An Overlapping Domain Decomposition Method for Parameter Identification Problems*

Xiao-Chuan Cai[1], Si Liu[1], and Jun Zou[2]

[1] Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309, USA, {cai,si.liu}@cs.colorado.edu
[2] Department of Mathematics, The Chinese University of Hong Kong, Shatin N. T., Hong Kong, zou@math.cuhk.edu.hk

Summary. A parallel fully coupled one-level Newton-Krylov-Schwarz method is investigated for solving the nonlinear system of algebraic equations arising from the finite difference discretization of inverse elliptic problems. Both L^2 and H^1 least squares formulations are considered with the H^1 regularization. We show numerically that the preconditioned iterative method is optimally scalable with respect to the problem size. The algorithm and our parallel software perform well on machines with modest number of processors, even when the level of noise is quite high.

1 Introduction

We consider an inverse elliptic problem [4]: Find $\rho(x)$, such that

$$\begin{cases} -\nabla\cdot(\rho\nabla u) = f,\, x\in\Omega \\ \qquad\quad u(x) = 0,\, x\in\partial\Omega. \end{cases} \tag{1}$$

When the measurement of $u(x)$ is given, denoted as $z(x)$, the inverse problem can be transformed into a minimization problem:

$$\text{minimize}\;\; J(\rho,u) = \frac{1}{2}\int_\Omega (u-z)^2dx + \frac{\beta}{2}\int_\Omega |\nabla\rho|^2dx, \tag{2}$$

which is usually referred to as the "L^2 least squares formulation". When the measurement of $\nabla u(x)$ is given, denoted as $\nabla z(x)$, the inverse problem can be transformed into another minimization problem:

$$\text{minimize}\;\; J(q,v) = \frac{1}{2}\int_\Omega \rho\,|\nabla u - \nabla z|^2dx + \frac{\beta}{2}\int_\Omega |\nabla\rho|^2dx, \tag{3}$$

* The work of XCC and SL was partially supported by DOE DE-FC02-01ER25479, NSF ACI-0305666 and ACI-0352334, the work of JZ was substantially supported by the Hong Kong RGC grant, Project 404105.

which is usually referred to as the "H^1 least squares formulation". Both minimization problems (2) and (3) are subject to the constraint (1). We introduce the Lagrangian functional

$$\mathcal{L}(\rho, u, \lambda) = \frac{1}{2}\int_{\Omega}(u-z)^2 dx + ((\nabla\lambda, \rho\nabla u) - (\lambda, f)) + \frac{\beta}{2}\int_{\Omega}|\nabla\rho|^2 dx \tag{4}$$

for the L^2 case, and

$$\mathcal{L}(\rho, u, \lambda) = \frac{1}{2}\int_{\Omega}\rho|\nabla u - \nabla z|^2 dx + ((\nabla\lambda, \rho\nabla u) - (\lambda, f)) + \frac{\beta}{2}\int_{\Omega}|\nabla\rho|^2 dx \tag{5}$$

for the H^1 case. The solution of both minimization problems can be obtained by solving the corresponding saddle-point problem: Find (ρ, u, λ) such that

$$(\nabla_\rho\mathcal{L})p = 0, (\nabla_u\mathcal{L})w = 0, \text{ and } (\nabla_\lambda\mathcal{L})\mu = 0 \tag{6}$$

for any (p, w, μ). More explicitly, we can reduce (6) to

$$\begin{cases} -\beta\Delta\rho + \nabla u\cdot\nabla\lambda = 0 \\ -\nabla\cdot(\rho\nabla\lambda) + (u-z) = 0 \\ -\nabla\cdot(\rho\nabla u) - f = 0 \end{cases} \tag{7}$$

in the L^2 case. Similarly, in the H^1 case, we have

$$\begin{cases} -\beta\Delta\rho + \nabla u\cdot\nabla\lambda + \dfrac{1}{2}|\nabla u - \nabla z|^2 = 0 \\ -\nabla\cdot(\rho\nabla\lambda) + \nabla\cdot(\rho\nabla z) + f = 0 \\ -\nabla\cdot(\rho\nabla u) - f = 0. \end{cases} \tag{8}$$

Both systems share the same boundary conditions $\partial\rho/\partial n = 0, u = 0, \lambda = 0$ on $\partial\Omega$. The rest of the paper is devoted to a Newton-Krylov-Schwarz method for solving the algebraic systems

$$F(U) = 0$$

arising from the finite difference discretization of (7) and (8) in a fully coupled fashion.

2 Newton-Krylov-Schwarz method

The family of Newton-Krylov-Schwarz (NKS) methods ([1]) is a general-purpose parallel algorithm for solving a system of nonlinear algebraic equations. NKS has three main components: an inexact Newton's method for the nonlinear system; a Krylov subspace linear solver for the Jacobian systems (restarted GMRES[5]); and a Schwarz type preconditioner [6]. We carry out Newton iterations as following:

$$U_{k+1} = U_k - \lambda_k J(U_k)^{-1} F(U_k), \quad k = 0, 1, ... \tag{9}$$

where U_0 is an initial approximation to the solution and $J(U_k) = F'(U_k)$ is the Jacobian at U_k, and λ_k is the steplength determined by a linesearch procedure [3]. The inexactness of Newton's method is reflected in the fact that we do not solve the Jacobian system exactly. The accuracy of the Jacobian solver is determined by some $\eta_k \in [0, 1)$ and the condition

$$\|F(U_k) + J(U_k)s_k\| \le \eta_k \|F(U_k)\|. \tag{10}$$

The vector s_k is obtained by approximately solving the linear Jacobian system

$$J(U_k)M_k^{-1}(M_k s_k) = -F(U_k),$$

where M_k^{-1} is a one-level additive Schwarz right preconditioner. To formally define M_k^{-1}, we need to introduce a partition of Ω. We first partition the domain into non-overlapping substructures Ω_l, $l = 1, \cdots, N$. In order to obtain an overlapping decomposition of the domain, we extend each subregion Ω_l to a larger region Ω_l', i.e., $\Omega_l \subset \Omega_l'$. Only simple box decomposition is considered in this paper – all subdomains Ω_l and Ω_l' are rectangular and made up of integral numbers of fine mesh cells. The size of Ω_l is $H_x \times H_y$ and the size of Ω_l' is $H_x' \times H_y'$, where the H's are chosen so that the overlap, *ovlp*, is uniform in the number of fine mesh cells all around the perimeter, i.e., $ovlp = (H_x' - H_x)/2 = (H_y' - H_y)/2$ for interior subdomains. For boundary subdomains, we simply cut off the part that is outside Ω.

On each extended subdomain Ω_l', we construct a subdomain preconditioner B_l, whose elements are extracted from the matrix $J(U_k)$. Homogeneous Dirichlet boundary conditions are used on the internal subdomain boundary $\partial\Omega_l' \cap \Omega$, and the original boundary conditions are used on the physical boundary, if present. The additive Schwarz preconditioner can be written as

$$M_k^{-1} = I_1 B_1^{-1} (I_1)^T + \cdots + I_N B_N^{-1} (I_N)^T. \tag{11}$$

Let n be the total number of mesh points, and n_l' the total number of mesh points in Ω_l', then I_l is an $3n \times 3n_l'$ extension matrix that extends each vector defined on Ω_l' to a vector defined on the entire fine mesh by padding an $3n_l' \times 3n_l'$ identity matrix with zero rows. The factor of 3 is included because each mesh point has 3 unknowns.

3 Numerical experiments

We study the performance of the proposed algorithm using the following test case with the observation function given as $z(x, y) = \sin(\pi x)\sin(\pi y)$, $\Omega =$

$(0,1) \times (0,1)$, and the right-hand side f chosen so that the elliptic coefficient to be identified is $\rho = 1 + 100(xy(1-x)(1-y))^2$. To test the robustness of the algorithms, we add some noise to the observation data as

$$z^\delta = z + \delta\, rand(x, y) \tag{12}$$

or

$$\nabla z^\delta = \nabla z + \delta\, (rand(x, y), rand(x, y))^T, \tag{13}$$

depending on if the formulation is L^2 or H^1. Here $rand(x, y)$ defines a random scalar function. δ is responsible for the magnitude of the noise. Results with three different levels of noise ($\delta = 0\%, 1\%$ and 10%) will be presented. Since u needs to satisfy the elliptic equation, we assume that u and ∇u have some continuity and differentiability. Therefore, we smooth z in the L^2 formulation or ∇z in the H^1 formulation before we start the Newton iteration. This is necessary especially when the noise level is high. In particular, when the noise level is 10%, we replace the value of z or ∇z by the average value around it using the following weights

$$\begin{array}{ccccc} \frac{1}{16} & & \frac{1}{8} & & \frac{1}{16} \\ & \searrow & \downarrow & \swarrow & \\ \frac{1}{8} & \rightarrow & \frac{1}{4} & \leftarrow & \frac{1}{8} \\ & \nearrow & \uparrow & \nwarrow & \\ \frac{1}{16} & & \frac{1}{8} & & \frac{1}{16} \end{array}$$

We repeat this operation 3 times in all the experiments when $\delta = 10\%$. No smoothing is applied when δ is smaller than 10%.

To measure the accuracy of the algorithm, we assume the exact solution of the test problem is known, and $error_u$ and $error_\rho$ are the normalized discrete L^2 norms of the errors defined as

$$error_u = \sqrt{\sum (u_{ij} - u_{ij}^{exact})^2 h_x h_y} \quad \text{and} \quad error_\rho = \sqrt{\sum (\rho_{ij} - \rho_{ij}^{exact})^2 h_x h_y},$$

where h_x and h_y are mesh sizes along x and y directions, respectively.

In our experiments, we choose the stopping conditions as follows: The relative residual is less than 10^{-6} or the absolute residual is less than 10^{-10} for the nonlinear system. The relative residual is less than 10^{-6} or the absolute residual is less than 10^{-10} for each linear solve in the nonlinear iteration. In Newton's method, we use the initial guess

$$(\rho^{(0)}, u^{(0)}, \lambda^{(0)})^T = (1, z, 0)^T$$

for the L^2 formulation. For the H^1 formulation, z is obtained as an integral of $\nabla_x z$ or $\nabla_y z$ along the x or y direction from one of the boundary points. In our experiments, at the mesh point (x_i, y_j),

$$z(x_i, y_j) = z(x_0, y_j) + \sum_{l=1}^{i} (\nabla_x z)|_{x_l} h_x$$

if we take the integral along the x direction, or a similar integral along the y direction.

We first test three meshes 40×40, 80×80, and 160×160. When the Jacobian systems are solved exactly with a Gaussian elimination, the total number of Newton iterations ranges from 3 to 6, and the iteration numbers are not sensitive to the level of noise, as shown in Table 1. The exact solution, and the numerical solutions for both L^2 and H^1 formulations with 3 levels of noise are shown in Fig.1.

We next look at the performance of the algorithm, in particular, we would like to know how the convergence depends on the mesh size, the number of subdomains, and the overlapping size. We solve the problem on a 320×320 mesh using different number of processors (np), and the results, in terms of the iteration number and the total compute time, are in Table 2. The numbers of Newton iterations do not change when we change the number of processors or the overlapping size.

If we fix the number of subdomains, which is the same as the number of processors, and increase the overlapping size, the number of GMRES iterations decreases. The compute time decreases to a certain point and then begins to increase. This suggests that an optimal overlapping size exists if the objective is to minimize the total compute time when the number of processors is fixed. On a fixed mesh the number of GMRES iterations increases as we use more processors. This is expected since this is a single-level algorithm.

To check the $h-$scalability of the algorithm, we increase the mesh size and the number of processors at the same ratio in order for each processor to have the same number of mesh points. Table 3 shows the results with different mesh sizes for np=4, 16 and 64. Both the number of Newton iterations and the number of GMRES iterations are almost constants when the number of processors is fixed.

4 Final remarks

We developed a fully parallel domain decomposition method for solving the system of nonlinear equations arising from the fully coupled finite difference discretization of some inverse elliptic problems. Traditionally this type of problems are solved by using Uzawa type of algorithms which split the system into two or three subsystems and each subsystem is solved individually. Subiterations are required between the subsystems. The subsystems are easier to solve than the global coupled system, but the iterations between subsystems are sequential in nature. The focus of this paper was to investigate a fully coupled approach without splitting the system into subsystems. Such an approach is more parallel than the splitting method. We showed numerically that with a powerful domain decomposition based preconditioner the convergence of the iterative methods can be obtained even for some difficult cases when the observation data has high level of noise. More details of the work will be included in a forthcoming paper [2].

Table 1. Errors and the number of Newton iterations for three different meshes and with different levels of noise.

		$error_u$	$error_\rho$	Newton
L^2 formulation	$\beta = 10^{-6}, \delta = 0$	0.000078	0.003163	3
40×40	$\beta = 10^{-5}, \delta = 1\%$	0.000765	0.010723	3
	$\beta = 10^{-4}, \delta = 10\%$	0.008222	0.038667	3
L^2 formulation	$\beta = 10^{-6}, \delta = 0$	0.000073	0.003177	3
80×80	$\beta = 10^{-5}, \delta = 1\%$	0.000532	0.010070	3
	$\beta = 10^{-4}, \delta = 10\%$	0.003849	0.029056	3
L^2 formulation	$\beta = 10^{-6}, \delta = 0$	0.000072	0.003203	3
160×160	$\beta = 10^{-5}, \delta = 1\%$	0.000504	0.009908	3
	$\beta = 10^{-5}, \delta = 10\%$	0.002064	0.026190	4
H^1 formulation	$\beta = 10^{-5}, \delta = 0$	0.000362	0.001744	6
40×40	$\beta = 10^{-5}, \delta = 1\%$	0.000355	0.006010	6
	$\beta = 10^{-4}, \delta = 10\%$	0.006980	0.022837	5
H^1 formulation	$\beta = 10^{-5}, \delta = 0$	0.000090	0.000406	4
80×80	$\beta = 10^{-5}, \delta = 1\%$	0.000109	0.003842	4
	$\beta = 10^{-4}, \delta = 10\%$	0.001921	0.011741	4
H^1 formulation	$\beta = 10^{-5}, \delta = 0$	0.000023	0.000187	3
160×160	$\beta = 10^{-5}, \delta = 1\%$	0.000030	0.002580	4
	$\beta = 10^{-4}, \delta = 10\%$	0.000473	0.007419	5

References

[1] X.-C. Cai, W. Gropp, D. Keyes, R. Melvin, and D. P. Young, *Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equation*, SIAM J. Sci. Comput., 19 (1998), pp. 246–265

[2] X.-C. Cai, S. Liu, and J. Zou, *Parallel fully coupled algorithms for inverse elliptic problems*, in preparation

[3] J. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, Philadelphia, 1996

[4] Y. L. Keung and J. Zou, *An efficient linear solver for nonlinear parameter identification problems*, SIAM J. Sci. Comput. 22 (2000), pp. 1511–1526

[5] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, 2003

[6] A. Toselli and O. Widlund, *Domain Decomposition Methods – Algorithms and Theory*, Springer, Berlin, 2005

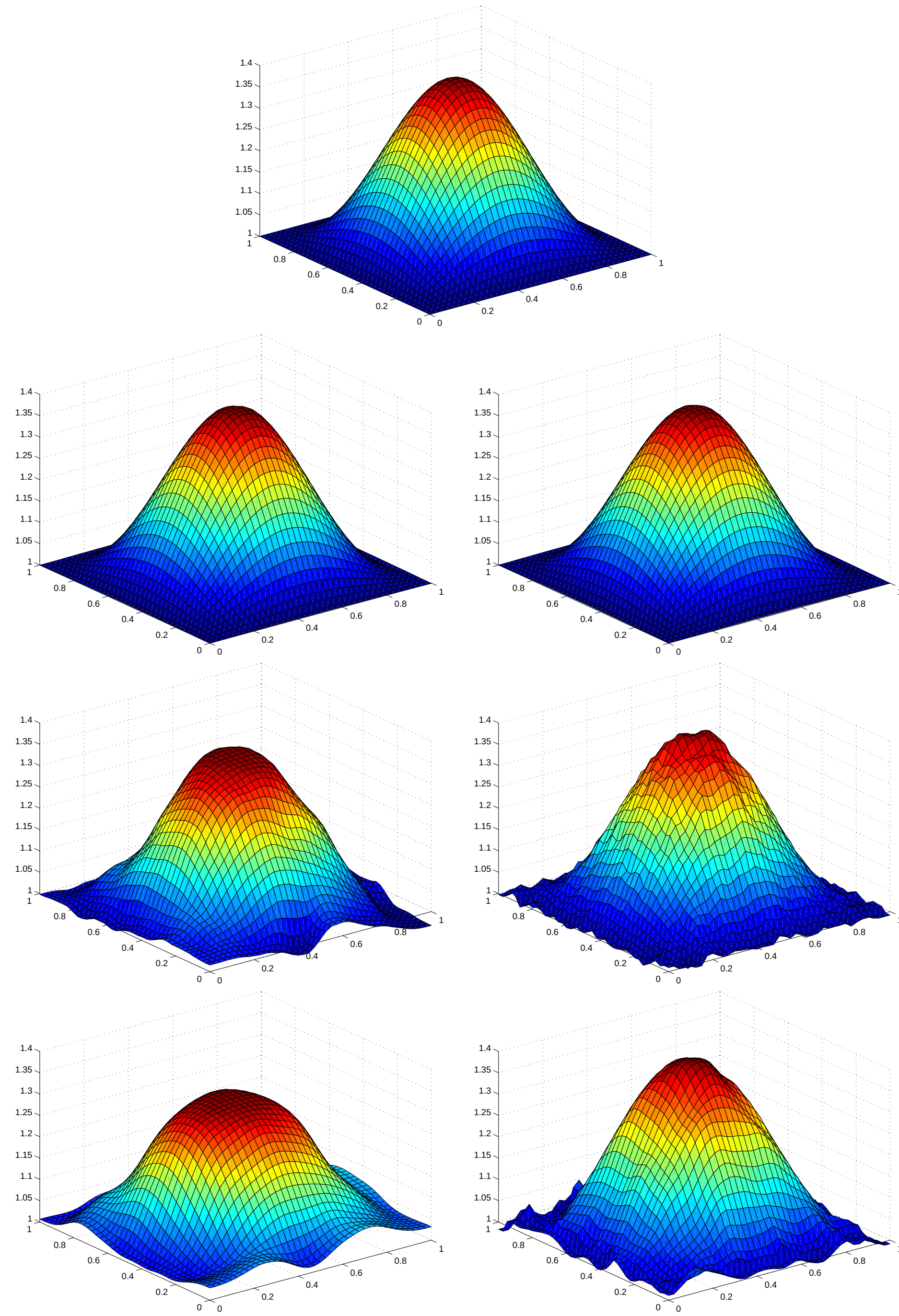

Fig. 1. The top picture is the exact solution ρ. The following six pictures are the numerical solution with $\delta = 0\%, 1\%, 10\%$ on a 40×40 mesh. The left three are for the L^2 formulation and the right three are for the H^1 formulation.

Table 2. The total number of Newton and the average number of GMRES iterations are shown below for a 320 × 320 mesh. The total compute time in seconds is in (·).

	np	$Newton$	$ovlp = 1$	$ovlp = 2$	$ovlp = 4$	$ovlp = 8$	$ovlp = 16$
L^2 formulation	1	3	1(374.33)	1(373.37)	1(375.98)	1(375.57)	1(374.62)
$\beta = 10^{-6}$	4	3	46(108.93)	33(97.62)	18(80.87)	13(79.21)	8(80.46)
$\delta = 0\%$	16	3	66(32.43)	46(26.39)	34(23.92)	22(22.66)	14(26.75)
	64	3	127(23.08)	92(19.22)	63(15.49)	42(14.83)	25(16.35)
L^2 formulation	1	3	1(374.98)	1(374.23)	1(372.92)	1(372.35)	1(374.21)
$\beta = 10^{-5}$	4	3	43(105.49)	26(86.60)	19(80.11)	14(79.02)	9(81.57)
$\delta = 1\%$	16	3	57(30.02)	45(25.89)	31(22.55)	22(23.44)	15(30.14)
	64	3	134(24.71)	94(19.50)	62(15.28)	45(15.09)	25(15.79)
L^2 formulation	1	5	1(623.39)	1(621.60)	1(627.58)	1(622.50)	1(629.40)
$\beta = 10^{-5}$	4	6	61(260.45)	47(225.89)	27(182.45)	18(168.59)	12(172.45)
$\delta = 10\%$	16	6	110(97.01)	81(77.46)	59(67.06)	39(59.56)	24(70.57)
	64	6	234(83.13)	162(62.44)	122(53.56)	78(45.28)	43(50.87)
H^1 formulation	1	3	1(382.09)	1(381.11)	1(384.03)	1(382.27)	1(380.59)
$\beta = 10^{-5}$	4	3	66(136.58)	41(106.42)	24(87.81)	17(84.60)	12(88.99)
$\delta = 0\%$	16	3	148(60.33)	96(43.64)	60(33.56)	37(30.11)	23(34.60)
	64	3	290(47.59)	212(38.34)	121(27.61)	92(25.11)	55(27.08)
H^1 formulation	1	4	1(505.06)	1(503.49)	1(501.99)	1(502.54)	1(504.08)
$\beta = 10^{-5}$	4	4	53(158.88)	34(129.94)	20(110.25)	15(107.46)	10(111.08)
$\delta = 1\%$	16	4	110(63.29)	72(47.44)	47(38.10)	29(34.19)	20(40.42)
	64	4	219(48.50)	142(35.01)	100(28.07)	58(22.82)	44(28.61)
H^1 formulation	1	5	1(624.17)	1(629.97)	1(627.58)	1(629.90)	1(628.54)
$\beta = 10^{-4}$	4	5	62(212.91)	47(178.81)	27(151.06)	18(139.06)	12(143.07)
$\delta = 10\%$	16	5	104(75.61)	82(65.45)	56(53.17)	36(47.70)	22(52.91)
	64	5	221(60.96)	161(49.38)	122(41.46)	71(33.36)	52(38.88)

Table 3. Newton and GMRES iteration numbers are shown below for three different meshes. The compute time in seconds is in (·). *ovlp* is 1/5 of the diameter of the subdomain.

	np	$Newton$	$GMRES$	$Newton$	$GMRES$	$Newton$	$GMRES$
		80 × 80 mesh		160 × 160 mesh		320 × 320 mesh	
L^2 formulation	4	3	6(2.62)	3	6(14.72)	3	6(100.44)
$\beta = 10^{-6}$	16	3	14(2.48)	3	14(6.33)	3	14(26.75)
$\delta = 0\%$	64	3	38(5.73)	3	40(7.28)	3	42(14.83)
L^2 formulation	4	3	7(2.41)	3	7(14.22)	3	6(100.23)
$\beta = 10^{-5}$	16	3	17(2.82)	3	16(6.60)	3	15(30.14)
$\delta = 1\%$	64	3	47(6.74)	3	45(7.68)	3	45(15.09)
L^2 formulation	4	3	9(3.03)	3	8(15.79)	3	8(100.47)
$\beta = 10^{-4}$	16	3	24(3.65)	3	23(8.02)	3	22(34.35)
$\delta = 10\%$	64	3	75(10.41)	3	72(11.47)	3	66(20.66)
H^1 formulation	4	4	8(3.43)	3	8(1.54)	3	8(106.40)
$\beta = 10^{-5}$	16	4	22(4.04)	3	24(7.47)	3	23(34.60)
$\delta = 0\%$	64	4	77(12.68)	3	81(12.14)	3	92(25.11)
H^1 formulation	4	4	8(3.43)	4	8(20.69)	4	7(131.44)
$\beta = 10^{-5}$	16	4	22(4.17)	4	19(9.25)	4	20(40.42)
$\delta = 1\%$	64	4	73(11.90)	4	75(11.89)	4	58(22.82)
H^1 formulation	4	4	8(3.85)	5	8(26.33)	5	8(163.20)
$\beta = 10^{-4}$	16	4	22(4.23)	5	21(12.14)	5	22(52.91)
$\delta = 10\%$	64	4	71(11.71)	5	69(16.88)	5	71(33.36)