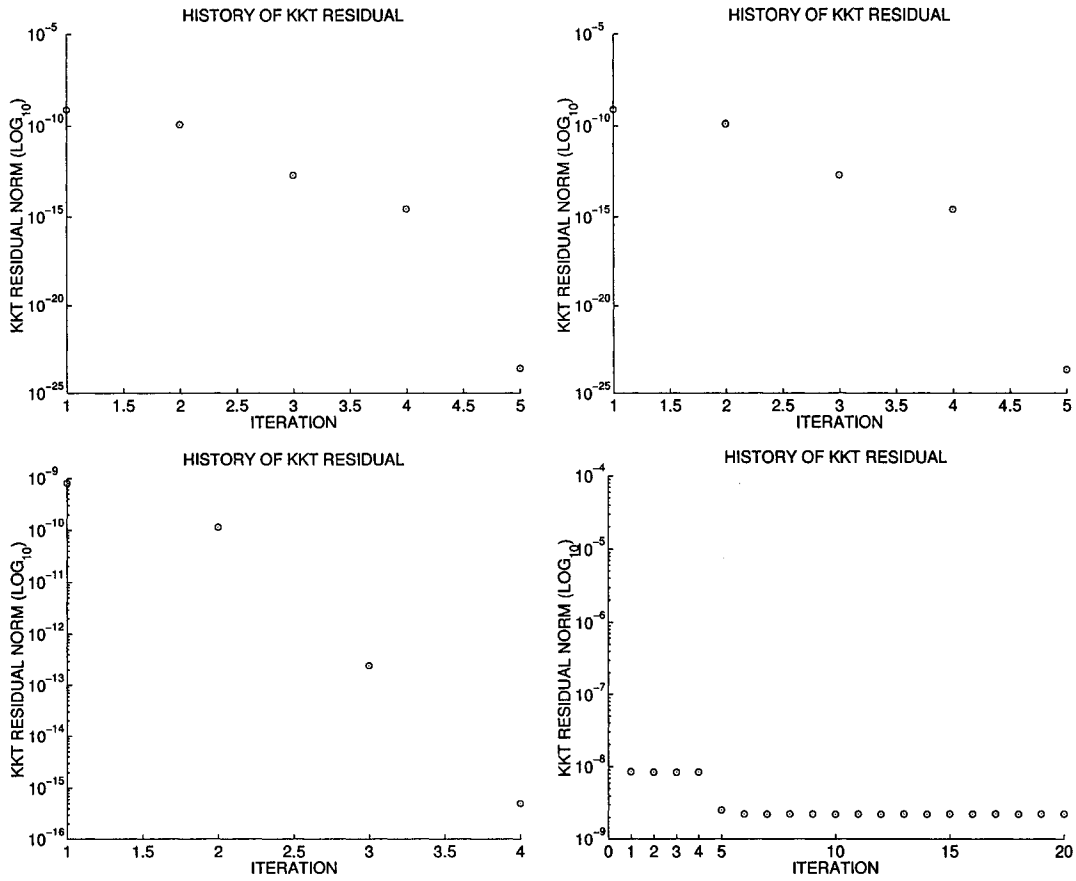

FIG. 5. *Convergence history of the residuals (the right-hand side of the KKT systems preconditioned by the three linear preconditioners with the inexact Schur complements approximate by identity matrices) (top left (PC1), top right (PC2), and bottom left (PC3)) and unpreconditioned (bottom right) with* 0% *noise.*

In addition, we can see the convergence histories of the residual norms of the right-hand sides of the preconditioned KKT systems with the inexact Schur complements in the top row (with PC1 (left) and PC2 (right)) and the left bottom (with PC3) figures in (Figure 5). Here, the inexact Schur complements are approximated by identity

matrices. The histories of the exact Schur complements show faster convergence than the inexact ones. Meanwhile, the convergence history of the unpreconditioned one is stagnant after the third iteration, as we can see in the bottom right of Figure 4 and Figure 5.

Similarly, we show the matching between the computed parameters and the exact parameter data for the exact Schur complements as the top row (with PC1 (left) and PC2 (right)) and the left bottom (with PC3) figures in Figure 6.

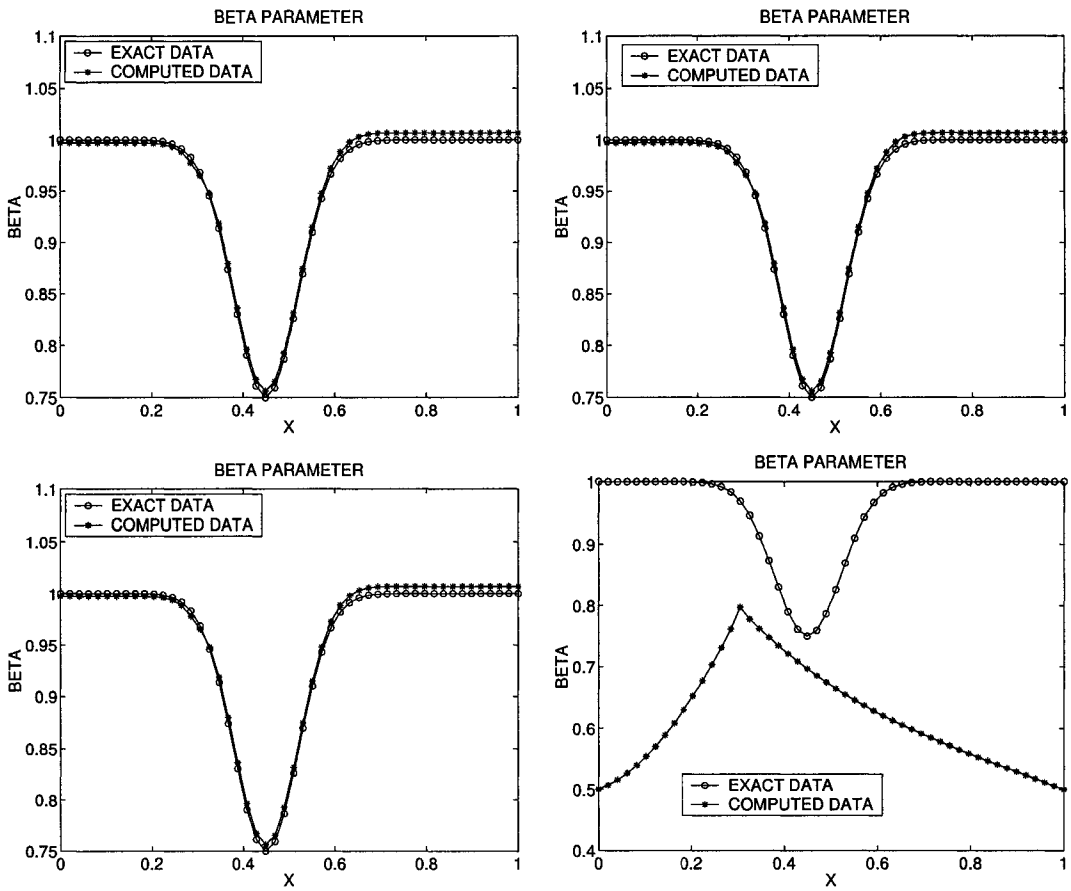

FIG. 6. *Reconstructed parameters of the systems preconditioned by the three linear preconditioners with the exact Schur complements (top left (PC1), top right (PC2), and bottom left (PC3)) and unpreconditioned (bottom right) with* 0% *noise.*

For comparison, we show the matching between the computed parameters and the exact parameter data for the inexact Schur complements as the top row (with PC1 (left) and PC2 (right)) and the left bottom (with PC3) figures in Figure 7. The differences are invisible. However, for the unpreconditioned case, as in the bottom right figures of Figure 6 and Figure 7, we can see the computed parameter does not match with the exact parameter data. Hence, without a preconditioner the inverse problem does not converge.

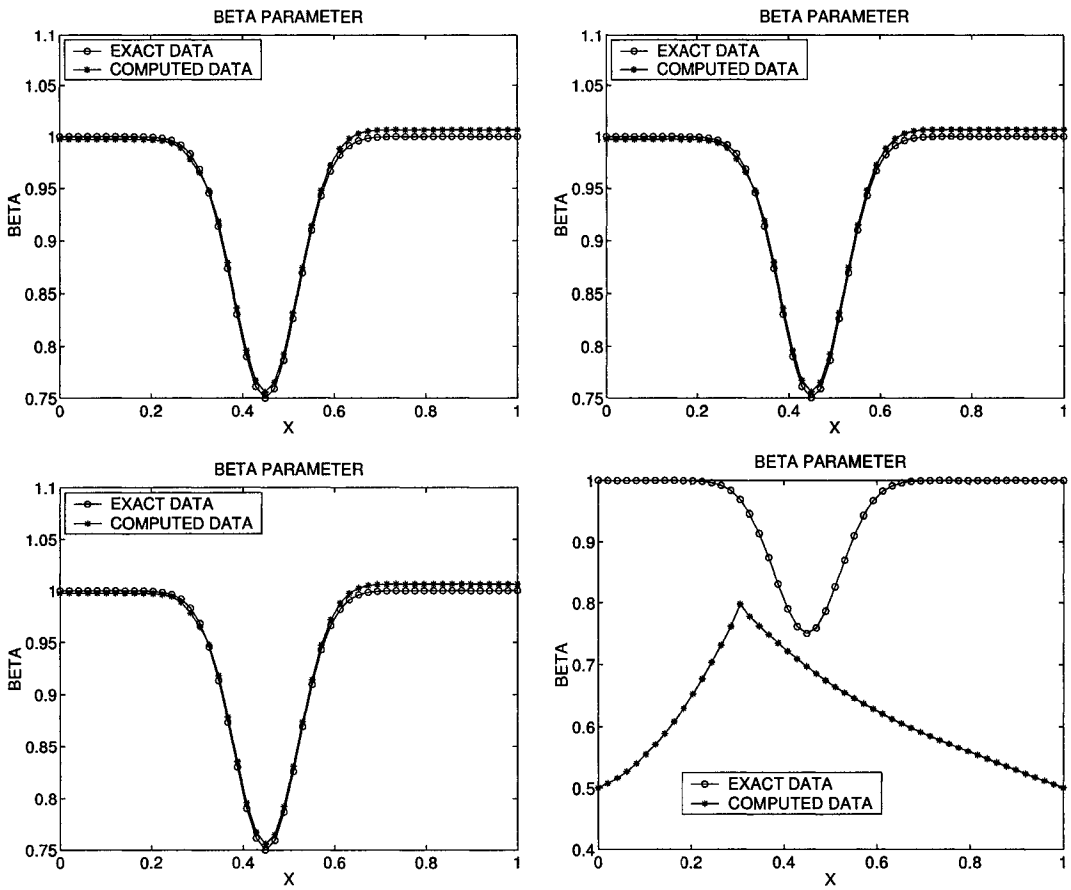

FIG. 7. *Reconstructed parameters of the systems preconditioned by the three linear preconditioners with the inexact Schur complements approximate by identity matrices (top left (PC1), top right (PC2), and bottom left (PC3)) and unpreconditioned (bottom right) with* 0% *noise.*

Some Comparisons for the approximations of the Schur complements

Here, we compare some different approximations for the Schur complements of the third type preconditioner. We approximate the Schur complements with identity matrix, probing, BFGS, the Laplacian, and the J operator.

Probing

What we mean by probing here is that we approximate the unknown tridiagonal part of the Schur complement by tridiagonal matrix. This method is less expensive then forming the exact Schur complement. As in Smith *et al.* [78] the algorithm can be explained as follows:

Algorithm for Probing method:

Assume that S is the Schur complement and approximately equal to $\hat{S}$, where

$$\hat{S} = \begin{bmatrix} a_1 & b_1 & & \\ b_1 & a_2 & b_2 & \\ & b_2 & a_3 & b_3 \\ & & \cdots & \end{bmatrix}. \tag{111}$$

To construct the elements of the probing matrix we use the following steps:

1. *Construct $u^1 = (1, ..., 1)^T$ and $u^2 = (-1, 1, -1, 1, ...)^T$.*
2. *Calculate $v^1 = Su^1$ and $v^2 = Su^2$.*
3. *If $b_0 = b_n = 0$, then for $i = 1, ..., n$*

$$a_i = \frac{1}{2}(v_i^1 + (-1)^i v_i^2) \tag{112}$$

and for $i = 1, ..., n-1$

$$b_i = \frac{1}{2}(v_i^1 - (-1)^i v_i^2) - b_{i-1} \tag{113}$$

The last step is from the condition that if S equals $\hat{S}$, then from the ith row of $\hat{S}$ we obtain

$$b_{i-1} + a_i + b_i = v_i^1 \tag{114}$$

and

$$(-1)^i(-b_{i-1} + a_i - b_i) = v_i^2, \tag{115}$$

for $i = 1, ..., n$.

J Operator

This approach exploits the solution of Laplacian with homogeneous boundary conditions discretized in uniform mesh on rectangular domain using a standard five-point (seven-point in three dimensions) finite difference stencil. For simplicity, we choose the case of the union of two square subdomains each with a uniform mesh of $n \times n$ interior mesh points with the mesh width $h = 1/(n+1)$. Following the language of Smith *et al.* [78] and the natural ordering, both $A_{II}^{(1)}$ and $A_{II}^{(2)}$ (matrices associated with the interior points in the subdomains) are diagonalizable by tensor product of the one dimensional discrete sine transform,

$$\mathcal{Q}_{ij} = \sqrt{\frac{2}{(n+1)}} \sin\left(\frac{ij\pi}{n+1}\right). \tag{116}$$

Hence, $A_{II}^{(i)} = (\mathcal{Q} \otimes \mathcal{Q})(\Lambda \otimes I + I \otimes \Lambda)(\mathcal{Q} \otimes \mathcal{Q})$, where the diagonal matrix Λ has components $\Lambda_{ii} = 4\sin^2(\frac{i\pi}{2(n+1)})$. If we define the diagonal matrices

$$\mathcal{A} = I + \frac{1}{2}\Lambda - (\Lambda + \frac{1}{4}\Lambda^2)^{1/2} \tag{117}$$

and

$$\mathcal{F} = (I - \mathcal{A}^{2n+2})^{-1}(I + \mathcal{A}^{2n+2}), \tag{118}$$

then it can be shown that

$$S^{(i)} = \mathcal{Q}\mathcal{F}^{1/2}(\Lambda + \frac{1}{4}\Lambda^2)^{1/2}\mathcal{F}^{1/2}\mathcal{Q}. \tag{119}$$

This is the explicit eigenvalue decomposition of the Schur complement. Then, by dropping the $\mathcal{F}$ terms and the second order term in Λ, we can have the *J operator* preconditioner

$$J = \mathcal{Q}\Lambda\mathcal{Q}. \tag{120}$$

We compare the convergence histories of the residual norms of the gradients to each other for the approximations of the Schur complement as in Figure 8. If we compare these approximate Schur complements with the exact, the fastest is the exact Schur complement. The slowest is the approximation by probing. Meanwhile, the Laplacian has the same rate as the J operator. It is interesting to note that the identity has the same rate as the BFGS, because we might expect that the BFGS should be a better preconditioner than the identity. This preconditioner is applied in [12, 13, 14, 11] to a problem in optimal control. The probing is very fast at the beginning but then stagnant at the end.

However, probing is still better than the unpreconditioned case as in the bottom right of Figure 6 and Figure 7. For more detail about the probing techniques as well as the J operators in solving PDE, we refer to Smith *et al.* book [78, pp. 119-124], [23], and the references therein, especially [52].

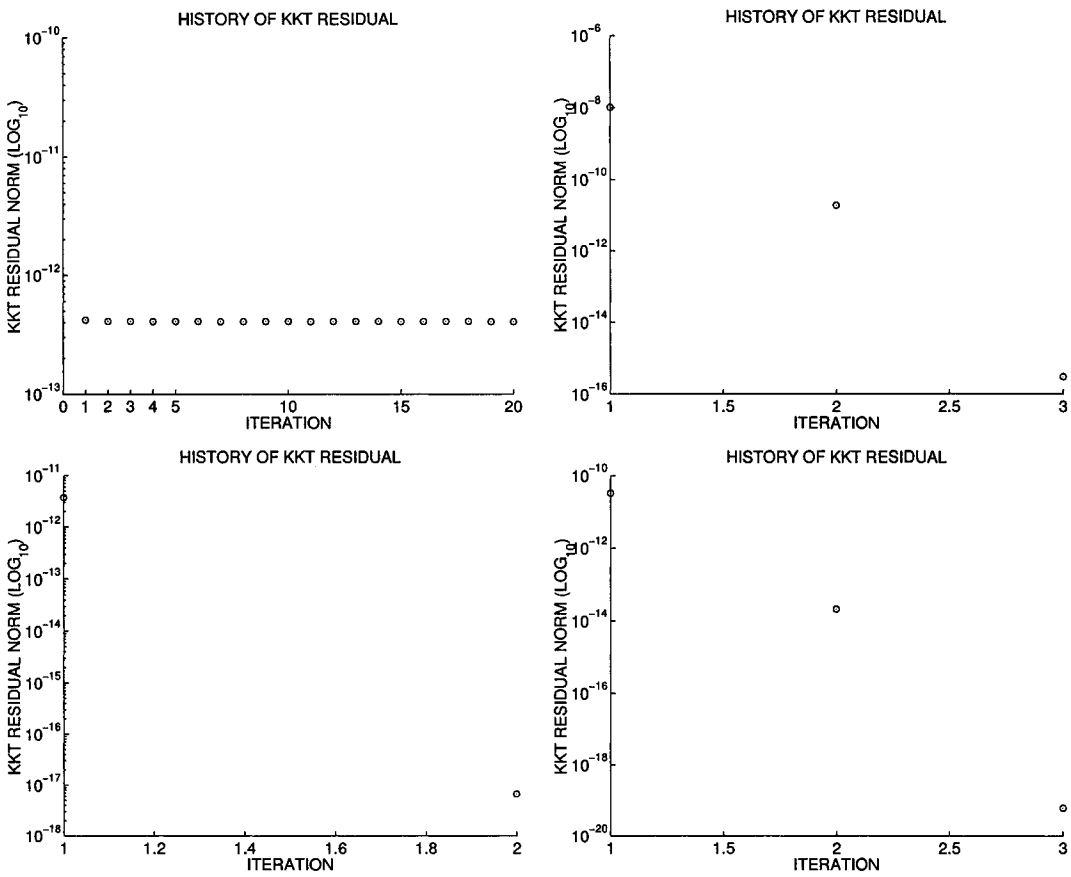

FIG. 8. *Convergence history of the residuals with the inexact Schur complements approximated by probing (top left), BFGS (top right), the Laplacian (bottom left), and the J operator (bottom right) with* 0% *noise.*

For further comparisons, we display history of the errors between the computed and the exact data (or the solution error norm) in Figure 9.

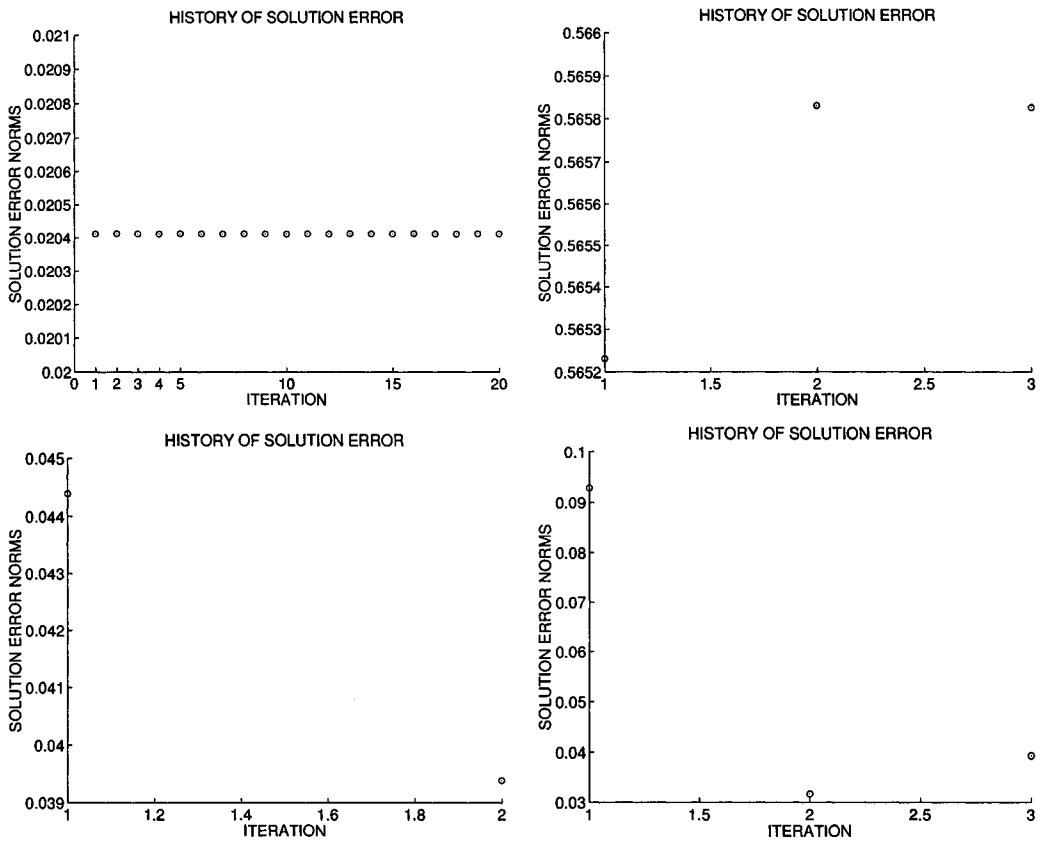

FIG. 9. *History of the errors between the exact and the computed parameters with the inexact Schur complements approximated by probing (top left), BFGS (top right), the Laplacian (bottom left), and the J operator (bottom right) with* 0% *noise.*

Notice that history of the errors by probing and Laplacian are coincide with convergence history of the KKT residuals but not for the BFGS and J operator. It is common in inverse problems that the fast convergence in the residual does not guarantee the fast convergence in the solution error due to the ill-posedness. Hence, a new stopping rule is needed besides using the residual. We will discuss this rule in Chapter 4. The error for the probing is constant at 0.0204 since the first iteration. The error for the Laplacian converges to 0.0394. The error for the J operator grows after the second iteration, in which it is 0.032. The error for the BFGS grows since the first iteration, in which it is 0.5852. From Figure 8 if we consider the maximum

tolerance for the residual is 1.0×10^{-12}, all the methods except the BFGS stop at the first iteration. Interestingly, and the error of the probing is the smallest as seen in Figure 9. Hence, in terms of solution error the probing is the best choice as a Schur approximation for the exact data case.

The matching between the exact and the computed parameters is shown in Figure 10.

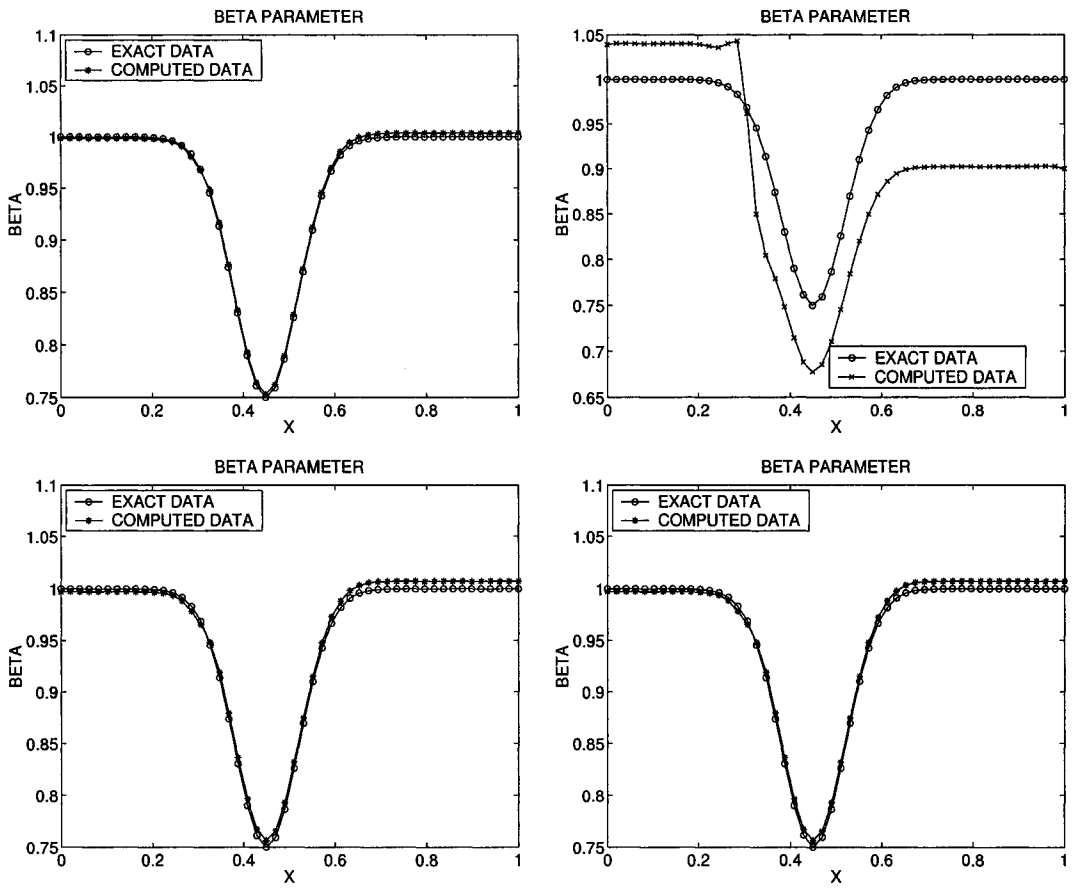

FIG. 10. *The matching between the exact and the computed parameters with the inexact Schur complements approximated by probing (top left), BFGS (top right), the Laplacian (bottom left), and the J operator (bottom right) with* 0% *noise.*

Furthermore, we present the spectra of the preconditioned systems in Figure 11.

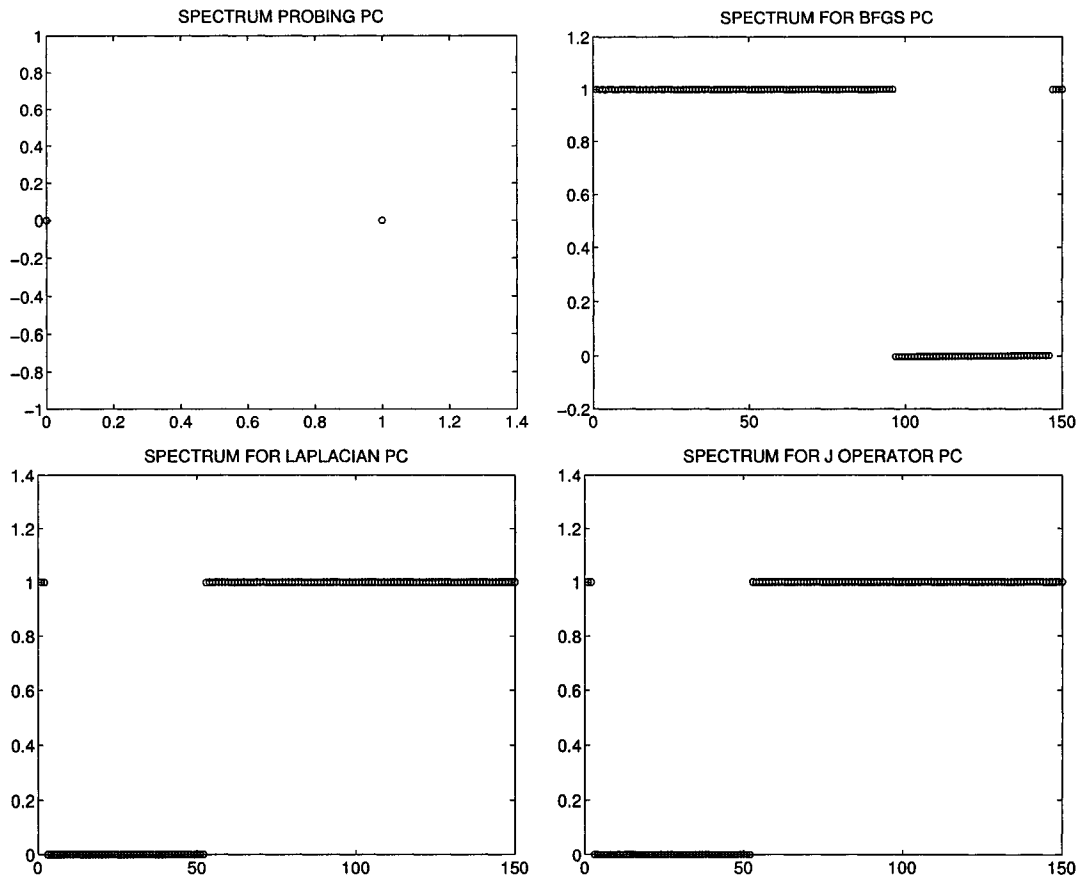

FIG. 11. *The spectrum of the KKT matrices preconditioned by probing on the complex plane (axes label by real and imaginary parts) (top left), BFGS (top right), the Laplacian (bottom left), and the J operator (bottom right) on the real line (real values plotted against index) with* 0% *noise.*

Noisy or Perturbed Data Case

The effectiveness of the third (block diagonal) preconditioner (PC3) by replacing the Schur complement with identity matrix compared to the first preconditioner with exact Schur complement in a case with a 1% noise data and a regularization parameter $\gamma = 1.0 \times 10^{-6}$ can be seen in Figure 12 for the spectra of the preconditioned KKT matrices, and Figure 13 for the matching parameters between the computed

and exact data. As we can see from Figure 12, the eigenvalues cluster to 0 and 1 for both Schur complements.

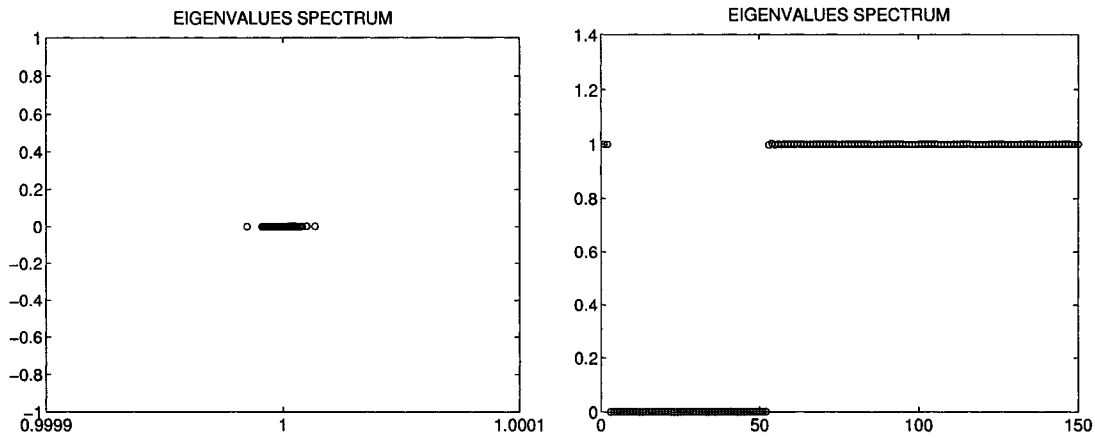

FIG. 12. *Spectrum of the KKT matrices preconditioned by the linear preconditioners PC1 using the exact Schur complement on the complex plane (axes labeled by real and imaginary parts) (left) and PC3 using the inexact Schur complements approximate by identity matrix on the real line (real values plotted against index) (right) with* 1% *noise.*

To achieve this optimal result (the closest matched parameters) as in Figure 13, we only need two nonlinear iterations with minimum residual norm approximately equal to 1.0×10^{-9}. The differences of the matching parameters are invisible. Hence, with an inexpensive approximation of the Schur complement we can still expect a good result for the noisy data.

Although it appears that the preconditioners are not significantly different, we still do need a preconditioner. Without a preconditioner the iteration will be stagnant as the top left figure in Figure 14. In Figure 14 we can also see the wide spectrum of the eigenvalues from 0 to 3.0 (bottom right), the unmatched parameters (top right), and the history of the solution error norm (bottom left).

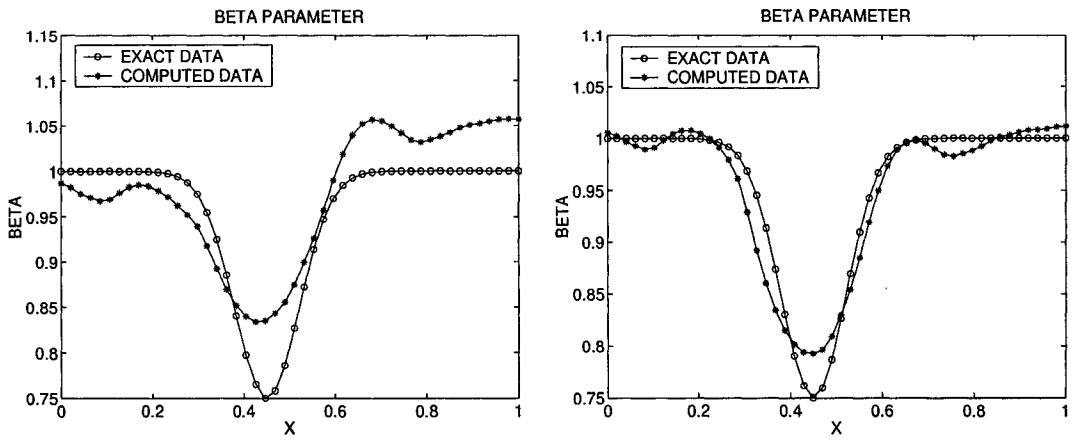

FIG. 13. *Matching parameters between the computed and exact data with the exact Schur complement for PC1 and the inexact (identity matrix) Schur complement for PC3 with* 1% *noise.*

3.5.2 Numerical Experiments with Example 2

For the exact data the differences among the three proposed linear preconditioners are invisible. We can see from Table 3 that all of the preconditioners are about equally efficient in term of execution time.

TABLE 3
Comparison of the preconditioners.

Preconditioner	*Execution (CPU) Time (second)*
PC1	168.4
PC2	167.3
PC3	169.1

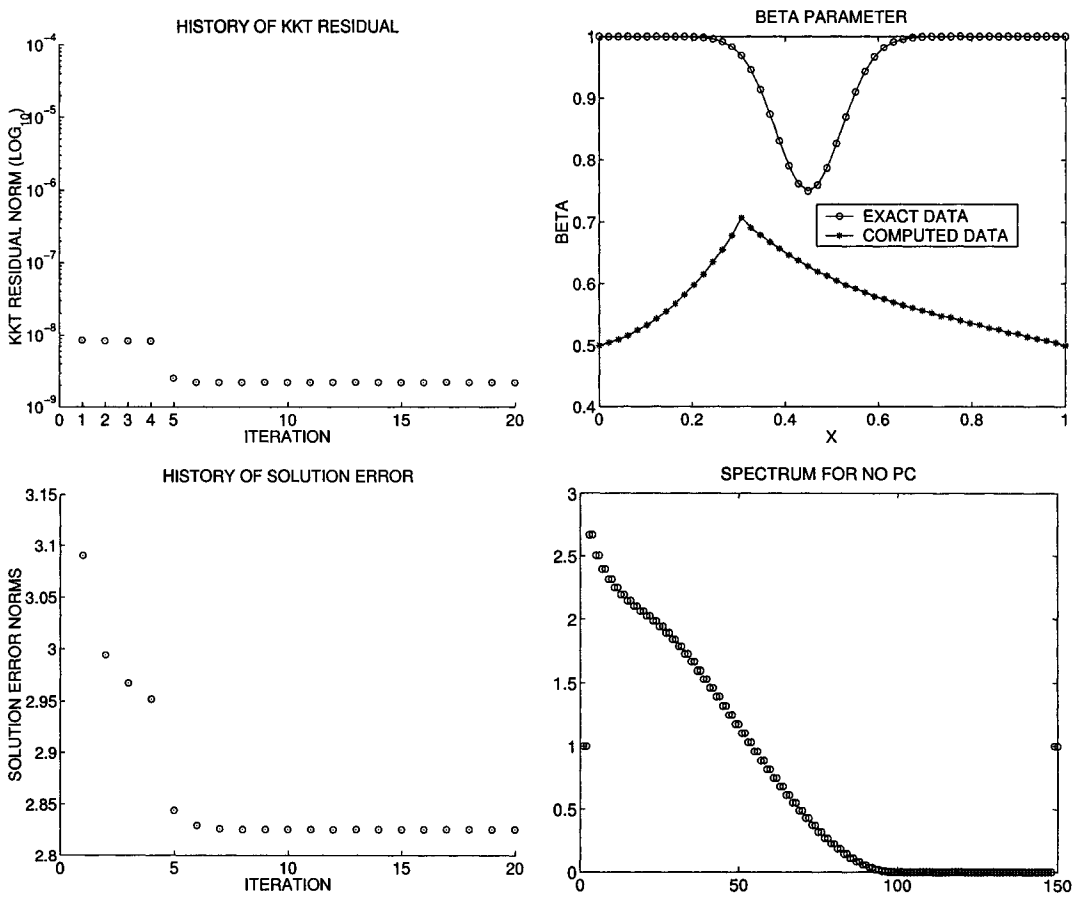

FIG. 14. *Convergence history of the residual (top left), the matching parameter (top right), convergence history of the parameter error (bottom left), and the spectrum on the real line (real values plotted against index) (bottom right) for* 1% *noise data without preconditioner.*

3.6 NUMERICAL EXPERIMENTS FOR NONLINEAR PRECONDITIONERS

The result of the nonlinear preconditioner for Numerical Example 2 with a 5% noise can be seen in the left part of the Figure 15.

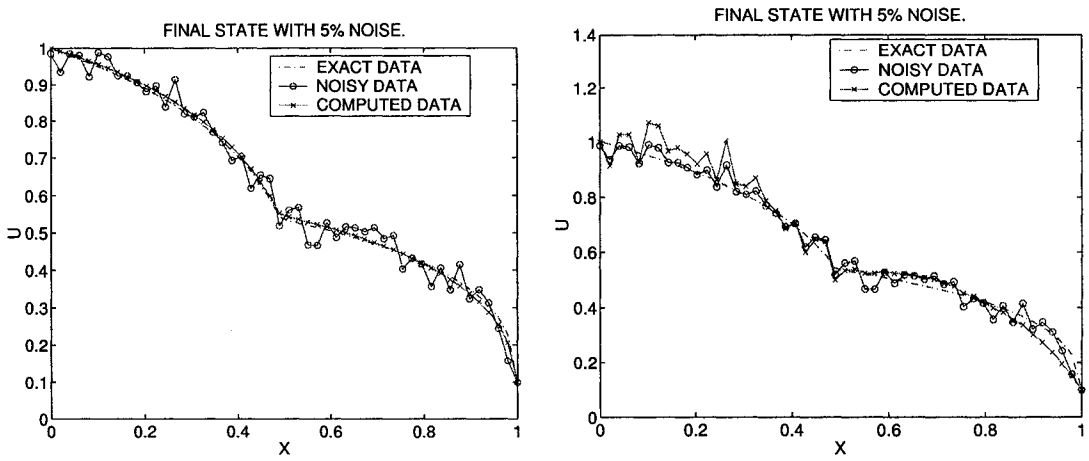

FIG. 15. *Each result when nonlinear preconditioner (left) and linear preconditioner (right) is applied to Example 2 with* 5% *noise.*

3.7 COMPARISON BETWEEN THE LINEAR AND NONLINEAR PRECONDITIONER

We use the 1-D Numerical Example 2 as the comparison tool. Firstly, we compare the three linear preconditioners to each other as in the previous subsection. Then, we compare one of them (the first linear preconditioner) with the nonlinear preconditioner.

We apply 5% noise to the exact data and use as initial data. Then, we find the exact data and the parameters. Comparing the left part and the right part of Figure 15, we see that the nonlinear preconditioner is superior to the linear one in terms of matching the computed data to the exact data. We note that the parameter recovered by using the nonlinear preconditioner is better than by using the linear one. By using the linear preconditioner we recover the parameters $\alpha = 2.2239$, $\beta_{left} = 1.0683$, and $\beta_{right} = 7.5636$ with a parameter regularization $\gamma = 1.0 \times 10^{-3}$. Meanwhile, by using the nonlinear preconditioner we can recover the parameters $\alpha = 2.4908$, $\beta_{left} = 1.0657$, and $\beta_{right} = 10.1840$ with a regularization parameter $\gamma = 1.0 \times 10^{-2}$.

3.8 SYMMETRIC VS. NONSYMMETRIC KKT MATRICES

As the first comparison tool we use our 1-D Numerical Example 1, where we choose

$$\beta(x) = -1 + 2.0e^{-\frac{(x-0.25)^2}{0.01}} + 1.0e^{-\frac{(x-0.75)^2}{0.02}}. \tag{121}$$

TABLE 4
Comparison of the Symmetric versus Nonsymmetric Methods.

Regularization Parameter	*Outer*	*Inner*	*Time*	*Parameter*
Preconditioner	*Iterarion*	*Iteration*	*(second)*	*Error*
10^{-9}:				
PC1	4	4	268.4	2.85
PC2	4	7	269.1	2.85
PC3	4	12	268.5	2.85
PCsm	5	5	339.6	1.09
10^{-11}:				
PC1	2	2	134.4	0.69
PC2	2	3	134.4	0.69
PC3	2	6	134.3	0.69
PCsm	2	2	136.5	0.45
10^{-13}:				
PC1	1	1	66.8	0.37
PC2	1	1	67.1	0.37
PC3	1	3	67.3	0.37
PCsm	1	2	68.7	0.27

This example is adopted from Haber and Ascher [42]. For the symmetric case we use the SQMR (Symmetric Quasi Minimum Residual) method also used by Haber and Ascher [42], Biros and Ghattas [12, 13, 14, 11], Burger and Muhlhuber [16, 17], and also Battermann and Sachs [8] with the *LU* preconditioner type. Meanwhile, for the nonsymmetric cases we use GMRES with the three proposed preconditioners. We implemented these in the framework of MATLAB and ADMAT with 50 subintervals in the domain. As shown in Table 4 in terms of execution time, error norm, and iteration counts the preconditioners are almost indistinguishable. The

maximum tolerance of the residual (gradient) norm is 1.0×10^{-15}. Hence, the nonsymmetric methods and preconditioners are competitive compared to the symmetric ones (PCsm). Our approaches are better than the symmetric ones in terms of independence of the regularization parameters, as we see in the 1-D case, and in terms of the implementation on the simple block diagonal case, as we can see in the next 2-D case. Here, we note that when using the block diagonal preconditioner the computation of the symmetric approach becomes stagnant. Our 1-D example with SQMR also shows this. Here, the preconditioners for the symmetric cases are denoted by PCsm.

As a further comparison tool we consider the 2-D test case. This is the 2-D example for Numerical Example 1, where the true conductivity $\alpha(x)$ is as in the top left of Figure 31 and 32. We adopted this example from Vogel [85]. We implemented in PETSc and ADIC with 64×64 grids. We use the simplest preconditioners, i.e. the block diagonal preconditioner with the Schur complement approximated by identity matrix. Our numerical example shows that the nonsymmetric case is preferable to the symmetric one regarding the spectrum and rate of convergence, as shown in Figure 16 (left) and (right) respectively.

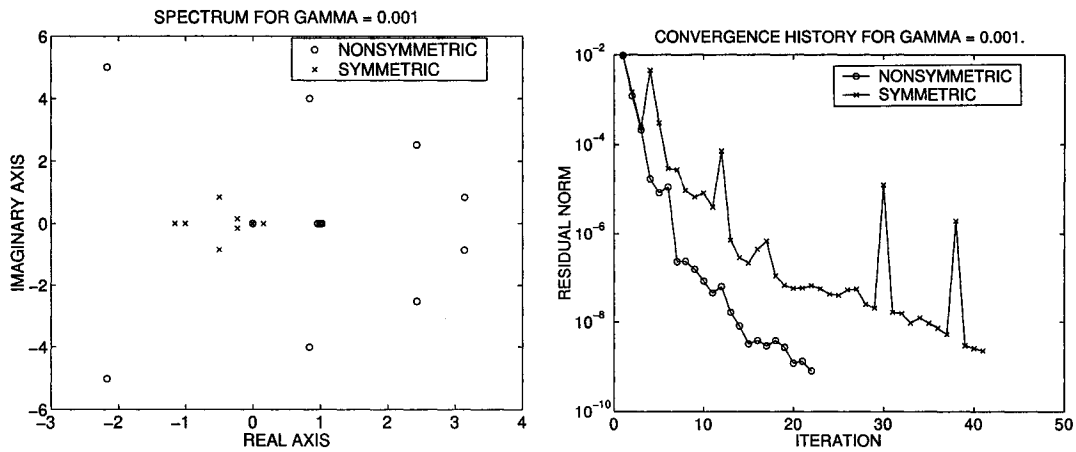

FIG. 16. *Results of the spectrum (left) and the residual norm history (right) for symmetric and unsymmetric case are implemented to the 2-D model problem with* 0% *noise.*

With regard to the spectrum as in Figure 16 (left), the symmetric case has eigenvalues that are closely clustered to the origin and many negative eigenvalues (severely indefinite). This kind of spectrum is undesirable for the application of Krylov methods. Meanwhile, in the spectrum of the nonsymmetric case the eigenvalues are clustered away from the origin and mildly indefinite. For both cases we use GMRES. From Figure 16 (right) we see that the rate of convergence for the nonsymmetric computation is about 50% faster than the symmetric.

CHAPTER 4

REGULARIZATION STRATEGIES

The elliptic inverse problem is generally ill-posed. Hence, regularization is needed. Choosing the regularization is not an easy task, because we not only need to define the regularization operator, but also we need to determine the magnitude of the regularization parameter. This makes the problem difficult even for exact data. For noisy data we have to handle another problem, that is, a noisy operator. It is impossible to recover the original underlying parameter, but we may impose desirable features through choice of the regularization strategy.

In this section, we consider different regularization operators and regularization parameter choice rules. The regularizations we propose are iterative Tikhonov and Total Variation regularizations. Basically, these regularizations are Tikhonov and Total Variation regularizations with the regularization parameters changing during the iterative processes.

Our regularization is inspired by the Landweber iterative regularization that was proposed in 1951 by Landweber [62], its variants [16, 17, 76, 33], and the references therein. However, our iterative regularizations are based not only on the iteration counts as in Landweber methods, but also on the regularization parameter α. Hence, basically our regularization methods are combinations of iterative regularizations and functional regularizations. Therefore, the advantages of these two methods can be combined.

As shown in [56, pp. 109-111] for Symm's integral equation, the Tikhonov functional regularization is fast but unstable. Meanwhile, the Landweber iterative regularization is slow but stable. By combining these two methods we hope to achieve a method that is both fast and stable.

In selection of the regularization parameters we combine two of the methods that we will mention later, *a priori* and *a posteriori* rules. Actually, the combined rule is a direct combination of both regulations mentioned above.

4.1 ILL-POSEDNESS AND REGULARIZATION THEORY

We adopt from Vogel's book [86, p. 16] the definition of well-posedness as follows:

Definition 4.1 *Let $K : \mathcal{H}_1 \to \mathcal{H}_2$. An operator equation*

$$K(f) = g \tag{122}$$

is said to be ***well-posed*** *provided*

- *for each* $g \in \mathcal{H}_2$ *there exists* $f \in \mathcal{H}_1$, *called a solution, for which (122) holds;*
- *the solution* f *is unique; and*
- *the solution is stable with respect to perturbation in* g. *This means that if* $Kf_* = g_*$ *and* $Kf = g$, *then* $f \to f_*$ *whenever* $g \to g_*$.

A problem that is not well-posed is called to be **ill-posed**.

For general regularization theories we can follow the notations and the explanations in Kirsch [56], Isakov [47], and Vogel [86].

From Kirsch [56, p.25] we adopt the definition of the regularization strategy.

Assume that there is a perturbed equation

$$Kx^\delta = y^\delta \tag{123}$$

relative to the unperturbed equation

$$Kx = y. \tag{124}$$

We need to construct the bounded approximation $R : Y \to X$ of the (unbounded) inverse operator $K^{-1} : K(X) \to X$.

Definition 4.2 *A regularization strategy is a family of linear and bounded operators*

$$\mathcal{R}_\alpha : Y \to X,$$

for a parameter choice rule $\alpha = \alpha(\delta, y^\delta)$, *such that*

$$\lim_{\alpha \to 0} \mathcal{R}_\alpha K x = x \tag{125}$$

for all $x \in X$, *i.e., the operators* $R_\alpha K$ *converge pointwise to the identity.*

4.2 A PRIORI AND A POSTERIORI PARAMETER CHOICE RULES

From the definition of regularization, α is the *regularization parameter.*

The choice rules of these parameters can be divided into two categories, *a priori* and *a posteriori* parameter choice rules.

According to the book by Engl *et al.* [34, p. 51], we define these rules as follows:

Definition 4.3 *Let α be a parameter choice rule according to Definition 4.2. If α does not depend on y^δ, but only on δ, then we call α an a priori parameter choice rule and write $\alpha = \alpha(\delta)$. Otherwise, we call α an a posteriori parameter choice rule.*

The first rule depends only on the noise level, not on the noisy data itself. Hence, it has to be chosen before the computational process. For exact data, this rule can easily be applied by a "trial and error" approach. Our numerical experiments show that upon decreasing the regularization parameter by an order of magnitude, we observe results that are close to the optimum parameter selection. However, for noisy or perturbed data this approach is very difficult to apply. It is also time-consuming.

An example of this first rule is in [34, p. 74] as **Corollary 4.4** and **Remark 4.5**. Here, it is suggested to choose $\alpha \sim \delta^{\frac{2}{2\mu+1}}$, where μ has to be known. Otherwise, we have to use the *a posteriori* rules.

Meanwhile, the second rule can be determined dynamically, during the computation, and can be related to $\|Tx_\alpha^\delta - y^\delta\|$.

One of the methods widely used that is based on the second rule is called *discrepancy principle*, due to Morozov [65]. There are several variants of this principle as in [83, 84].

One of the versions is as follows:

Let $g_\alpha : [0, \|T\|^2] \to \mathcal{R}$ be piecewise continuous. Let there be a $C > 0$ such that $|\lambda g_\alpha(\lambda)| < C$, and $\lim_{\alpha \to 0} g_\alpha(\lambda) = \frac{1}{\lambda}$ for all $\lambda \in (0, \|T\|^2]$, and let $r_\alpha := 1 - \lambda g_\alpha(\lambda)$. Furthermore, let $\tau > \sup\{|r_\alpha(\lambda)| \mid \alpha > 0, \lambda \in [0, \|T\|^2]\}$.

The regularization parameter determined by *discrepancy principle* is

$$\alpha(\delta, y^\delta) := \sup\{\alpha > 0 \mid \|Tx_\alpha^\delta - y^\delta\| \leq \tau\delta\}. \tag{126}$$

Unfortunately, this approach needs *a priori* knowledge of the error level $\|Tx_\alpha^\delta - y^\delta\|$. In practice, this is often difficult to get. Therefore, this approach is not necessarily superior to the previous one. However, if we have the knowledge, the last method is the method of choice. For variants of these methods, see [86, Ch. 7].

This reference also describes a method that needs no prior information of the noise. It is called *L-Curve method.* However, the analysis in [86] shows that this method may not be convergent. Therefore, the prior information of the noise in the data should be available in solving of practical ill-posed problems.

4.3 TIKHONOV REGULARIZATION

Tikhonov regularization was proposed initially by A. N. Tikhonov [81, 82]; sometimes it is also called *Tikhonov-Phillips regularization* [70]. There are two forms of Tikhonov Regularization, without derivative and with derivative.

In the discrete form, the Tikhonov regularization with a first derivative is

$$R(p) = \frac{1}{2}\|\nabla p\|^2, \tag{127}$$

while without derivative it is

$$R(p) = \frac{1}{2}\|p\|^2. \tag{128}$$

4.4 TOTAL VARIATION REGULARIZATION

As in [86, pp. 129-134] this regularization based on the following definition of the Total Variation (TV) of a function f defined on the interval $[0, 1]$:

$$TV(f) \stackrel{\text{def}}{=} \sup \sum_i |f(x_i) - f(x_{i-1})|, \tag{129}$$

where the supremum is taken over all partitions $0 = x_0 < x_1 < \ldots < x_n = 1$ of the interval.

If f is smooth, for two space dimensions we can write:

$$TV(f) = \int_0^1 \int_0^1 |\nabla f| \, dx \, dy, \tag{130}$$

where $\nabla f = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial x})$ denotes the gradient and $|(x, y)| = \sqrt{x^2 + y^2}$ denotes the Euclidean norm.

If f is not smooth, we can represent this in weak form as follows:

$$TV(f) = \sup_{v \in \mathcal{V}} \int_0^1 \int_0^1 f(x, y) \, \text{div}\, \vec{v} \, dx \, dy, \tag{131}$$

where $\mathcal{V}$ consists of vector-valued functions $\vec{v} = (v_1(x, y), v_2(x, y))$ whose Euclidean norm is bounded by 1 and whose components v_i are continuously differentiable and

vanish on the boundary of the unit square. $\operatorname{div}\vec{v} = \frac{\partial v_1}{\partial x} + \frac{\partial v_2}{\partial y}$ gives the divergence of $\vec{v}$.

However, the weak representation is difficult to apply. Hence, as in Vogel's book [86, p. 130], we can represent the approximation of the Total Variation regularizations in a two-dimensional space as follows:

$$R(p) = \int_0^1 \int_0^1 |\nabla p| dx dy \sim \int_0^1 \int_0^1 \sqrt{|\nabla p|^2 + \beta^2}\, dx\, dy, \tag{132}$$

where β is a small positive parameter for the approximation to the Euclidean norm.

4.5 ITERATIVE REGULARIZATION

The iterative regularization used in solving inverse problems is very simple. It depends only on the iteration count as the regularization parameter. However, the method can be limiting in practice due to too many iterations en route to an accurate regularized solution. One of the methods is Landweber Iterative Regularization.

4.5.1 Landweber Iterative Regularization

Landweber Iterative Regularization was originally used in solving first-kind Fredholm integral equations, and was introduced by L. Landweber in 1951. See [62]. It can be explained as follows: Consider the following least squares fit data functional

$$J(f) = \frac{1}{2}\|Kf - d\|^2. \tag{133}$$

The iteration is

$$f_{v+1} = f_v - \tau \text{ grad } J(f_v), \tag{134}$$

for $v = 0, 1, \ldots$, where v is the iteration count, τ is chosen to minimize $J(\tau) = J(f_v) - \tau$ grad $J(f_v)$, and $0 < \tau < \frac{1}{\|K\|^2}$.

4.6 PROPOSED REGULARIZATION METHODS

As mentioned previously, we propose new types of regularizations, iterative regularizations based on gradient types of Tikhonov and Total Variation functionals. On these approaches the regularizations are not only depent on the iteration count but also on the iterative regularization parameter and the regularization functional.

4.6.1 Iterative Tikhonov and Total Variation Regularization

Consider the Lagrangian functional

$$L(u,p,\lambda) = \frac{1}{2}\|u - u_d\|^2 + \lambda h(u,p) + \gamma_k R(p), \tag{135}$$

where u_d is the data, $h(u,p)$ is the discretized PDE, $\gamma_k = \mu^k \gamma_0$ is the iteration regularization parameter, and $R(p)$ is the regularization functional.

The purpose of the method is to minimize the Lagrangian functional with respect to u, p, and λ for optimal values of the parameter regularization γ.

If $R(p)$ is a Tikhonov functional, we call the method iterative Tikhonov regularization. Likewise, if $R(p)$ is Total Variation functional, we call it iterative Total Variation regularization.

4.7 STOPPING CRITERIA FOR REGULARIZED ALGORITHM

We use a stopping rule based on the *discrepancy principle* as follows: For noisy data, if

$$\|u^n - u_d^\delta\| < \delta r < \|u^{n+1} - u_d^\delta\|, \tag{136}$$

where $r > 1$ is a constant and δ is the noise level, then we stop the iteration process.

With noisy data our numerical experiments show that this stopping rule may lead to the optimal solution (the closest approximation to the underlying solution) for every regularization parameter that we selected. Without this stopping rule the solution may converge to a non-optimal solution, although the residual of the Newton step reaches the maximum tolerance. This is illustrated in the history of the solution error norms in Figure 27 and the history of the residual (gradient) norms in Figure 28. Here, we see that the itetarions should be stopped at the third or fourth iterations.

For the comparison between Landweber and conjugate gradient method applied to Symm's integral equation with respect to the stopping parameter r, see [56, p. 111]. The Landweber method is stable but slow with respect to the stopping parameter r. Meanwhile, the conjugate gradient method is sensitive to the exact stopping rule but the error decreases very quickly. The latter has the same behavior as the Tikhonov regularization method in [56, pp. 109-110].

4.8 NUMERICAL STUDY

4.8.1 Case with Exact Data

First of all, we study and compare the Tikhonov and Total Variation regularizations with respect to the regularization parameters as illustrated in Figure 17 and 18, respectively. Then, we compare the standard (noniterative) and iterative regularizations.

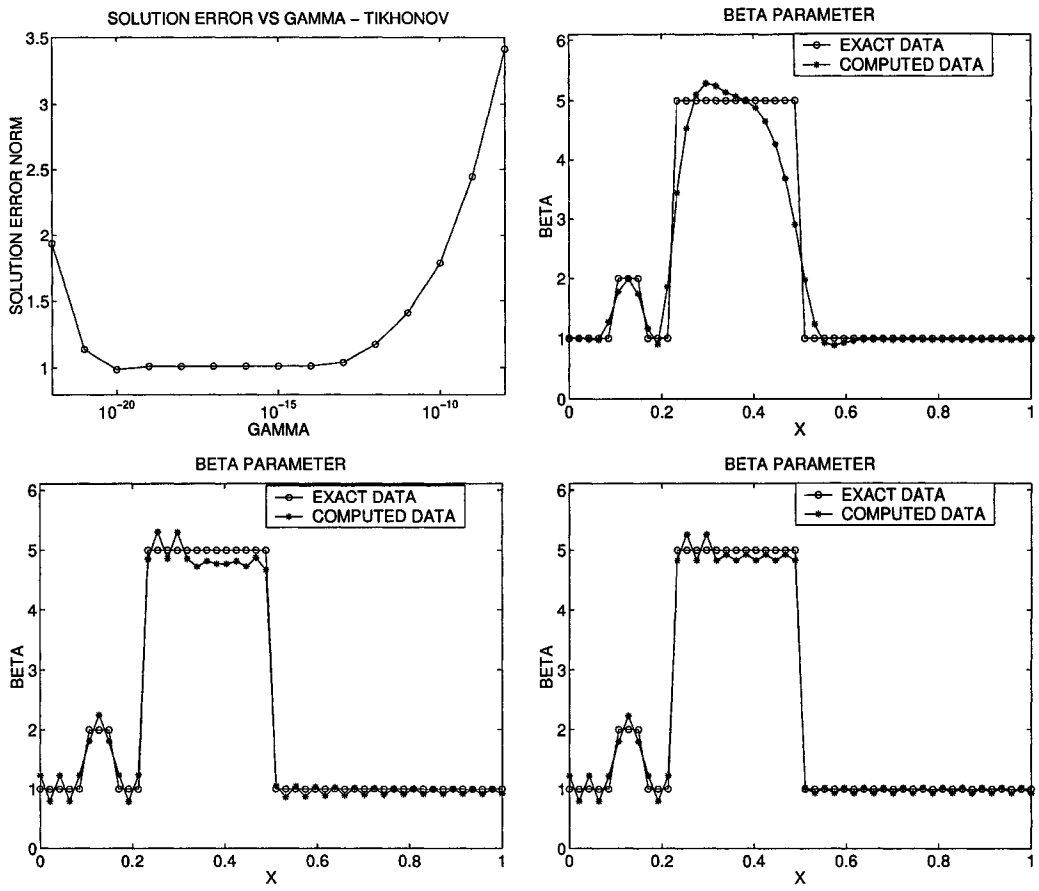

FIG. 17. *Solution error norm versus the regularization parameter γ (top left), the matching parameter for $\gamma = 1.0 \times 10^{-8}$ (top right), the matching parameter for $\gamma = 1.0 \times 10^{-12}$ (bottom left), and the matching parameter for $\gamma = 1.0 \times 10^{-20}$ (bottom right) for 0% noise data with PC1 and Tikhonov regularization.*

Tikhonov vs. Total Variation Regularization

In the study of the Tikhonov and the Total Variation regularizations, we use the 1-D Numerical Example 1 and choose the diffusivity constant

$$\beta(x) = \begin{cases} 2.0 & \text{if } 0.09 < x < 0.1 \\ 5.0 & \text{if } 0.225 < x < 0.5. \\ 1.0 & \text{elsewhere} \end{cases} \tag{137}$$

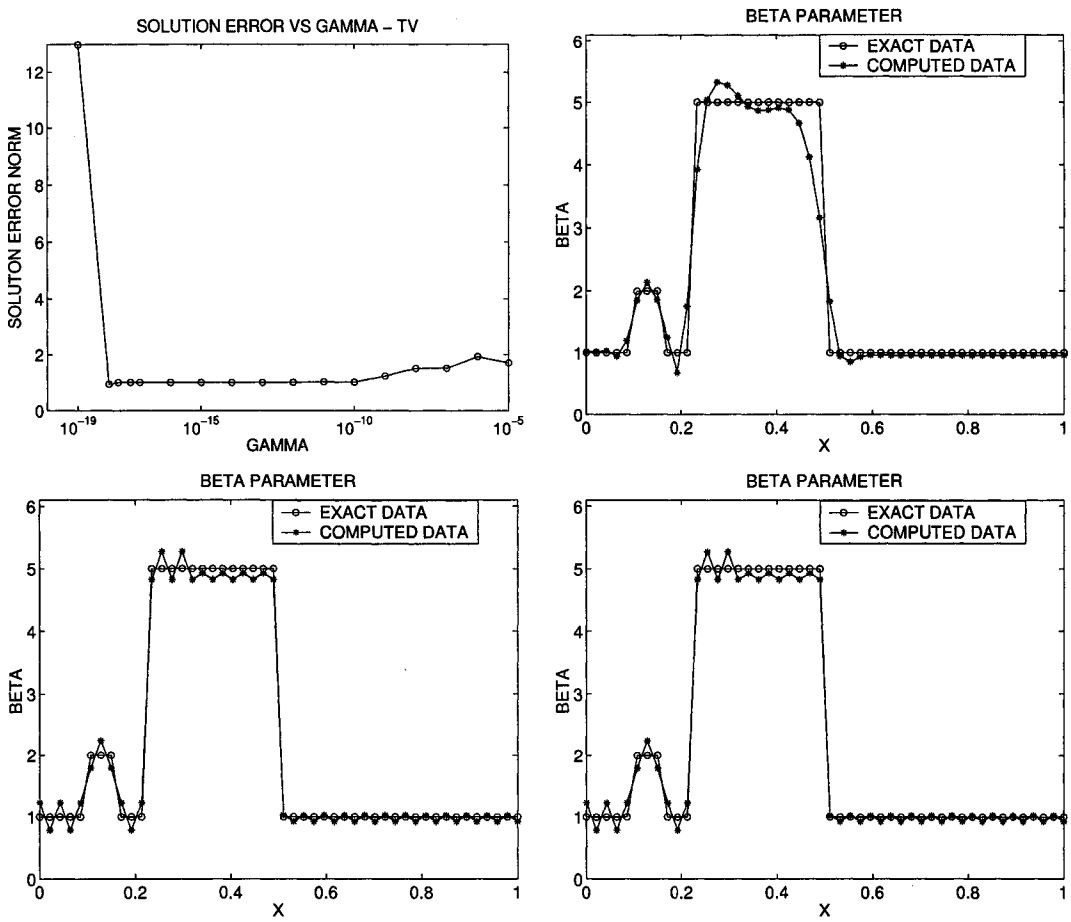

FIG. 18. *Solution error norm versus the regularization parameter γ (top left), the matching parameter for $\gamma = 1.0 \times 10^{-8}$ (top right), the matching parameter for $\gamma = 1.0 \times 10^{-15}$ (bottom left), and the matching parameter for $\gamma = 1.0 \times 10^{-18}$ (bottom right) for 0% noise data with PC1 and Total Variation regularization.*

We illustrate results based on the solution error norms versus the regularization parameters, and the matching between the exact parameter and the computed data with respect to different the regularization parameters for the Tikhonov and Total Variation regularization as in Figure 17 and 18, respectively.

The Tikhonov regularization is more robust than the Total Variation regularization, as can be seen by comparing the solution error norms vs the regularization parameters in the top left of Figure 17 and 18, respectively. From Figure 18 (top left), for Total Variation regularization) if the regularization parameter closes to zero, the solution error norm becomes bigger rapidly. Meanwhile, for the Tikhonov regularization as in the top left of Figure 17 the solution error norm does not jump so rapidly.

From the top right, left and right bottom of Figure 17 and 18 we can see that the Total Variation regularization is better than Tikhonov regularization for recovering the parameter with a jump discontinuity. This result can also be found in the literature; see e.g. Vogel's book [86, Ch. 8]. However, this is not necessarily the case for a problem with noisy data, as we discuss in the next subsection.

For this particular test case the proposed iterative regularizations are identical with the regular Tikhonov and Total Variation regularizations respectively, because only one iteration is needed in order to converge to the exact solution.

Iterative vs. Noniterative

To further investigate the advantages of iterative and noniterative regularization we use again our 1-D Numerical Example 1 with $\beta(x)$ as Equation (121)

$$\beta(x) = -1 + 2.0e^{-\frac{(x-0.25)^2}{0.01}} + 1.0e^{-\frac{(x-0.75)^2}{0.02}}.$$

First, we run the code with fixed $\gamma = 1.0 \times 10^{-9}$, then we run with iterative $\gamma = (0.2)^k \times 10^{-9}$. From the Table 5 we can see that the iterative regularization is competitive to or even better than the regular regularization in terms of the iteration counts, the execution-time, and the solution error norms for Numerical Example 1. The maximum tolerance of the residual (gradient) norm is 1.0×10^{-15}. Here, we use Tikhonov regularization.

TABLE 5
Comparison of the Iterative versus Noniterative Tikhonov Regularizations.

Regularization Parameter	*Outer*	*Inner*	*Time*	*Parameter*
Preconditioner	*Iteration*	*Iteration*	*(second)*	*Error*
10^{-9}:				
PC1	4	4	268.4	2.85
PC2	4	7	269.1	2.85
PC3	4	12	268.5	2.85
PCsm	5	5	339.6	1.09
$(0.18)^k 10^{-9}$:				
PC1	3	3	205.8	0.82
PC2	2	3	204.3	0.82
PC3	3	9	205.3	0.82
PCsm*	5	7	345.3	1,833

Surprisingly, for the symmetric case the computation failed; see the PCsm* (preconditioner for the symmetric case) row on the bottom of the Table 5. The iterative regularization caused the computation to diverge. Though the cause for this failure is not yet understood, it highlights interest in the nonsymmetric case for iterative regularization.

4.8.2 Case with Noisy Data

Tikhonov vs. Total Variation Regularization

We assume that the data have 1% noise for the same test case as in § 4.8.1. We pose the noise by using a normally distributed random number generator. The stopping rule follows the discrepancy principle as in § 4.7.

There is an interesting observation here that Tikhonov regularization is more robust than Total Variation regularization in the terms of solution error norms vs regularization parameters and the matching between the exact and the computed parameters. In all of these aspects the Tikhonov regularization is better than the Total Variation regularization. The Tikhonov regularizations can come the closest to the actual parameter, as in the left bottom of Figure 19. According to the left top of the Figure 19, the optimum solution error norm is around 3.0.

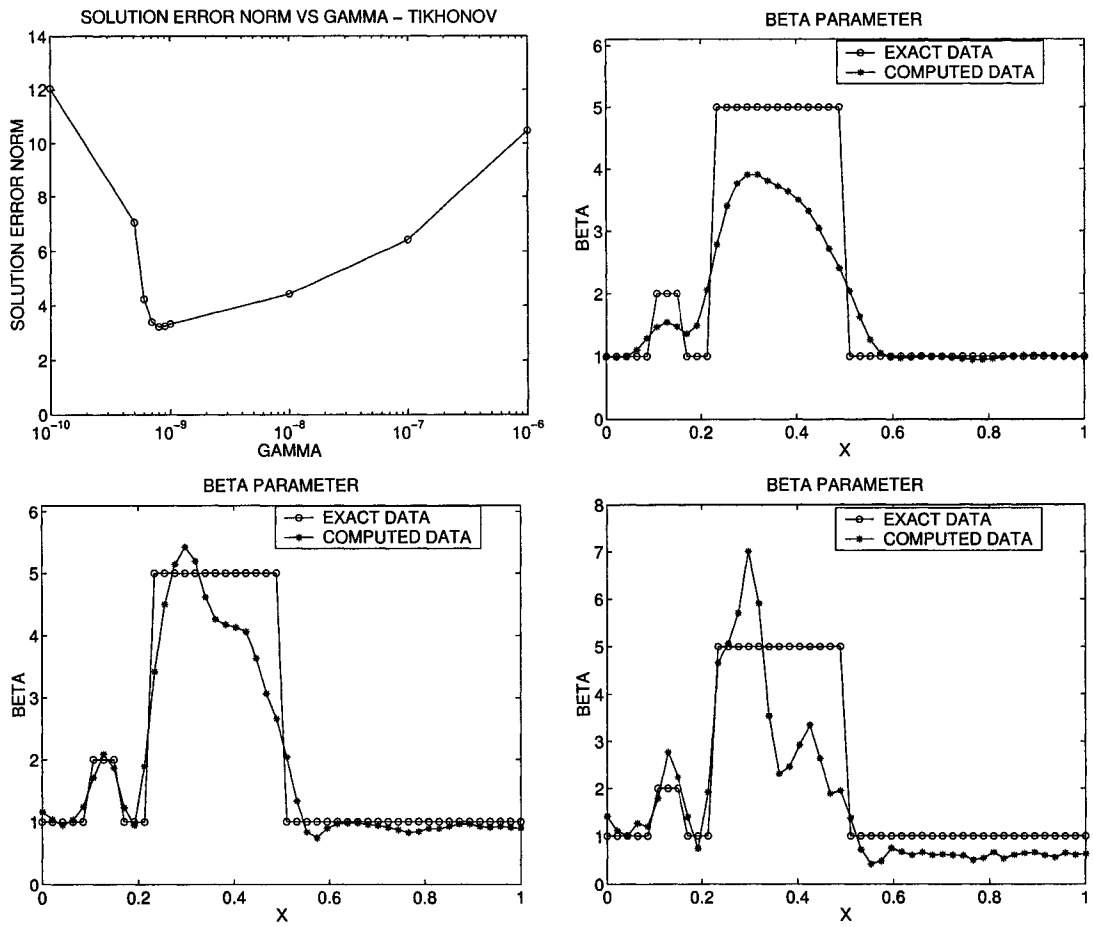

FIG. 19. *Solution error norm versus the regularization parameter γ (top left), the matching parameter for $\gamma = 1.0 \times 10^{-7}$ (top right), the matching parameter for $\gamma = 1.0 \times 10^{-8}$ (bottom left), and the matching parameter for $\gamma = 1.0 \times 10^{-9}$ (bottom right) for 1% noise data with PC1 and Tikhonov regularization.*

Meanwhile, the solutions from the functional regularized by Total Variation is far from the true solution, as we see in Figure 20. We also observe from the top left of Figure 20 that the optimum solution error norm is above 6.0. This is twice as large as the Tikhonov approach.

Iterative vs. Noniterative

The proposed iterative Tikhonov regularization is competitive to the standard Tikhonov regularization for certain criteria. By combining the *a posteriori* parameter selection with the iterative and functional regularization, we found interesting results.

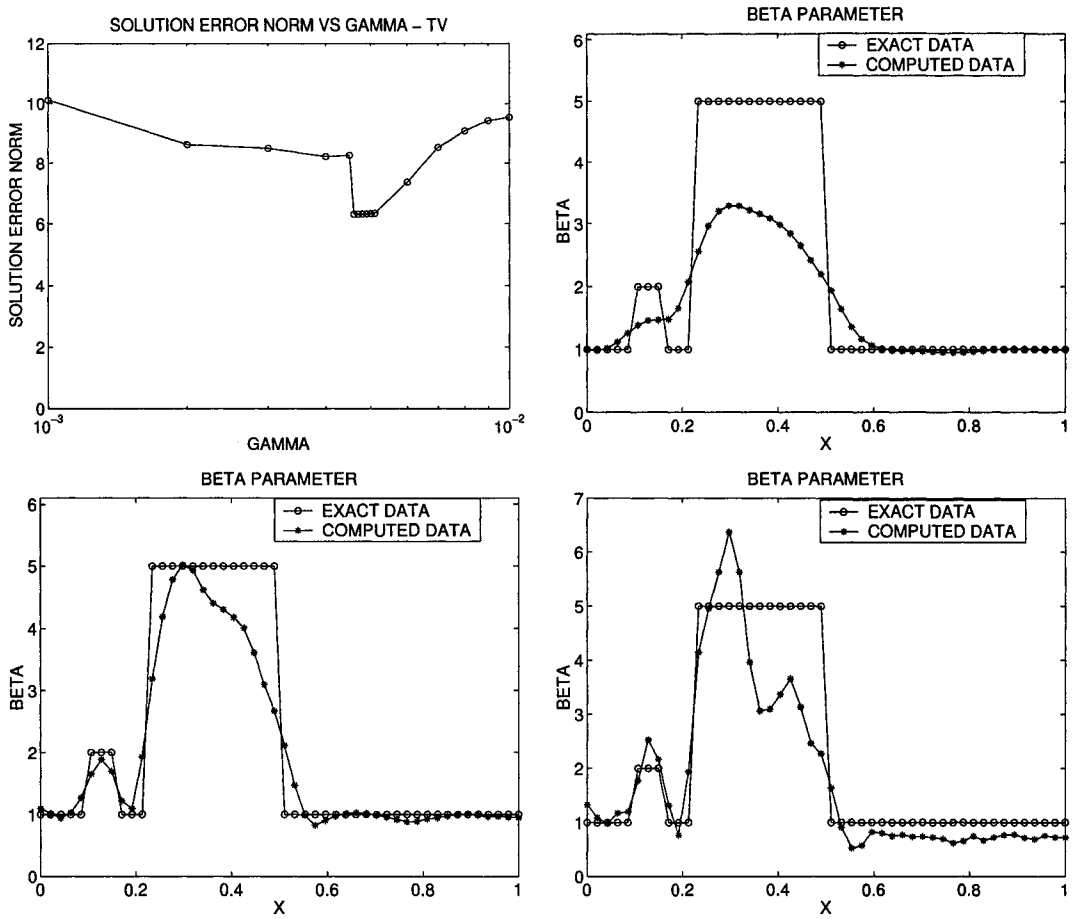

FIG. 20. *Solution error norm versus the regularization parameter γ (top left), the matching parameter for $\gamma = 1.0 \times 10^{-6}$ (top right), the matching parameter for $\gamma = 1.0 \times 10^{-7}$ (bottom left), and the matching parameter for $\gamma = 1.0 \times 10^{-8}$ (bottom right) for 1% noise data with PC1 and Total Variation regularization.*

The result of the iterative Tikhonov regularization for 1-D Numerical Example 1 with $\beta(x)$ as Equation (108),

$$\beta(x) = 1 - 0.25e^{-\frac{(x-0.45)^2}{0.01}}$$

is better than the result of the regular Tikhonov regularizations in terms of the matching parameters and the convergence histories of the solution error norms, respectively; see Figure 21 and 22.

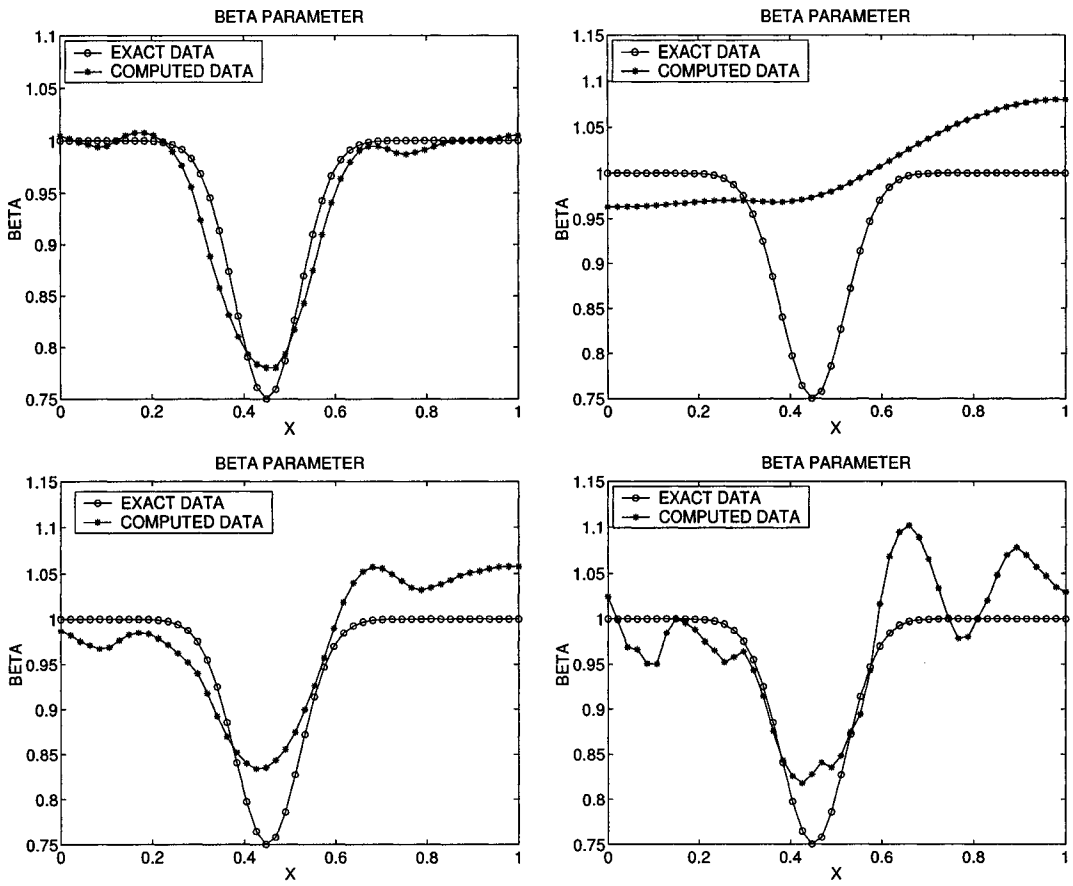

FIG. 21. *Matching parameter for* $\gamma_k = (0.12)^k \times 10^{-4}$ *(top left), the matching parameter for* $\gamma = 1.0 \times 10^{-4}$ *(top right), the matching parameter for* $\gamma = 1.0 \times 10^{-6}$ *(bottom left), and the matching parameter for* $\gamma = 1.0 \times 10^{-7}$ *(bottom right) for* 1% *noise data with PC1, the exact Schur complements and the iterative Tikhonov regularization.*

Meanwhile, for the convergence histories of the residual norms, the iterative regularization is competitive to the standard one. As in Figure 23, we see that the residual norm of the iterative regularization converges fast in the first four iterations. In addition, for the second and third iterations the solution error norm is at its minimum as in Figure 22. Notice that the error grows. This is due to the ill-posedness, which is common in inverse problems.

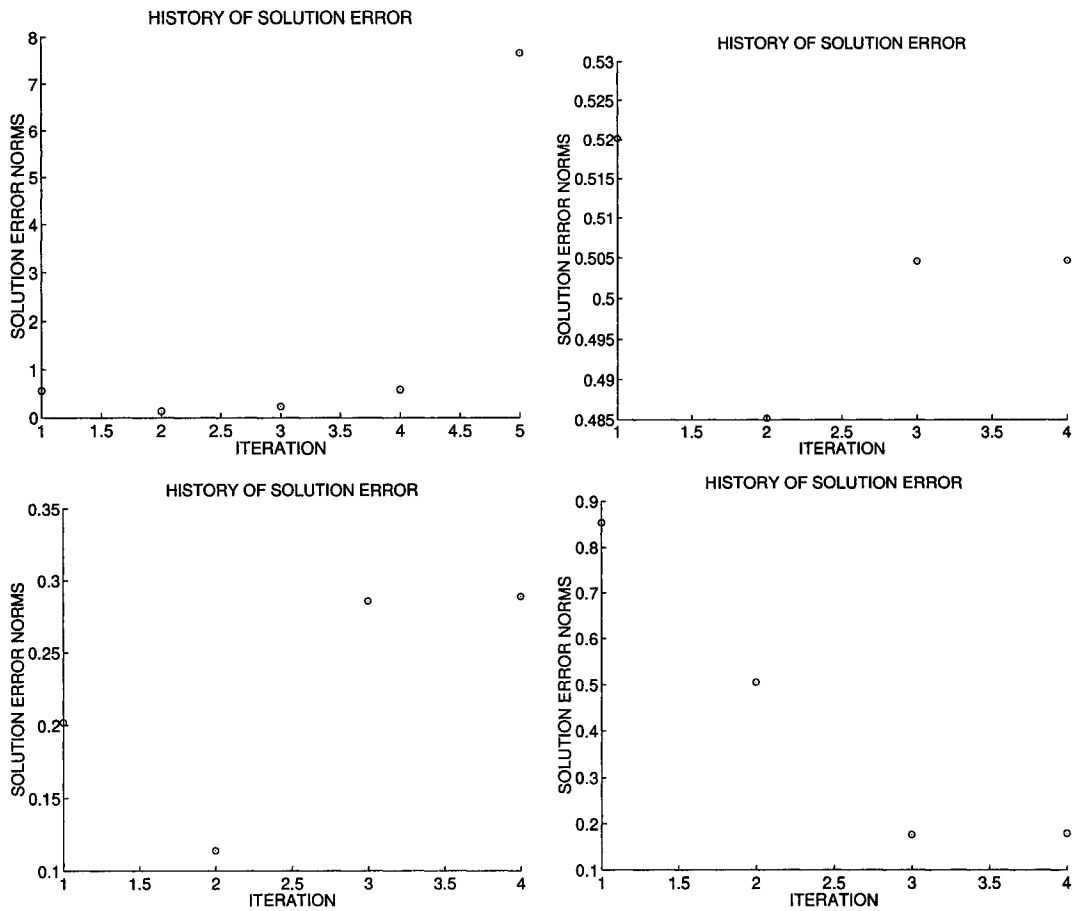

FIG. 22. *Convergence history of the solution error norm for* $\gamma_k = (0.12)^k \times 10^{-4}$ *(top left), for* $\gamma = 1.0 \times 10^{-4}$ *(top right), for* $\gamma = 1.0 \times 10^{-6}$ *(bottom left), and for* $\gamma = 1.0 \times 10^{-7}$ *(bottom right) for* 1% *noise data with PC1, the exact Schur complements and Tikhonov regularization.*

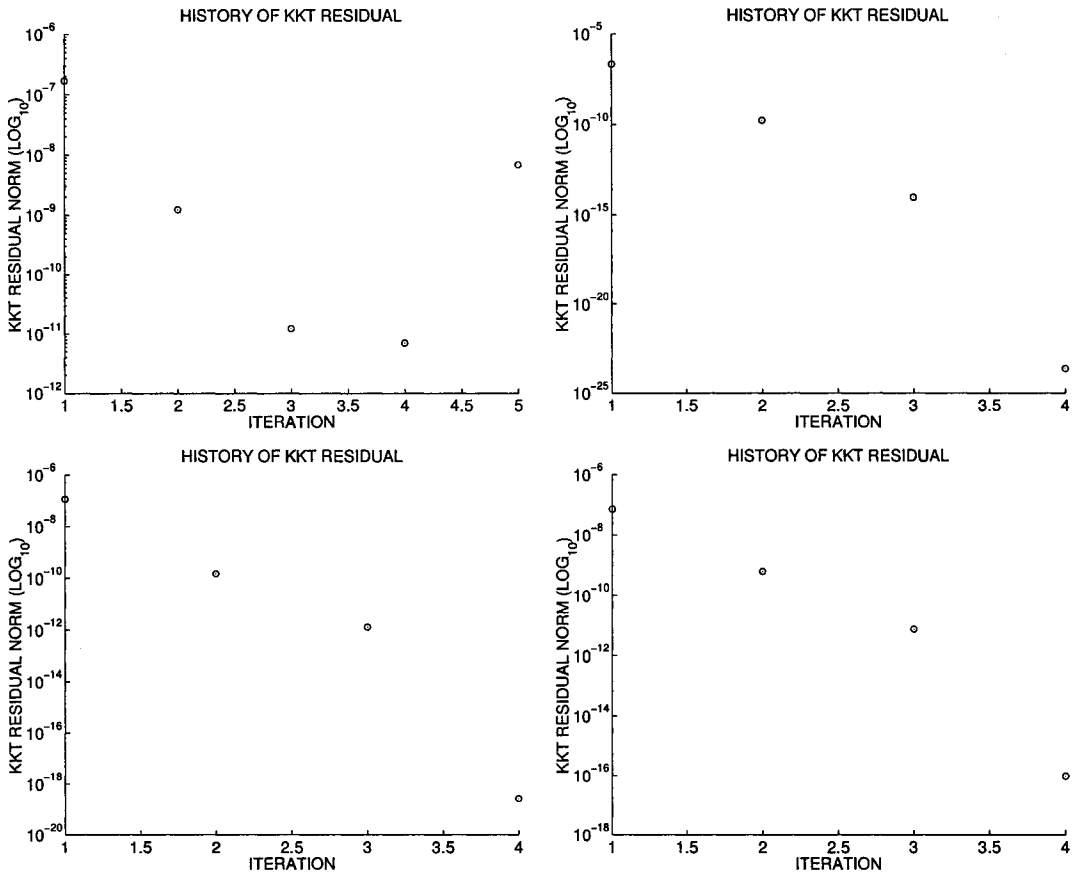

FIG. 23. *Convergence history of the residual norm for* $\gamma_k = (0.12)^k \times 10^{-4}$ *(top left), for* $\gamma = 1.0 \times 10^{-4}$ *(top right), for* $\gamma = 1.0 \times 10^{-6}$ *(bottom left), and for* $\gamma = 1.0 \times 10^{-7}$ *(bottom right) for 1% noise data with PC1, the exact Schur complements and the Tikhonov regularization.*

However, as in Figure 24 the state errors $\|u - u^*\|$, where u^* is the known state variables and u is the computed state variables, are decreasing. By using the stopping rule discussed in 4.7 with $r = 1.5$, for $\gamma_k = (0.12)^k \times 10^{-4}$ the iterations can be stopped at the second iteration, where the solution error minimum.

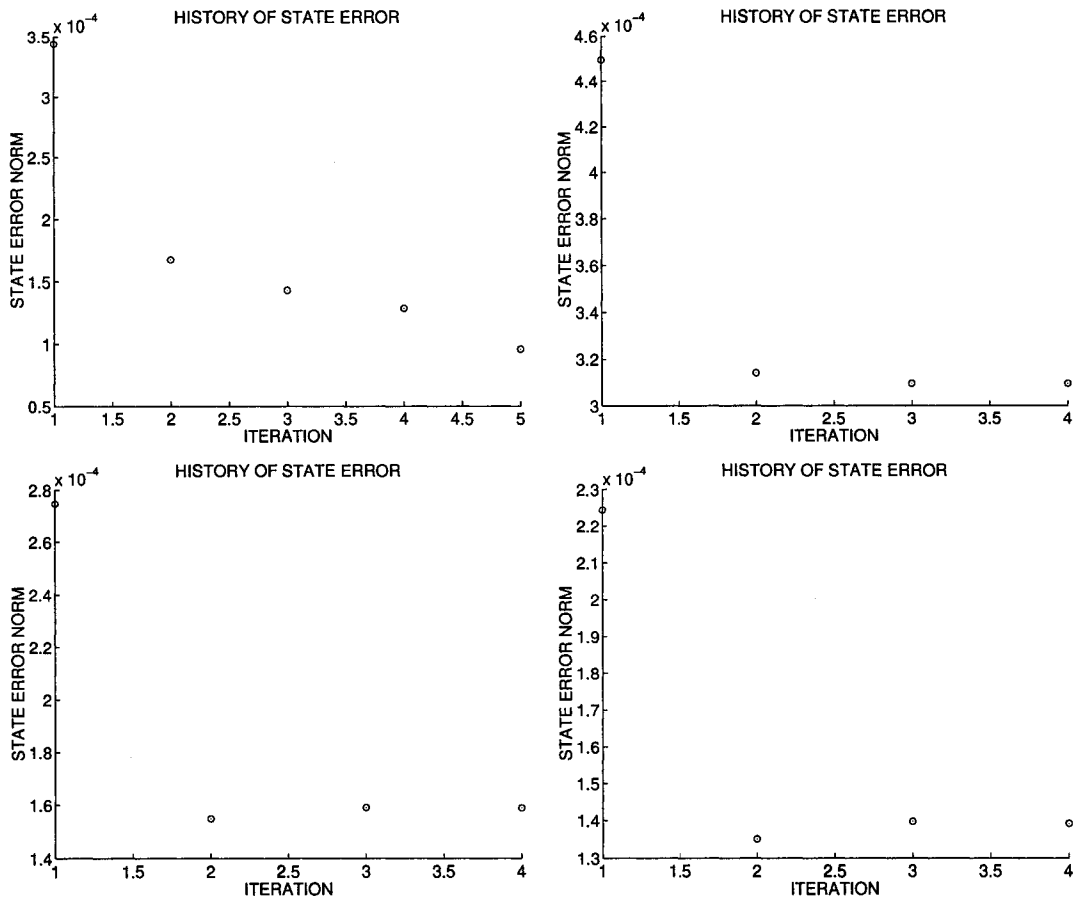

FIG. 24. *Convergence history of the state error for* $\gamma_k = (0.12)^k \times 10^{-4}$ *(top left), for* $\gamma = 1.0 \times 10^{-4}$ *(top right), for* $\gamma = 1.0 \times 10^{-6}$ *(bottom left), and for* $\gamma = 1.0 \times 10^{-7}$ *(bottom right) for* 1% *noise data with PC1, the exact Schur complements and the Tikhonov regularization.*

We see further that the solutions of the iterative Tikhonov regularization for different regularization parameters can achieve parameters closest to actual, as in Figure 25.

Other interesting results of the iterative Tikhonov regularizations are as follows:

1. The convergence histories of the solution error norms are monotonically decreasing. See Figure 26.

2. The spectra of the KKT matrices are clustered and independent of the regularization parameters. See Figure 27.

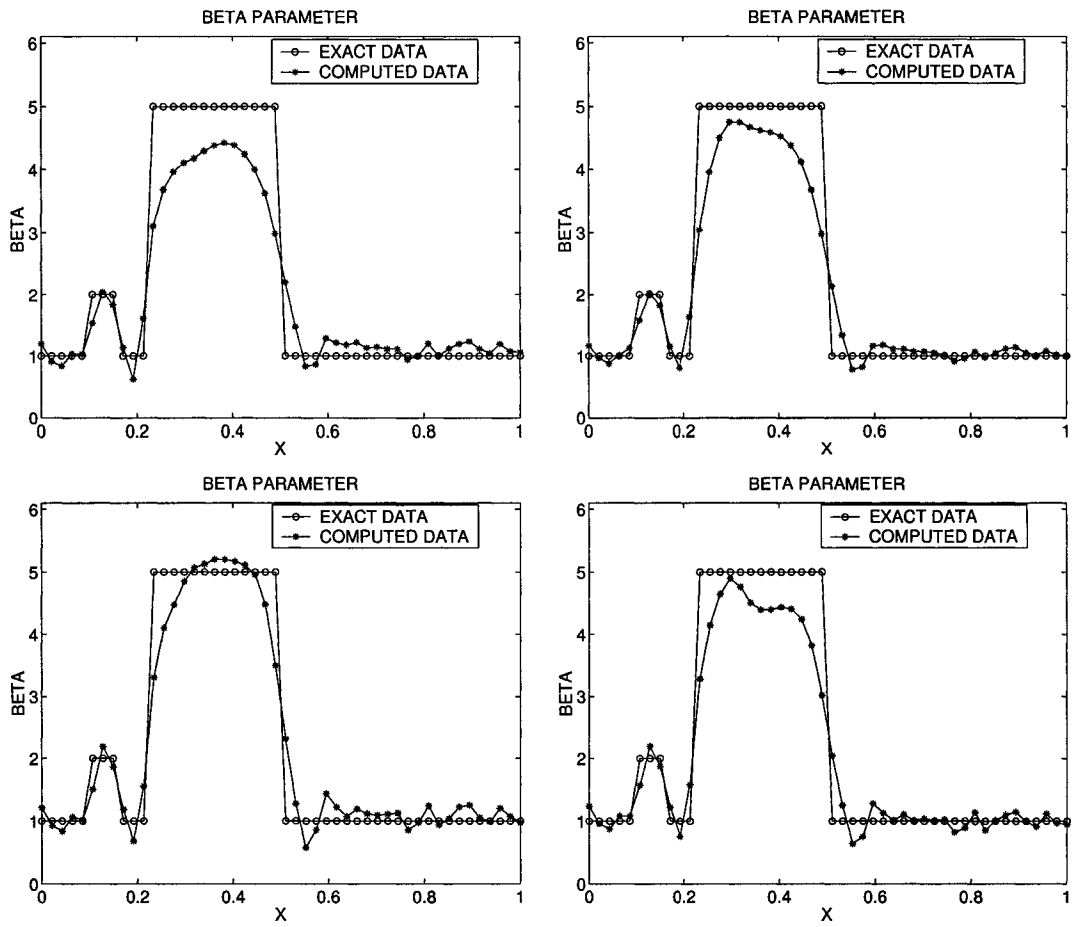

Fig. 25: Matching parameter for $\gamma_k = (0.1)^k \times 10^{-1}$ (top left), the matching parameter for $\gamma_k = (0.8)^k \times 10^{-8}$ (top right), the matching parameter for $\gamma_k = (0.9131)^k \times 10^{-9}$ (bottom left), and the matching parameter for $\gamma_k = (0.211)^k \times 10^{-5}$ (bottom right) for 1% error data with PC1 and the iterative Tikhonov regularization.

3. The convergence histories of the residual norms are non-monotone. See Figure 28. However, even though it is so, because of the stopping rule the iteration process still reaches the optimal solution. The later is demonstrated in Figure 29. Here, we can see the convergence history of the error data norms are monotonically decreasing that coincide with the history of the solution error norms as in Figure 26. Hence, if we choose the correct stopping parameters and error levels, we still can get the optimum solutions. It is also shown that by the correct choice of the iterative

reguralization parameter γ_k and the stopping rule, e.g. discrepancy principal, the method can be more robust.

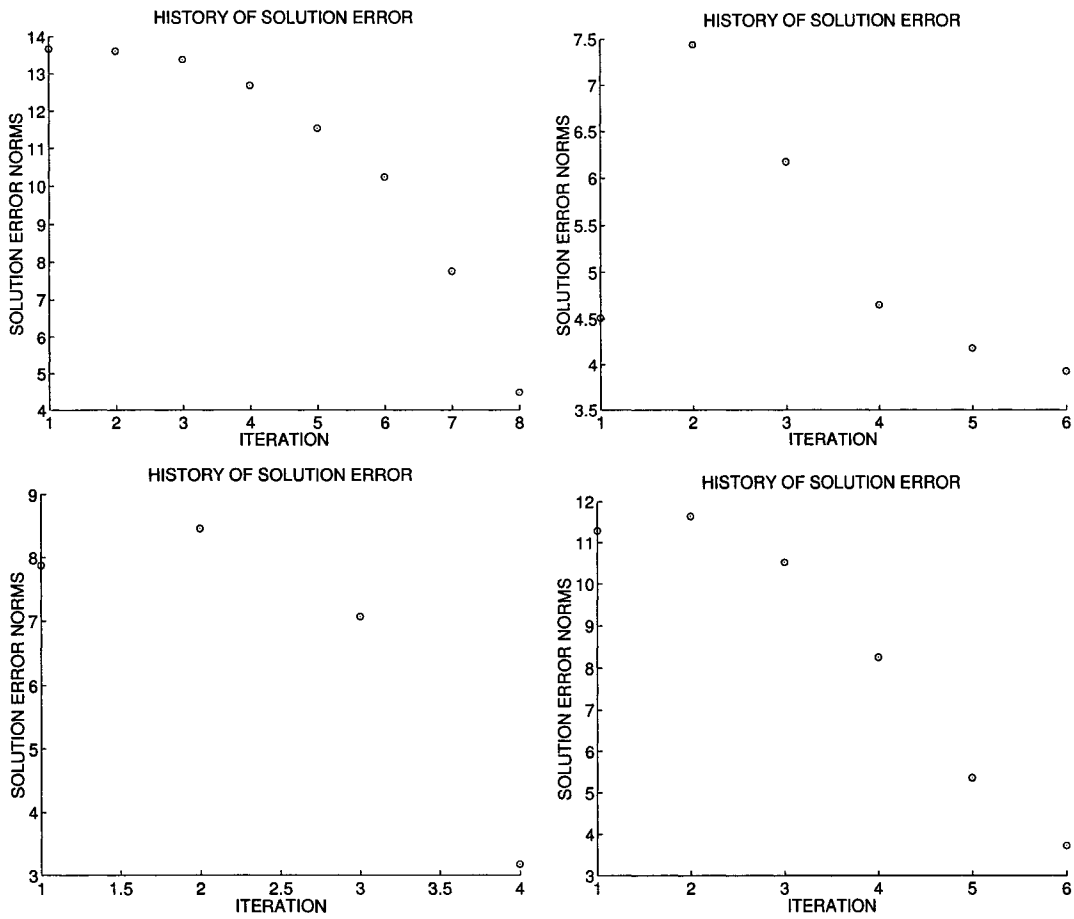

FIG. 26. *Convergence history of the solution error norm for* $\gamma_k = (0.1)^k \times 10^{-1}$ *(top left), for* $\gamma_k = (0.8)^k \times 10^{-8}$ *(top right), for* $\gamma_k = (0.9131)^k \times 10^{-9}$ *(bottom left), and for* $\gamma_k = (0.211)^k \times 10^{-5}$ *(bottom right) for* 1% *noise data with PC1, the exact Schur complements and iterative Tikhonov regularization.*

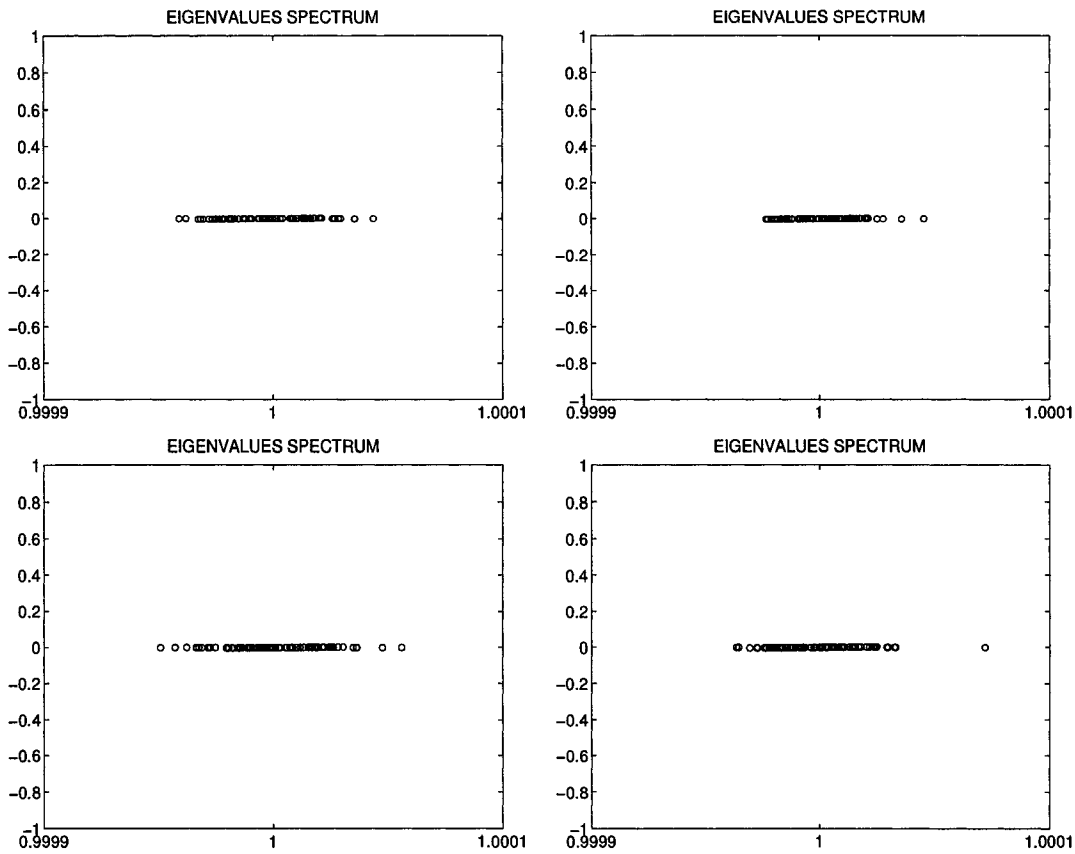

FIG. 27. *Spectrum of the KKT matrix for* $\gamma_k = (0.1)^k \times 10^{-1}$ *(top left), for* $\gamma_k = (0.8)^k \times 10^{-8}$ *(top right), for* $\gamma_k = (0.9131)^k \times 10^{-9}$ *(bottom left), and for* $\gamma_k = (0.211)^k \times 10^{-5}$ *(bottom right) for* 1% *noise data on the complex plane (axes label by real and imaginary parts) with PC1, the exact Schur complements and iterative Tikhonov regularization.*

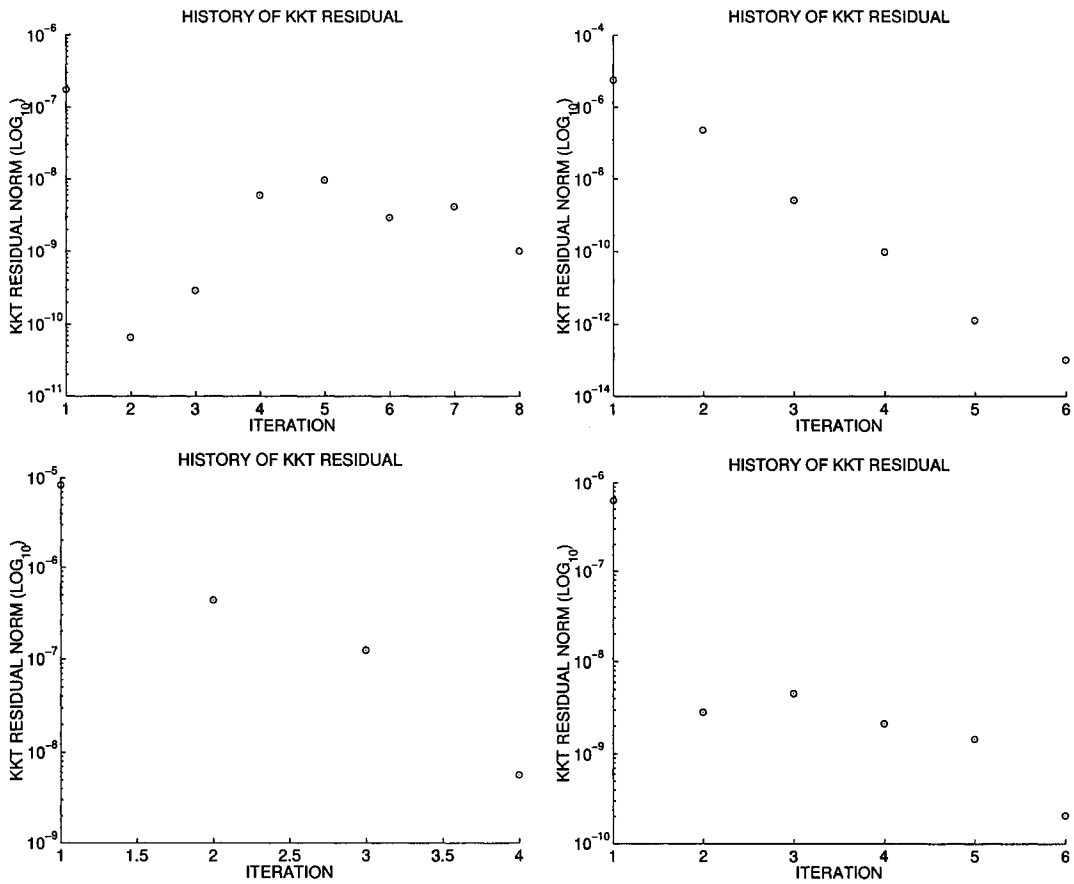

FIG. 28. *Convergence history of the residual norm of gradient for* $\gamma_k = (0.1)^k \times 10^{-1}$ *(top left), for* $\gamma_k = (0.8)^k \times 10^{-8}$ *(top right), for* $\gamma_k = (0.9131)^k \times 10^{-9}$ *(bottom left), and for* $\gamma_k = (0.211)^k \times 10^{-5}$ *(bottom right) for* 1% *noise data with PC1, the exact Schur complements and iterative Tikhonov regularization.*

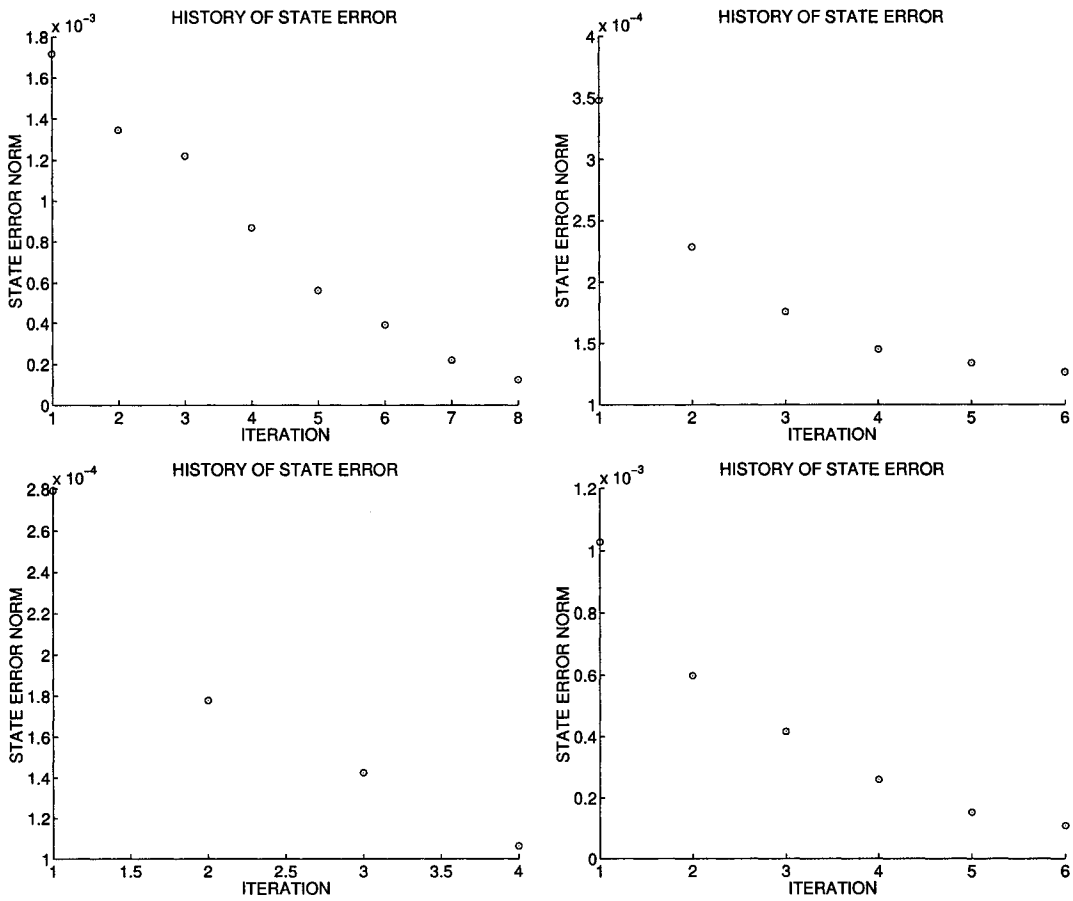

FIG. 29. *Convergence history of the state error norm for* $\gamma_k = (0.1)^k \times 10^{-1}$ *(top left), for* $\gamma_k = (0.8)^k \times 10^{-8}$ *(top right), for* $\gamma_k = (0.9131)^k \times 10^{-9}$ *(bottom left), and for* $\gamma_k = (0.211)^k \times 10^{-5}$ *(bottom right) for* 1% *noise data with (PC1, the exact Schur complements and iterative Tikhonov regularization.*

CHAPTER 5

IMPLEMENTATION IN PETSC AND ADIC

In this chapter we discuss how the algorithm is implemented in the parallel software PDE solver PETSc and the automatic differentiation software ADIC. We discuss how the data structures and the components of the PDE solver and the PDE-constrained optimization algorithm can be matched. Also, we discuss how the ADIC can be used to form the derivative of the subroutine of the function code.

5.1 PETSC

The Portable, Extensible Toolkit for Scientific Computing (PETSc) is a rich library of routines that can be used to solve large-scale PDE problems in different parallel computers. This library is developed by a team consisting of mathematicians and computer scientists in Mathematics and Computer Science Division at Argonne National Laboratory [5].

5.2 ADIC

Automatic Differentiation in C (ADIC) is a software tool that can be used to generate a derivative code of a routine of a function in C. This tool is very useful in solving a problem that needs derivatives, especially for a large-scale nonlinear PDE and a nonlinear optimization, since a hand-coding is error-prone. This software is also developed by a team at Argonne National Laboratory [46].

5.3 INTEGRATION OF PETSC AND ADIC

Recently, the ADIC and PETSc teams have worked together to allow use of ADIC within PETSc. The ADIC is the automatic differentiation tool that can be used to generate derivative codes of parallel function codes that are very difficult and complicated to code by hand. For more details about this capability see [1].

5.4 DATA STRUCTURES AND IMPLEMENTATION

In this section, we discuss the data structures of the PDE solver, the PDE-constrained optimization, and the implementation. The PETSc team has created new data structures motivated by our requirements that can be used to solve PDE-constrained optimizations. However, it is nontrivial to implement an inverse problem with this tool, since we need preconditioners and regularizations.

5.4.1 Data Structures and Subroutines for the PDE in PETSc

In PETSc the data structures are divided in two classes: distributed arrays (DA and DMMG) for structured grids and index sets (IS) for unstructured grids.

We use distributed arrays since our applications are based on the structured grids.

The subroutines we need for the PDE problems and the data structures called in them are as follows:

- `SNESCreate(comm,SNES_NONLINEAR_EQUATIONS,&snes)`
- `DACreate2d(comm,wrap,ST,M,N,m,n,dof,s,lx,ly,&user.da)`, where `comm` is MPI communicator, `wrap` is the type of the periodicity, `ST` is the stencil type, `M,N` are the global dimensions in each direction of the array, `m,n` are corresponding numbers of processors in each dimension, `dof` is number of degrees of freedom per node, `s` is stencil width, `lx,ly` are arrays containing the number of nodes in each cell along the x and y coordinates, and `user.da` is the resulting distributed array object.
- `DACreateGlobalVector(user.da,&x)`
- `DACreateLocalVector(user.da,&user.alpha)`
- `DASetLocalFunction(user.da,(DALocalFunction1)PDELocalFunction)`
- `DASetLocalJacobian(user.da,(DALocalFunction1)PDELocalJacobian)`
- `SNESSetFunction(snes,r,PDEFormFunction,&user)`
- `DASetLocalAdicFunction(user.da,ad_PDEFunctionLocalState)`
- `SNESSetJacobian(snes,A,J,SNESDACompJacobianWithAdic,&user)`

- `PDEFormInitialGuess(x,&user)`
- `SNESSolve(snes,x,&its)`

5.4.2 Data Structures and Subroutines for the PDE-constrained Optimization in PETSc

In the PDE problem we have only one variable vector. However, in the PDE-constrained optimization we have three different vectors that need to be stacked. Some of these are associated with spatial regions and can be decomposed together with spatially dependent fields for distributed-memory implementations while others are global and must be consistently replicated across distributed memories. The stacked vectors consist of the vectors of the design variable, the state variable, and the Lagrange multiplier. Since our approach is a full space method, we have a global vector that consists of these three vectors. Hence, the data structures of the optimization is different than the PDE problem.

The subroutines we need for the PDE-constrained optimization and the data structures called in them are as follows:

- `VecPackCreate(comm,&user.packer)`
- `DACreate2d(comm,wrap,ST,M,N,m,n,dof,s,lx,ly,&user.da)`, where `comm` is MPI communicator, `wrap` is the type of the periodicity, `ST` is the stencil type, `M,N` are the global dimensions in each direction of the array, `m,n` are corresponding numbers of processors in each dimension, `dof` is number of degrees of freedom per node, `s` is stencil width, `lx,ly` are arrays containing the number of nodes in each cell along the x and y coordinates, and `user.da` is the resulting distributed array object.
- `VecPackAddDA(user.packer,user.da)`
- `VecPackAddDA(user.packer,user.da)`
- `VecPackAddDA(user.packer,user.da)`
- `VecPackCreateGlobalVector(user.packer,&U)`
- `DASetLocalAdicFunction(user.da,ad_PDELocalFunction)`

- DASetLocalJacobian(user.da,(DALocalFunction1) J**),
  where J** is JacobianLocalDesign
- SNESCreate(comm,SNES_NONLINEAR_EQUATIONS,&snes)
- SNESSetFunction(snes,FU,LNKSFormFunction,&user)
- PCShellSetApply(pc,(int (*)(void*,Vec,Vec))LNKSPC,&user)
- LNKSFormInitialGuess(snes,U,&user)
- SNESSolve(snes,U,&its)

CHAPTER 6

NUMERICAL EXPERIMENTS WITH LNKSS

In this chapter, we discuss the effectiveness of the algorithm in terms of regularizations and preconditioners. We pose a synthetic two-dimensional problem, and solve it by using PETSc and ADIC. We assume that we have the data of the state variable in a square closed region and the data of the source. We would like to recover the diffusivity constant for the region. We assume that the true parameter is as shown in the top left of Figure 30 and 31.

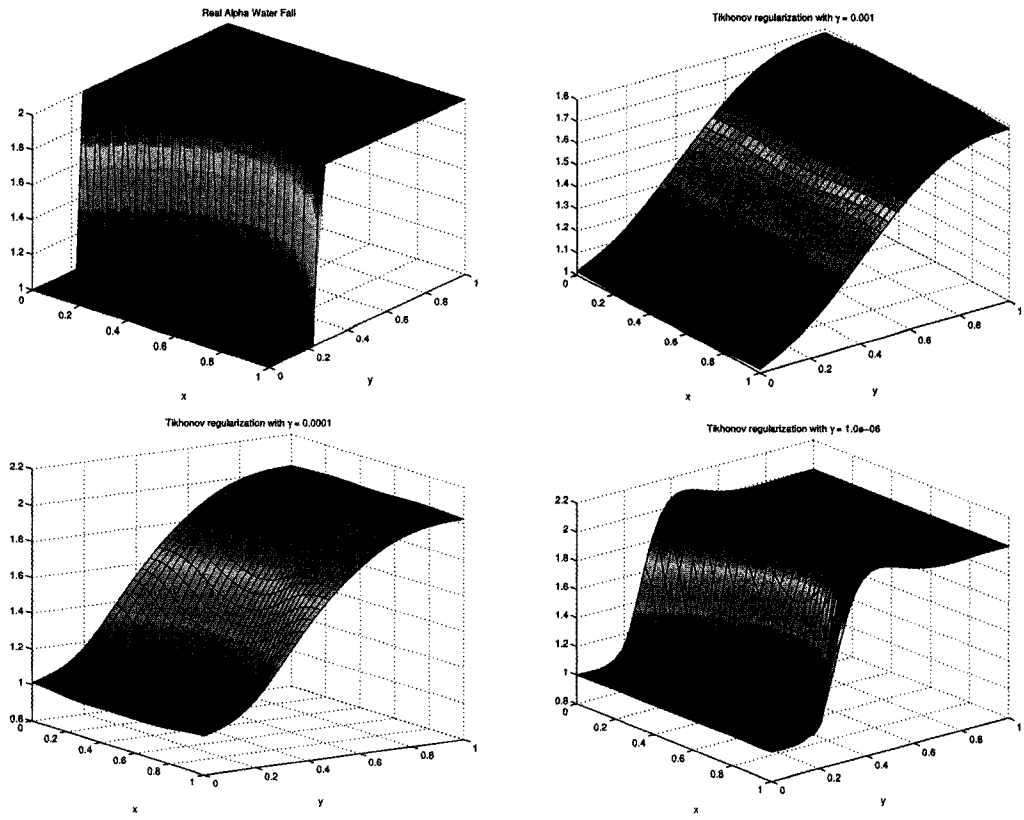

FIG. 30. *Recovered parameters with the Tikhonov regularization for exact alpha (top left), the recovered parameter for* $\gamma = 1.0 \times 10^{-3}$ *(top right), the recovered parameter for* $\gamma = 1.0 \times 10^{-4}$ *(bottom left), and the recovered parameter for* $\gamma = 1.0 \times 10^{-6}$ *(bottom right) for* 0% *noise data.*

6.1 NUMERICAL EXPERIMENTS WITH THE REGULARIZATIONS

This section focuses on the effects of the regularizations on the quality of the recovered parameters.

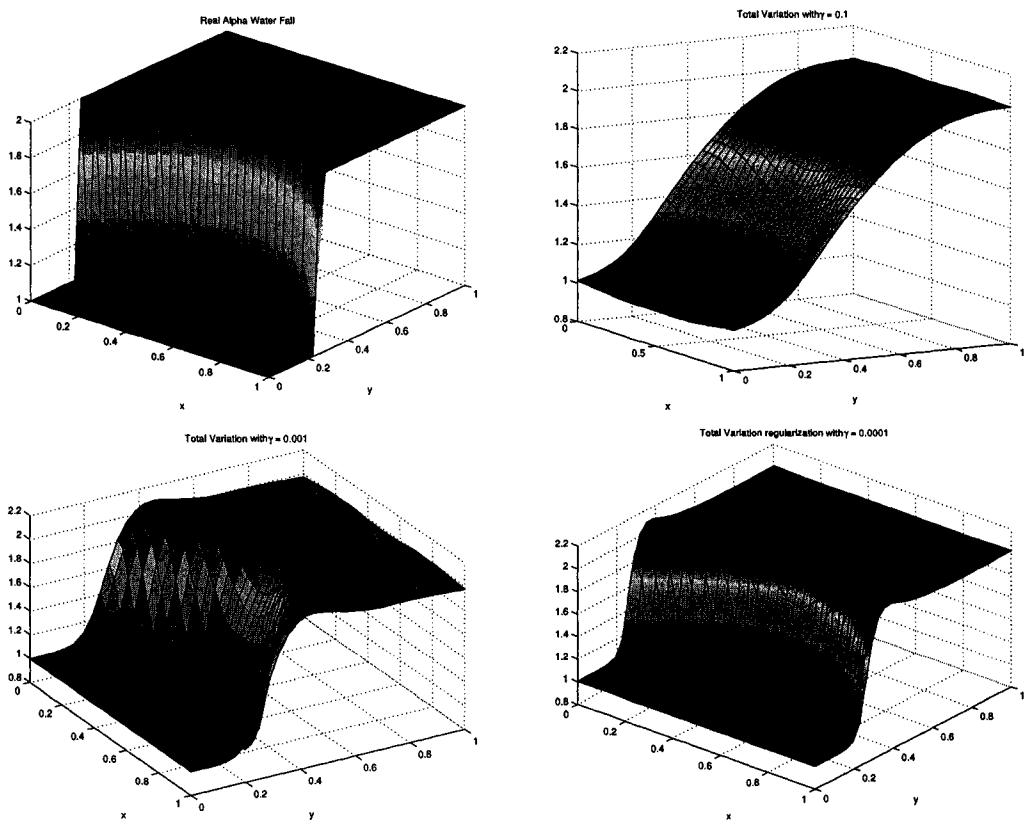

FIG. 31. *Recovered parameters with the Total Variation regularization for exact alpha (top left), the recovered parameter for* $\gamma = 0.1$ *(top right), the recovered parameter for* $\gamma = 1.0 \times 10^{-3}$ *(bottom left), and the recovered parameter for* $\gamma = 1.0 \times 10^{-4}$ *(bottom right) for* 0% *noise data.*

6.1.1 Tikhonov Regularization

We use the Tikhonov functional with the derivative as in the previous chapter:

$$R(p) = \frac{1}{2}\|\nabla p\|^2. \tag{138}$$

We apply this regularization with different regularization parameters, and their effects can be seen on the Figure 30.

This example is adopted from Vogel [85].

6.1.2 Total Variation Regularization

As mentioned earlier in the two-dimensional context the approximation of the Total Variation regularization functional is

$$R(p) = \int_0^1 \int_0^1 |\nabla p| dxdy \sim \int_0^1 \int_0^1 \sqrt{|\nabla p|^2 + \beta^2} dxdy, \tag{139}$$

where β is a small positive parameter for the approximation to the Euclidean norm.

Comparing the bottom right figures of Figure 30 and Figure 31, we see that the Total Variation is better than Tikhonov regularization in terms of the recovered parameters, especially in capturing the edge of the discontinuity. In addition, the Total Regularization seems less sensitive to the choice of the regularization parameter.

In addition, we consider another two-dimensional example similar to Akcelik's example [2] as in Figure 32. The algorithm recovers this more difficult problem very well. Akcelik employes this example in wave propagation (hyperbolic inverse problem), whereas we use it in goundwater modelling (elliptic inverse problem).

The true parameter consists of circular disks. This figure is a rough figure, because the grids are 32×32.

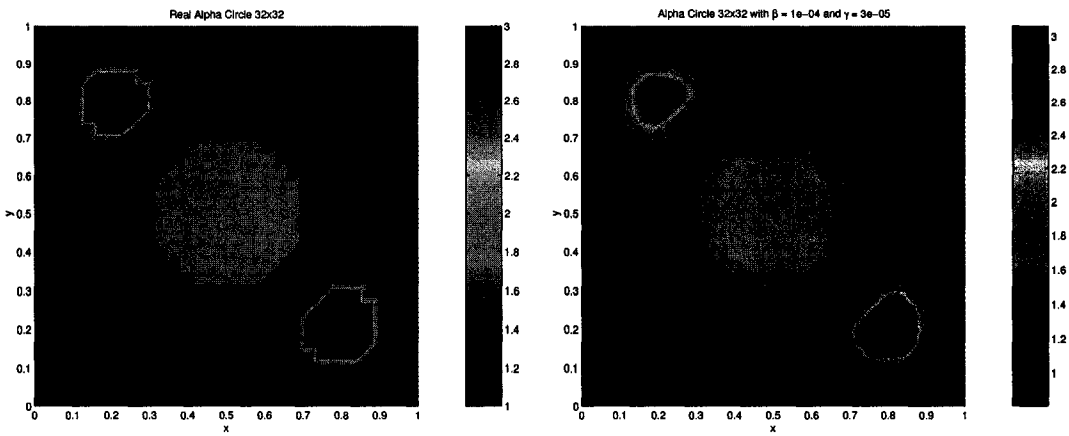

FIG. 32. *The exact parameter (left) and the recovered parameter (right) with Total Variation regularization for* $\gamma = 3.0 \times 10^{-5}$ *and* 0% *noise data.*

CHAPTER 7

CONCLUSIONS AND FUTURE DIRECTIONS

Inverse problems are difficult due to ill-posedness. Hence, we need regularizations. Choosing good regularizations is also difficult, since regularization is problem-dependent. The problems are even more difficult due to the difficulties in selecting the optimal regularization parameter. From our numerical test cases, our conclusions regarding regularizations are as follows:

- Our proposed iterative regularizations are competitive to the standard regularizations.
- In the exact data case or 0% noise data and the parameters with some jump discontinuities the Total Variation regularizations are better in capturing the discontinuity edge of the parameters than the Tikhonov regularizations. However, with noisy data the Tikhonov regularizations are more robust systems than the Total Variation regularizations. Hence, in the noisy data case the Tikhonov regularizations are better than the Total Variation regularizations in recovering the parameters. In practice most likely we will encounter noisy data.
- With noisy data another stopping rule based on the discrepancy principle is needed besides the minimum residual. Otherwise, the iteration can not reach the closest solution to the true parameter. This stopping rule also acts *a posteriori* as a selection rule for the regularization parameter.

Furthermore, the KKT, linear systems that arise in PDE-constrained optimizations are ill-conditioned and indefinite. Hence, good preconditioners and solvers are required. Preconditioners based on Schur complement preconditioners are difficult to form and compute. Hence, good approximations of the Schur complements are necessary.

From our numerical study of the preconditioners, we observe that:

- With exact data the exact Schur complement preconditioners are better in all computational aspects than the approximations. However, with noisy data the differences are not clearly apparent. Hence, in practice we can use the Schur approximations for real life problems.

- The nonlinear preconditioners defined in Subsection 3.3 are better than the linear one in solving inverse problems of nonlinear elliptic PDEs as we can see by comparing Figure 15, and we can also see from the results in Subsection 3.7.

- The unsymmetric KKT systems and preconditioners are competitive to the symmetric ones as illustrated in Table 4 in terms of execution time. The unsymmetric KKT systems and preconditioners can lead to more favorable results than the symmetric ones in terms of the iterative regularization as in Table 5 and in terms of the KKT matrix spectrum (and also iteration counts) as in Figure 16. Another advantage is that the nonsymmetric cases do not need to have symmetric preconditioners. The advantages of being symmetric can be lost due to being severely indefinite, as discussed in Section 3.8.

- Our algorithms with both Tikhonov and Total Variation regularizations are constructed to be applicable in parallel setting through the paradigm of domain decomposition of field variables and consistent replication of scalar parameters that do not uniquely belong to any single subdomain.

In the sense of solution error norms that are small and recovered parameters that are close to the exact solution (e.g., Figure 32), we also conclude that our algorithm is robust.

We propose some future directions for our research as follows:

- Our approach has been implementable with restricted one-level Schwarz method (as the default Schwarz method in PETSc for solving PDEs). It can be extended to multilevel Schwarz preconditioners by using approach similar to those in solving the PDE done with the multilevel methods.

- In one-dimensional and two-dimensional test cases our algorithm is competitive to or even better than well-established PDE-constrained optimization methods. There is no dimension-dependence in our logarithms. Therefore, they can be implemented in three dimensions, where we expect them to be even more effective relative to standard methods that satisfy the PDE constraints at each optimization step.

REFERENCES

[1] J. ABATE, S. BENSON, L. GRIGNON, P. HOVLAND, L. MCINNES, AND B. NORRIS, *Integrating Automatic Differentiation with Object-Oriented Toolkits for High-Performance Scientific Computing*, In George Corliss, Christele Fuare, Andreas Griewank, Laurent Hascoet, and Uwe Naumann, eds., *Automatic Differentiation of Algorithms: From Simulation to Optimization*, Springer, 2000, pp. 173-178.

[2] V. AKCELIK, *Multiscale Newton-Krylov Methods for Inverse Acoustic Wave Propagation*, Ph.D. Thesis, Department of Civil and Environmental Engineering, Carnegie Mellon University, 2002.

[3] V. AKCELIK, B. BIROS, AND O. GHATTAS, *Parallel Multiscale Gauss-Newton-Krylov Methods for Inverse Wave Propagation*, Proceedings of the IEEE/ACM SC2002 Conference, Baltimore, November 2002.

[4] G. ALESSANDRINI, E. ROSSET, AND J. K. SEO, *Optimal Size Estimates for The Inverse Conductivity Problem with One Measurement*, Proc. Amer. Math. Soc., 128 (1999), pp. 53-64.

[5] S. BALAY, K. BUSCHELMAN, W. D. GROPP, D. K. KAUSHIK, M. KNEPLEY, L. C. MCINNES, B. F. SMITH, AND H. ZHANG, *The Portable, Extensible Toolkit for Scientific Computing PETSc Users Manual*, Technical Report ANL-95/11-Revision 2.1.3, Argonne National Laboratory, May 2002. See http://www-unix.mcs.anl.gov/petsc/petsc-current/docs/manual.pdf.

[6] H. T. BANKS AND K. KUNISCH, *Estimation Techniques for Distributed Parameter Systems*, Birkhauser, 1989.

[7] R. BARRETT, M. BERRY, T. F. CHAN, J. DEMMEL, J. DONATO, J. DONGARRA, V. EIJKHOUT, R. POZO, C. ROMINE AND H. VAN DER VORST, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM, 1994.

[8] A. BATTERMANN AND E. W. SACHS, *Block Preconditioners for KKT systems in PDE-governed optimal control problems*, In H. K. Hoffmann, R. H. Hoppe, and V. Schulz, eds., *Fast solution of discretized optimization problems*, Workshop held at the Weierstrass Institute for Applied Analysis and Stochastics, Berlin, Internat. Series Numer. Math., vol. 138, Birkhauser, 2001, pp. 1-18.

[9] M. BENZI, G. H. GOLUB, AND J. LIESEN, *Numerical solutions of saddle point problems*, Acta Numerica, 14 (2005), pp. 1-137.

[10] L. T. BIEGLER, O. GHATTAS, M. HEINKENSCHLOSS, B. VAN BLOEMEN WAANDERS, *Large-Scale PDE-Constrained Optimization*, Springer, 2003.

[11] G. BIROS, *Parallel Algorithms for PDE-Constrained optimization and applications to optimal control of viscous flows*, Ph.D. Thesis, Department of Civil and Environmental Engineering, Carnegie Mellon University, 2000.

[12] G. BIROS AND O. GHATTAS, *A Lagrange-Newton-Krylov-Schur method for PDE-constrained optimization*, SIAG/OPT *Views-and-News*, 2 (2000), pp. 1-6.

[13] G. BIROS AND O. GHATTAS, *Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization. Part I: the Krylov-Schur solver*, SIAM J. Sci. Comput., 27 (2005), pp. 687-713.

[14] G. BIROS AND O. GHATTAS, *Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization. Part II: the Lagrange-Newton solver, and its application to optimal control of steady viscous flows*, SIAM J. Sci. Comput., 27 (2005), pp. 714-739.

[15] P. BJORSTAD, M. ESPEDAL, AND D. KEYES, eds., *Proceedings of Ninth International Conference on Domain Decomposition Methods for Partial Differential Equations*, Ullensvang, 1997. Wiley, 1999.

[16] M. BURGER AND W. MUHLHUBER *Iterative regularization of parameter identification problems by sequential quadratic programming methods*, Inverse Problems, 18 (2002), pp. 943-969.

[17] M. BURGER AND W. MUHLHUBER, *Numerical Approximation of an SQP-type method for parameter identification*, SIAM J. Numer. Anal., 40 (2002), pp. 1775-1797.

[18] X. C. CAI AND D. E. KEYES, *Nonlinearly Preconditioned Inexact Newton Algorithms*, SIAM J. Sci. Comp., 24 (2002), pp. 183-200.

[19] X. C. CAI AND M. SARKIS, *A Restricted Additive Schwarz Preconditioner for General Sparse Linear Systems*, SIAM J. Sci. Comput., 21 (1999), pp. 792-797.

[20] T. F. CHAN, R. GLOWINSKI, J. PERIAUX, AND O. B. WIDLUND, eds., *Domain Decomposition Methods: Proceedings of the Second International Symposium on Domain Decomposition Methods*, Los Angeles, SIAM, 1989.

[21] T. CHAN, R. GLOWINSKI, J. PERIAUX, AND O. B. WIDLUND, eds., *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, Houston, SIAM, 1990.

[22] T. CHAN, T. KAKO, H. KAWARADA, AND O. WIDLUND, eds., *Proceedings of Twelfth International Conference on Domain Decomposition Methods for Partial Differential Equations*, Chiba, DDM.org, 2001.

[23] T. F. CHAN AND D. E. KEYES, *Interface Preconditionings for Domain-Decomposed Convection-Diffusion Operators*, In Tony F. Chan, Rowland Glowinski, Jacques Peiaux, and Olof. B. Widlund, eds., *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, 1990, pp. 245-262.

[24] G. CHAVENT, G. PAPANICOLAOU, P. SACKS, AND W. SYMES, *Inverse Problems in Wave Propagation*, Springer, 1997.

[25] T. COLEMAN AND V. VERMA, *Structure and efficient Hessian calculation*, In Y. X. Yuan, eds., *Proceedings of the '96 International Conference on Advances in Nonlinear Programming*, Kluwer Academic, pp. 57-72.

[26] T. COLEMAN AND V. VERMA, *The efficient computation of sparse Jacobian matrices using automatic differentiation*, SIAM J. Sci. Comput., 19 (1998), pp. 1210-1233.

[27] T. COLEMAN AND V. VERMA, *ADMAT: An automatic differentiation toolbox for MATLAB*, In Proceedings of the SIAM Workshop on Object Oriented Methods for Inter-Operable Scientific and Engineering Computing, SIAM, 1998.

[28] T. Coleman and V. Verma, *ADMIT-1: Automatic Differentiation and MATLAB Interface Toolbox*, ACM Transactions on Mathematical Software, 26 (2000), pp. 150-175.

[29] M. Di Cristo and L. Rondi, *Examples of exponential instability for elliptic inverse problems*, preprint arXiv:math.AP/0303126 (2003).

[30] M. Debit, M. Garbey, R. Hoppe, D. Keyes, Y. Kuznetsov, and J. Periaux, eds., *Proceedings of Thirteenth International Conference on Domain Decomposition Methods for Partial Differential Equations*, Lyon, 2000. CINME, 2002.

[31] R. S. Dembo, S. C. Eisenstat and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400-4008.

[32] J. E. Dennis, Jr. and R. M. Lewis, *A Comparison of Nonlinear Programming Approaches to an Elliptic Inverse Problem and a New Domain Decomposition Approach*, CRPC-TR94468, 1994.

[33] P. Deuflhard, H. W. Engl, and O. Scherzer, *A convergence analysis of iterative methods for the solution of non-linear ill-posed problems under affinely invariant conditions*, Inverse Problems, 14 (1998), pp. 1081-1106.

[34] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer, 1996.

[35] R. W. Freund and N. M. Nachtigal, *Software for simplified Lanczos and QMR algorithms*, Appl. Num. Math., 19 (1995), pp. 319-341.

[36] R. Glowinski, G. H. Golub, G. A. Meurant, and J. Periaux, eds., *Domain Decompotion Methods for Partial Differential Equations, Proceedings of the First International Symposium on Domain Decomposition Methods for Partial Diferential Equations*, Paris, SIAM, 1988.

[37] R. Glowinski, Y. A. Kuznetsov, G. A. Meurant, J. Periaux, and O. B. Widlund, eds., *Fourth International Symposium on Domain Decomposition Methods for Patial Differential Equations*, Moscow, SIAM, 1991.

[38] R. GLOWINSKI, J. PERIAUX, Z.-C. SHI, AND O. WIDLUND, eds., *Proceedings of Eight International Conference on Domain Decomposition Methods for Partial Differential Equations*, Beijing, Wiley, 1997.

[39] G. E. GOLUB AND C. GREIF, *On solving block-structured indefinite linear systems*, SIAM J. Sci. Comput., 24 (2003), pp. 2076-2092.

[40] J. GOTTLIEB AND P. DUCHATEAU, *Parameter Identification and Inverse Problems in Hydrology, Geology, and Ecology*, Kluwer, 2002.

[41] A. GREENBAUM, *Iterative Methods for Solving Linear Systems*, SIAM, 1997.

[42] E. HABER AND U. M. ASCHER, *Preconditioned all-at-once methods for large, sparse parameter estimation problems*, Inverse Problems, 17 (2001), pp. 1847-1864.

[43] J. HADAMARD, *Lectures on Cauchy's problem in linear partial differential equations*, Dover, 1952.

[44] S. B. HAZRA, H. CLASS, R. HELMIG, AND V. SCHULZ, *Forward and inverse problems in modeling of multiphase flow and transport through porous media*, Computational Geosciences, 8 (2004), pp. 21-47.

[45] I. HERRERA, D. KEYES, O. WIDLUND, AND R. YATES, eds., *Proceedings of Fourteenth International Conference on Domain Decomposition Methods in Science and Engineering*, Coyoco, 2002. UNAM, 2003.

[46] P. D. HOVLAND AND B. NORRIS, *Users' Guide to ADIC 1.1*, Argonne Technical Memorandum ANL/MCS-TM-225, Argonne National Laboratory, 2000. See http://www-unix.mcs.anl.gov/autodiff/ADIC.

[47] V. ISAKOV, *Inverse Problems for Partial Differential Equations*, Springer, 1998.

[48] K. ITO AND K. KUNISCH, *The Augmented Lagrangian Method for Parameter Estimation in Elliptic Systems*, SIAM J. Control Optim., 28 (1990), pp. 113-136.

[49] K. ITO, M. KROLLER, AND K. KUNISCH, *A Numerical Study of An Augmented Lagrangian Method for The Estimation of Parameters in Elliptic Systems*, SIAM J. Sci. Stat. Compt., 12 (1991), pp. 884-910.

[50] C. T. KELLEY, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, 1995.

[51] C. T. KELLEY, *Solving Nonlinear Equations with Newton's Method*, SIAM, 2003.

[52] D. E. KEYES AND W. D. GROPP, *A Comparison of Domain Decomposition techniques for Elliptic Partial Differential Equations and their Parallel Implementation*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. 166-202.

[53] D. E. KEYES, T. F. CHAN, G. A. MEURANT, J. S. SCROGGS, AND R. VOIGT, eds., *Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations*, Norfolk, SIAM, 1992.

[54] D. E. KEYES AND J. XU, eds., *Domain Decomposition Methods in Science and Engineering: Proceedings of the Seventh International Conference on Domain Decomposition*, State College, AMS, 1995.

[55] D. E. KEYES, P. D. HOVLAND, L. C. MCINNES, AND W. SAMYONO, *Using Automatic Differentiation for Second-Order Matrix-free Methods in PDE-constrained Optimization*, in Proceedings of Automatic Differentiation 2000, Springer, 2000, pp. 33-50.

[56] A. KIRSCH, *An Introduction to the Mathematical Theory of Inverse Problems*, Springer, 1996.

[57] D. A. KNOLL AND D. E. KEYES, *Jacobian-free Newton-Krylov Methods: A Survey of Approaches and Applications*, J. Comp. Phys., 193 (2004), pp. 357-397.

[58] I. KNOWLES, *Uniqueness for An Elliptic Inverse Problem*, SIAM J. Appl. Math., 59 (1999), pp. 1356-1370.

[59] R. KORNHUBER, R. HOPPE, J. PERIAUX, O. PIRONNEAU, O. B. WIDLUND, AND J. XU, eds., *Proceedings of Fifteenth International Conference on Domain Decomposition Methods in Science and Engineering*, Berlin, Springer, 2004.

[60] K. KUNISCH AND E. W. SACHS, *Reduced SQP Methods for Parameter Identification Problems*, SIAM J. Numer. Anal., 29 (1992), pp. 1793-1820.

[61] C.-H. LAI, P. BJORSTAD, M. CROSS, AND O. WIDLUND, eds., *Proceedings of Eleventh International Conference on Domain Decomposition Methods for Partial Differential Equations*, Greenwich, DDM.org, 2000.

[62] L. LANDWEBER, *An iteration formula for Fredholm integral equations of the first kind*, Amer. J. Math., 73 (1951), pp. 615-624.

[63] J. MANDEL, C. FARHAT, AND X.-C. CAI, eds., *Proceedings of Tenth International Conference on Domain Decomposition Methods for Partial Differential Equations*, Boulder, AMS, 1999.

[64] The MathWorks. Matlab 5.3.1, 2000. See www.mathworks.com/products/matlab

[65] V. A. MOROZOV, *Methods for Solving Incorrectly Posed Problems*, Springer, 1984.

[66] M. F. MURPHY, G. E. GOLUB, AND A. J. WATHEN, *A Note on Preconditioning for Indefinite Linear Systems*, SIAM J. Sci. Comput., 21 (2000), pp. 1969-1972.

[67] N. M. NACHTIGAL, L. REICHEL, AND L. N. TREFETHEN, *How fast are nonsymmetric matrix iterations?*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 778-795.

[68] S. P. NEUMAN, *Calibration of Distributed parameter ground-water flow models viewed as a multiple-objective decision process under uncertainty*, Water. Resour. Res., 9 (1973), pp. 1006-1021.

[69] J. NOCEDAL AND S. J. WRIGHT, *Numerical Optimization*, Springer, 1999.

[70] D. L. PHILLIPS, *A technique for the numerical solution of certain integral equations of the first kind*, J. Assoc. Comput. Mach., 9 (1962), pp. 84-97.

[71] A. QUARTERONI, Y. A. KUZNETSOV, J. PERIAUX, AND O. B. WIDLUND, eds., *Domain Decomposition Methods in Science and Engineering: The Sixth International Conference on Domain Decomposition*, Como, Italy, AMS, 1994.

[72] A. QUARTERONI AND A. VALLI, *Domain Decomposition Methods for Partial Differential Equations*, Oxford, 1999.

[73] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, Second Edition, Retrieved June 23, 2005 from http://www-users.cs.umn.edu/ saad/books.html

[74] Y. SAAD AND M. SCHULTZ, *GMRES: A generalized minimal residual algorithm for solving nosymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1985), pp. 417-424.

[75] Y. SAAD AND H. A. VAN DER VORST, *Iterative solution of linear systems in the 20th century*, J. of Comp. Appl. Math., 123 (2000), pp. 1-33.

[76] O. SHCERZER, *A Modified Landweber Iteration for Solving Parameter Estimation Problems*, Appl. Math. Optim., 38(1998), pp. 45-68.

[77] H. A. SCHWARZ, *Gesammelte Mathematische Abhandlungen*, volume 2 (1890), Springer, Berlin, pp. 133-143. First published in *Vierteljahrsschrift Naturforsch. Ges. Zurich*, 15 (1870), pp. 272-286.

[78] B. F. SMITH, P. E. BJϕRSTAD, AND W. D. GROPP, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1999.

[79] E. DE STURLER AND J. LIESEN, *Block-diagonal and constraint Preconditioners for Nonsymmetric Indefinite Linear Systems*, SIAM J. Sci. Comput., 26 (2005), pp. 1598-1619.

[80] N.-Z. SUN, *Inverse Problems in Groundwater Modeling*, Kluwer, 1994.

[81] A. N. TIKHONOV, *Regularization of incorrectly posed problems*, Soviet Math. Dokl., 4 (1963), pp. 1624-1627.

[82] A. N. TIKHONOV, *Solution of incorrectly formulated problems and the regularization method*, Soviet Math. Dokl., 4 (1963), pp. 1035-1038.

[83] G. M. VAINIKKO, *The discrepancy principle for a class of regularization methods*, USSR Comp. Math. Math. Phys. 22, 3 (1982), pp. 1-19.

[84] G. M. VAINIKKO, *The critical level of discrepancy in regularization methods*, USSR Comp. Math. Math. Phys. 23, 6 (1983), pp. 1-9.

[85] C. R. VOGEL, *Sparse matrix computation arising in distributed parameter identification*, SIAM J. Matrix Anal. Appl., 20, 4 (1999), pp. 1027-1037.

[86] C. R. VOGEL, *Computational Methods for Inverse Problems*, SIAM, Frontiers in Applied Mathematics, 2002.

[87] O. WIDLUND AND A. TOSELLI, *Domain Decomposition Methods: Algorithms and Theory*, Springer, 2004.

VITA

WIDODO SAMYONO

Department of Computational and Applied Mathematics
Old Dominion University
Norfolk, VA 23529

Widodo Samyono was born on June 23, 1961, in Pasuruan, East Java Province, Indonesia. He received his B. S. in Pure Mathematics in March 1986 from Sepuluh Nopember Institute of Technology, Surabaya, Indonesia. He received his M. S. in Applied Mathematics in August 1995 from Hampton University, Hampton, Virginia, USA.

Typeset using LaTeX.