Contemporary Mathematics
Volume 180, 1994

Domain Decomposed Preconditioners with Krylov Subspace Methods as Subdomain Solvers

MICHAEL PERNICE

ABSTRACT. Domain decomposed preconditioners for nonsymmetric partial differential equations typically employ exact subdomain solves. Iterative methods for subdomain problems require less storage and allow flexibility in specifying accuracy on the subdomains. Savings in solution time is possible if the effectiveness of the domain decomposed preconditioner is not reduced by lower accuracy subdomain solutions. Numerical experiments compare the overall iteration count as the accuracy of the subdomain solutions is varied. The results demonstrate that the strategy is effective even for low accuracy subdomain solutions.

1. Introduction

Domain decomposition is a technique for constructing parallel algorithms for the solution of partial differential equations. Optimal and near-optimal methods have been developed, but many practical issues must be considered to obtain efficient, scalable implementations. This paper explores how the accuracy of subdomain solutions affects the performance of the overall method for nonsymmetric problems.

The solution of subdomain problems constitutes the largest single computational cost of a domain decomposition method and strongly influences the overall performance and cost of an implementation. Direct subdomain solves require storage for a banded matrix that represents the discrete subdomain operator. The small local memory of many multicomputers limits the size of problems that may be attempted. Also, the subdomain operator may not be explicitly available and may be too expensive to calculate. An obvious alternative is to solve the subdomain problems with an iterative method. This option leads to

1991 *Mathematics Subject Classification.* 65Y05, 65N55, 65F10.
This work was supported in part by a grant from the IBM Corporation.
This paper is in final form, and no version of it will be submitted for publication elsewhere.

other practical considerations, such as the accuracy of the subdomain solutions, possible degradation of the convergence rate of the overall procedure, and the choice of iterative method.

Some of these issues have been explored for symmetric problems [**1, 6, 9**]. Similar studies for nonsymmetric problems have not been pursued. This paper considers several transpose-free Krylov subspace methods to solve the subdomain problems: GMRES [**8**], CGS [**10**], TFQMR [**4**], and CGS with minimum residual smoothing (referred to as SCGS) [**11**].

2. The Domain Decomposition Framework

A nonoverlapping $p \times q$ partition for a total of $N_P = pq$ subdomains and a simplified version of the methods in [**3, 2**] are used. The discrete system

$$Ax = b \tag{1}$$

is solved using preconditioned GMRES [**8**]. The preconditioner has the structure

$$M = \begin{pmatrix} \tilde{A}_\Omega & A_{\Omega\Gamma} & \\ & \tilde{A}_\Gamma & A_{\Gamma\chi} \\ & & \tilde{A}_\chi \end{pmatrix}$$

where $\tilde{A}_\Omega$ is a preconditioner for the subdomains, $\tilde{A}_\Gamma$ is a preconditioner for the interfaces, and $\tilde{A}_\chi$ is a preconditioner for the crosspoint problem [**2**]. $A_{\Omega\Gamma}$ and $A_{\Gamma\chi}$ couple subdomains, interfaces and crosspoints. The preconditioning step requires solution of a linear system involving M and is implemented as a block backsolve. A crosspoint system

$$\tilde{A}_\chi u_\chi = v_\chi$$

is solved first. The crosspoint system is duplicated on every processor using all-to-all communication and is solved using a banded factorization procedure. The operator $\tilde{A}_\chi$ is a coarse-grid analog of the discrete operator A.

The solution of the crosspoint system is used in problems on the interfaces

$$\tilde{A}_\Gamma u_\Gamma = v_\Gamma - A_{\Gamma\chi} u_\chi,$$

which is done in parallel. $\tilde{A}_\Gamma$ is constructed using IP(0) interface probing [**3**].

Finally the subdomain problems

$$\tilde{A}_\Omega u_\Omega = v_\Omega - A_{\Omega\Gamma} u_\Gamma \tag{2}$$

are solved in parallel. Exact subdomain solves use $\tilde{A}_\Omega = A_\Omega$. Using an iterative method to solve (2) produces a variable preconditioner that cannot be accommodated by GMRES. A flexible variant [**7**] is used that allows variable preconditioning but doubles the required storage. Despite this a net savings in storage can be realized. Iterative methods on the subdomains are preconditioned by an MILU factorization [**5**] of the local discrete operator.

3. Numerical Results

Sharp estimates of convergence rates for the methods on the subdomains are not available, making it difficult to predict the amount of work needed to solve (2) and the resulting impact on the solution of (1). Consequently the advantages of using these methods are illustrated with numerical experiments.

3.1. Model Problem. The model problem is the convection-diffusion equation

$$c_1 u_x + c_2 u_y - \epsilon \Delta u = f \quad (x, y) \in \Omega. \tag{3}$$

The equation is discretized using second-order centered differences for the diffusion term and first-order upwind differences for the convection terms.

Problem 1 solves (3) on $\Omega = [0,1] \times [0,1]$ with constant coefficients. Dirichlet boundary conditions and source term f are specified so that the exact solution is

$$u(x,y) = \frac{e^{c_1 x/\epsilon} - 1}{e^{c_1/\epsilon} - 1} + \frac{e^{c_2 y/\epsilon} - 1}{e^{c_2/\epsilon} - 1}.$$

All tests run for this problem were parameterized to produce a fixed $Re_c = 2$.

Problem 2 solves (3) on $\Omega = [-1,1] \times [0,1]$ with variable coefficients

$$c_1(x,y) = 2y(1-x^2) \qquad c_2(x,y) = -2x(1-y^2)$$

and mixed Dirichlet-Neumann boundary conditions:

$$u(x,y) = \begin{cases} 0 & \text{if } x = -1, \text{ or } y = 1 \\ 0 & \text{if } y = 0 \text{ and } -1 \le x < 0 \\ 100 & \text{if } x = 1 \end{cases},$$

$$u_n = 0 \text{ if } y = 0 \text{ and } 0 \le x \le 1$$

and source term $f = 0$. All tests that were run for this problem were parameterized to produce $Re_c \le 4$ throughout the domain.

3.2. Convergence behavior. Sample convergence histories of the tested methods are provided for reference. Results for problem 1 with a uniform mesh size of $h = 1/256$ and horizontal convection appear in Fig. 1. An initial approximation of $x_i = 1$ is used, and the iterations are halted when $\|r_n\| < 10^{-6}\|r_0\|$.

These histories indicate that relaxing the accuracy of the subdomain solutions will benefit GMRES most. They also indicate that CGS is likely to be the most economical method when accurate subdomain solutions are sought and that a fixed number of iterations may not be advisable for CGS and its smoothed variants. Convergence histories for other directions of convection and for problem 2 are similar.

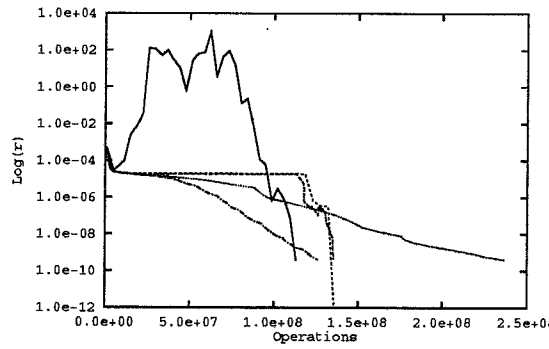

FIGURE 1. Convergence histories for problem 1. CGS: solid line; TFQMR: long dashed line; SCGS: short dashed line; GMRES(20): dotted line; GMRES(5): dashed-dotted line.

3.3. Performance. This section compares the effectiveness of the various domain decomposed preconditioners as the accuracy of the subdomain solves is relaxed. In all cases $p \geq q$ was chosen to favor the computational and storage requirements of the direct methods on the subdomains with a natural ordering, irrespective of other considerations. The measurements were obtained on the nCUBE/2 at the San Diego Supercomputer Center. The time for constructing the preconditioner is included, since this is the dominant cost of using exact solves on the subdomains.

An initial approximation of $x_i = 0$ is used for the subdomain problems and the iterations on the subdomains are halted when $\|r_n\| < \eta\|r_0\|$ for various values of η. In the case of iterative solvers on the subdomains, the subdomain problems needed for interface probing were solved with the iterative method and a fixed tolerance of 10^{-6}, making $\tilde{A}_\Gamma$ independent of η. The overall FGMRES method uses an initial approximation of $x_i = 1$ and the iterations are halted when $\|r_n\| < 10^{-6}$.

Table 1 shows the results for problem 1. In most cases, decreasing η to 10^{-2} does not substantially increase the iteration counts. When it does, the work saved on the subdomains more than compensates for this. For $h = 1/256$ the memory per node was inadequate for a direct method on the subdomains when $N_P = 4$ or 8. For both problem sizes, reducing η to 10^{-1} greatly increases the overall iteration count but surprisingly is still beneficial for GMRES. Similar results were obtained with directions of convection.

Table 2 shows the results for problem 2. Smaller gridsizes were used for this problem because of memory constraints and a uniform meshsize. These results were quite similar to those for problem 1, except to note that for the small problem, the subdomain problems were more difficult for the iterative methods,

TABLE 1. Results for problem 1 with horizontal convection. Execution times are in seconds and iteration counts are in parentheses.

DIRECT METHOD ON SUBDOMAINS

$h = 1/128$			$h = 1/256$			
$p = 2$ $q = 2$	$p = 4$ $q = 2$	$p = 4$ $q = 4$	$p = 2$ $q = 2$	$p = 4$ $q = 2$	$p = 4$ $q = 4$	$p = 8$ $q = 4$
59.9 (8)	14.4 (12)	9.22 (18)	n/a	n/a	86.3 (19)	44.9 (59)

ITERATIVE METHODS ON SUBDOMAINS

$h = 1/128$	η	GMR(20)	GMR(5)	TFQMR	SCGS	CGS
$p = 2$	10^{-6}	53.1 (8)	40.4 (8)	29.9 (8)	30.8 (8)	26.2 (8)
$q = 2$	10^{-4}	39.7 (8)	31.2 (8)	26.0 (8)	26.7 (8)	23.0 (8)
	10^{-2}	23.6 (8)	20.5 (8)	21.2 (8)	22.3 (8)	19.2 (8)
	10^{-1}	18.3 (11)	16.0 (11)	17.2 (11)	18.2 (11)	19.1 (11)
$p = 4$	10^{-6}	22.9 (12)	21.6 (12)	16.5 (12)	17.0 (12)	14.9 (12)
$q = 2$	10^{-4}	17.6 (12)	16.3 (12)	14.1 (12)	14.3 (12)	12.6 (12)
	10^{-2}	10.8 (12)	10.8 (12)	12.5 (13)	12.7 (13)	10.4 (12)
	10^{-1}	8.55 (15)	8.52 (15)	11.1 (16)	11.4 (17)	10.7 (16)
$p = 4$	10^{-6}	17.2 (18)	16.2 (18)	12.8 (18)	13.0 (18)	11.4 (18)
$q = 4$	10^{-4}	13.3 (18)	12.2 (18)	10.7 (18)	10.9 (18)	9.50 (18)
	10^{-2}	8.87 (20)	8.79 (20)	9.97 (20)	9.52 (19)	8.44 (20)
	10^{-1}	8.18 (35)	8.06 (35)	14.5 (46)	20.1 (64)	13.2 (42)

$h = 1/256$	η	GMR(20)	GMR(5)	TFQMR	SCGS	CGS
$p = 2$	10^{-6}	308 (7)	221 (7)	174 (7)	180 (7)	153 (7)
$q = 2$	10^{-4}	239 (7)	170 (7)	154 (7)	161 (7)	132 (7)
	10^{-2}	149 (8)	114 (8)	136 (8)	137 (8)	110 (7)
	10^{-1}	121 (14)	99.4 (14)	127 (13)	132 (13)	115 (13)
$p = 4$	10^{-6}	156 (12)	120 (12)	92.8 (12)	95.1 (12)	81.8 (12)
$q = 2$	10^{-4}	118 (12)	91.5 (12)	80.9 (12)	82.8 (12)	72.0 (12)
	10^{-2}	66.7 (13)	59.7 (13)	69.5 (13)	74.9 (14)	65.3 (14)
	10^{-1}	51.4 (18)	47.5 (18)	65.3 (18)	71.0 (19)	69.9 (19)
$p = 4$	10^{-6}	123 (19)	99.9 (19)	75.1 (19)	76.7 (19)	65.5 (19)
$q = 4$	10^{-4}	92.7 (19)	74.0 (19)	63.4 (19)	65.1 (19)	55.8 (19)
	10^{-2}	50.3 (20)	45.6 (20)	59.0 (22)	56.1 (20)	48.8 (20)
	10^{-1}	43.3 (39)	41.2 (39)	259 (163)	420 (263)	347 (224)
$p = 8$	10^{-6}	106 (59)	99.8 (59)	77.8 (59)	79.5 (59)	68.8 (59)
$q = 4$	10^{-4}	78.9 (59)	71.6 (59)	64.5 (59)	65.7 (59)	57.7 (59)
	10^{-2}	44.2 (60)	44.6 (60)	53.1 (59)	53.7 (60)	47.6 (59)
	10^{-1}	46.3 (120)	46.2 (120)	95.8 (157)	260 (423)	203 (345)

TABLE 2. Results for problem 2. Execution times are in seconds and iteration counts are in parentheses.

DIRECT METHOD ON SUBDOMAINS

$h = 1/64$			$h = 1/128$			
$p = 2$ $q = 2$	$p = 4$ $q = 2$	$p = 4$ $q = 4$	$p = 2$ $q = 2$	$p = 4$ $q = 2$	$p = 4$ $q = 4$	$p = 8$ $q = 4$
26.3 (5)	6.15 (9)	4.64 (19)	n/a	64.4 (9)	42.3 (19)	19.5 (50)

ITERATIVE METHODS ON SUBDOMAINS

$h = 1/64$	η	GMR(20)	GMR(5)	TFQMR	SCGS	CGS
$p = 2$	10^{-6}	16.8 (5)	12.6 (5)	11.4 (5)	11.3 (5)	10.1 (5)
$q = 2$	10^{-4}	12.4 (5)	9.92 (5)	9.47 (5)	9.90 (5)	8.44 (5)
	10^{-2}	8.52 (5)	7.12 (5)	8.09 (5)	8.17 (5)	7.01 (5)
	10^{-1}	8.61 (9)	7.22 (9)	7.89 (7)	8.01 (7)	6.63 (6)
$p = 4$	10^{-6}	12.6 (9)	10.1 (9)	8.61 (9)	8.74 (9)	7.55 (9)
$q = 2$	10^{-4}	9.42 (9)	7.89 (9)	7.15 (9)	7.21 (9)	6.24 (9)
	10^{-2}	6.67 (10)	6.06 (10)	5.47 (10)	5.68 (9)	5.02 (9)
	10^{-1}	5.89 (14)	5.26 (14)	5.28 (11)	5.29 (11)	4.75 (10)
$p = 4$	10^{-6}	11.1 (19)	9.75 (19)	7.54 (19)	7.79 (19)	6.66 (19)
$q = 4$	10^{-4}	8.01 (19)	7.45 (19)	6.38 (19)	6.50 (19)	5.65 (19)
	10^{-2}	5.29 (19)	4.88 (19)	5.08 (19)	5.10 (19)	4.49 (19)
	10^{-1}	5.57 (35)	5.26 (35)	5.27 (24)	5.89 (28)	4.79 (24)

$h = 1/128$	η	GMR(20)	GMR(5)	TFQMR	SCGS	CGS
$p = 2$	10^{-6}	106 (5)	83.4 (5)	78.0 (5)	82.0 (5)	68.4 (5)
$q = 2$	10^{-4}	84.3 (5)	66.3 (5)	67.7 (5)	72.0 (5)	60.5 (5)
	10^{-2}	53.2 (5)	45.4 (5)	56.7 (5)	57.7 (5)	49.6 (5)
	10^{-1}	53.5 (9)	43.8 (9)	53.9 (7)	51.5 (6)	50.2 (7)
$p = 4$	10^{-6}	79.0 (9)	66.4 (9)	52.0 (9)	53.8 (9)	45.5 (9)
$q = 2$	10^{-4}	61.7 (9)	53.2 (9)	44.6 (9)	46.0 (9)	40.1 (9)
	10^{-2}	36.7 (9)	35.7 (9)	36.4 (9)	37.2 (9)	31.8 (9)
	10^{-1}	27.5 (13)	24.9 (13)	31.7 (10)	32.7 (10)	29.8 (10)
$p = 4$	10^{-6}	68.9 (19)	63.2 (19)	45.5 (19)	46.8 (19)	40.0 (19)
$q = 4$	10^{-4}	52.6 (19)	49.3 (19)	39.1 (19)	40.4 (19)	34.5 (19)
	10^{-2}	30.4 (19)	29.1 (19)	30.4 (19)	31.5 (19)	27.7 (19)
	10^{-1}	23.3 (35)	23.2 (36)	28.0 (23)	25.5 (20)	22.9 (20)
$p = 8$	10^{-6}	75.7 (53)	57.9 (53)	45.2 (50)	46.7 (50)	40.0 (50)
$q = 4$	10^{-4}	49.1 (53)	43.4 (53)	37.4 (50)	38.4 (50)	33.4 (50)
	10^{-2}	29.1 (55)	27.7 (55)	28.8 (51)	29.1 (51)	25.5 (50)
	10^{-1}	21.6 (80)	21.1 (80)	43.3 (96)	45.3 (102)	30.5 (74)

resulting in faster performance for the direct method on the subdomains with $N_P = 16$ and 32.

In general, using an iterative method to solve (2) is more effective for small values of N_P and all values of η. For larger values of N_P the strategy is not competetive unless large values of η are used.

4. Conclusions

The domain decomposed preconditioners remain effective until η is decreased to 10^{-1}. Performance improvement can be achieved by reducing the accuracy of the subdomain solutions. The lower memory requirements of the iterative methods allow larger problems to be attempted, which can be a critical factor when incorporating these techniques into actual applications.

Direct methods are preferred when the subdomain problems are small. For larger subdomain problems CGS and GMRES with a small restart parameter were the most effective of the iterative methods. The results for $\eta = 10^{-1}$ suggest that a fixed number of iterations should not be used with CGS and related methods. For the model problems that were evaluated, the additional cost of smoothing the CGS residuals does not appear to be justified. Future work will extend these ideas to overlapping domain partitions.

References

1. C. Börgers, *The Neumann-Dirichlet domain decomposition method with inexact solvers on the subdomains*, Numer. Math. **55** (1989), 123–136.
2. X.-C. Cai, W. D. Gropp, and D. E. Keyes, *A comparison of some domain decomposition algorithms for nonsymmetric elliptic problems*, Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations (David E. Keyes, Tony F. Chan, Gérard Meurant, Jeffrey S. Scroggs, and Robert G. Voigt, eds.), 1992, pp. 224–235.
3. T. F. Chan and D. E. Keyes, *Interface preconditionings for domain-decomposed convection-diffusion operators*, Third International Symposium on Domain Decomposition Methods for Partial Differential Equations (T. F. Chan, R. Glowinski, J. Périaux, and O. B. Widlund, eds.), 1989, pp. 245–262.
4. R. W. Freund, *A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems*, SIAM J. Sci. Comput. (1993), 470–482.
5. I. Gustafsson, *A class of first order factorization methods*, BIT (1978), 142–156.
6. A. Meyer, *A parallel preconditioned conjugate gradient method using domain decomposition and inexact solvers on each domain*, Computing **45** (1990), 217–234.
7. Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput. **14** (1993), 461–469.
8. Y. Saad and M. H. Schultz, *GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput. **7** (1986), 856–869.
9. B. F. Smith, *A parallel implementation of an iterative substructuring algorithm for problems in three dimensions*, SIAM J. Sci. Comput. **14** (1993), 406–423.
10. P. Sonneveld, *CGS, a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput. **10** (1989), 36–52.
11. L. Zhou and H. F. Walker, *Residual smoothing techniques for iterative methods*, SIAM J. Sci. Comput. **15** (1994), 297–312.

Utah Supercomputing Institute. University of Utah. Salt Lake City. Utah 84112
E-mail address: usimap@sneffels.usi.utah.edu