SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 905–925, July 1992

013

PRECONDITIONING FOR BOUNDARY INTEGRAL EQUATIONS*

STEPHEN A. VAVASIS†

Abstract. New classes of preconditioners are proposed for the linear systems arising from a boundary integral equation method. The problem under consideration is Laplace's equation in three dimensions. The system arising in this context is dense and unsymmetric. These preconditioners, which are based on solving small linear systems at each node, reduce the number of iterations in some cases by a factor of 8. Three iterative methods are considered: conjugate gradient on the normal equations, CGS of Sonneveld, and GMRES of Saad and Schultz. For a simple model problem, the exact relationship between the preconditioners and the resulting condition number of the system is investigated. This analysis proves that the condition number of the preconditioned system is decreased by a factor asymptotically greater than any constant.

Key words. integral equations, boundary element methods, preconditioners, iterative methods

AMS(MOS) subject classifications. 65F10, 65R20, 65N38, 65F35

1. Boundary integral equations.

The *boundary integral equation method* (BIEM) is a technique for solving certain kinds of boundary value problems including Laplace's equation. The specific problem under consideration is: Given a compact region $\Omega \in \mathbb{R}^3$ whose boundary is homeomorphic to a sphere, find a solution to $\nabla^2\phi = 0$ given Neumann or Dirichlet boundary data. The Neumann and Dirichlet boundary data can be expressed in terms of one another via integral equations. By discretizing these integral equations, we arrive at a dense linear system relating the Neumann and Dirichlet data on the boundary. If Dirichlet data is given, it can be solved for the Neumann data or vice versa. Once both sets of boundary data are known, further integrations yield interior values of ϕ.

Our application for Laplace's equation is a free-surface irrotational incompressible fluid flow in three dimensions (see Liu, Hsu, Lean, and Vavasis [16]). The data determined by BIEM in this case is the normal derivative of the velocity potential on the free surface.

Accordingly, each timestep of the fluid calculation requires a new solution of Laplace's equation, which in turn requires the solution of the dense unsymmetric system of linear equations arising from BIEM.

For the remainder of this section we explain the details of the integral equations and linear systems. The boundary integral equation method is credited to Hess and Smith [12], Jaswon [14], and Symm [24]. For more information about the method, we refer the reader to the series edited by Brebbia [5].

Suppose ϕ satisfies Laplace's equation. Let z be a point on $\partial\Omega$. Let $g_z(x)$ be the function

$$g_z(x) = \frac{1}{2\pi\|x - z\|}.$$

* Received by the editors April 5, 1990; accepted for publication (in revised form) August 26, 1991. This work was supported in part by a gift from Xerox Webster Research Center and by the U.S. Army Research Office through the Mathematical Sciences Institute of Cornell University. Revision work was supported by National Science Foundation Presidential Young Investigator award and by the Applied Mathematical Sciences program, U.S. Department of Energy, under contract DE-AC04-76DP00789, while the author was visiting Sandia National Laboratories.

† Department of Computer Science, Cornell University, Ithaca, New York 14853 (vavasis@cs.cornell.edu).

For now and for the rest of the paper, $\|\cdot\|$ will denote the 2-norm. Lower case letters are used for scalars and for two- or three-dimensional vectors. Boldface is used for vectors of higher dimension. Capital letters are used for matrices. With this definition of g, the following identity always holds:

$$c\phi(z) = \int_{\partial\Omega} \frac{\partial\phi}{\partial n}(x) g_z(x)\, ds - \int_{\partial\Omega} \frac{\partial g_z}{\partial n}(x)\phi(x)\, ds. \tag{1}$$

Here c measures the angle at the point z and would be 1 if the boundary is smooth. There is an analogous equation for two-dimensional Laplace problems with a different choice for the function g (called the *Green's function*). See §5. Note that the integrals in this equation are improper because the function g_z and its derivative are infinite at $x = z$. The integrals still have a well-defined value, but these singularities make numerical methods more complicated.

Suppose we divide the surface of Ω into elements joined by nodes. Let the nodes be $\{z_1, \cdots, z_n\}$. Then an equation like (1) holds for each z_i. Writing down equations for each z_i in this fashion is generally known as *collocation*. We can approximate the solution to the Laplace equation by assuming that the ϕ and $\partial\phi/\partial n$ are piecewise polynomial interior to each element and are therefore determined by their values at nodal points. This means that there are $2n$ variables $\phi(z_1), \cdots, \phi(z_n)$ and $\partial\phi/\partial n(z_1), \cdots, \partial\phi/\partial n(z_n)$. The integrations in (1) are approximated with numerical quadrature; the integral on each element can be expressed as a linear combination of the variables at its vertices. This gives a dense linear system of n equations in the $2n$ variables; the system looks like this:

$$\begin{aligned} c_1\phi(z_1) &= \sum_{j=1}^{n} d_{1j}\partial\phi/\partial n(z_j) - \sum_{j=1}^{n} e_{1j}\phi(z_j), \\ &\vdots \\ c_n\phi(z_n) &= \sum_{j=1}^{n} d_{nj}\partial\phi/\partial n(z_j) - \sum_{j=1}^{n} e_{nj}\phi(z_j). \end{aligned} \tag{2}$$

In this system of equations the coefficients d_{ij} and e_{ij} arise from evaluating the Green's function. Accordingly, all of the coefficients (c's, d's, and e's) are determined entirely by the geometry of $\partial\Omega$. See Atkinson [1] for a description of specific methods for collocation.

In a well-posed Laplace problem, for each point on the boundary either $\partial\phi/\partial n$ or ϕ is specified at the point. Therefore, in the above system of equations, n out of the $2n$ variables will be specified and n will be unknown. Since (2) has n equations, we use these equations to solve for the remaining unknowns. This system will generally be dense and unsymmetric.

It follows that the coefficient matrix of the system to be solved is determined entirely by $\partial\Omega$ and by the specification of which kind of boundary data is provided for each node.

Once the values of ϕ and $\partial\phi/\partial n$ are known, there is an equation similar to (1) to compute any interior value of ϕ.

In some applications of BIEM, interior values are not needed. In our fluid application it is enough to know $\partial\phi/\partial n$ everywhere on the boundary.

The remainder of the paper is organized as follows. In §2 we discuss iterative methods for solving (2). In §3 we introduce our classes of preconditioners for this

problem. In §4 we report on our computational experience with the algorithm. In §5 we give an analysis of one of the preconditioners for a simple model problem. We show that the condition number is reduced by a factor of $\sqrt{\ln n}$ for the model. Finally, in §6 we discuss computational complexity issues connected with the BIEM solution.

The emphasis of this paper is on the use of the preconditioners for Laplace problems in three dimensions with collocation and mixed boundary conditions. There appears to be very little known about such matrices; they lack many of the properties (such as symmetry and positive definiteness) that are instrumental for analysis of finite-element methods.

Accordingly, we do not attempt an analysis of the preconditioners for the problems actually of interest to us. The analysis given in §5 is for a simple two-dimensional model problem and appears to be pessimistic compared to our computational experiments.

2. Iterative methods for linear systems. An *iterative* method for solving a system of linear equations $A\mathbf{x} = \mathbf{b}$ is a method that generates a sequence of vectors converging to the true solution. Applied to a dense $n \times n$ system, one iteration of an iterative method ordinarily requires $O(n^2)$ steps. Solving the system with elimination requires $O(n^3)$ steps. Accordingly, the hope is that a good iterative method will require only a few iterations and be faster than elimination.

Canning [6] has developed a technique to reduce the number of nonzero elements in a BIEM matrix from n^2 to $O(n)$ in many cases. Iterative methods would become even more attractive, because the time per iteration is $O(n)$. We have not tried Canning's method for our problem.

The three iterative methods under consideration in this paper are GMRES, CGS, and conjugate gradient applied to the normal equations. The GMRES algorithm is due to Saad and Schultz [21] and operates directly on the original system. CGS, due to Sonneveld [23], also operates on the original system. Conjugate gradient (CG), due to Hestenes and Stiefel [13], only works for symmetric positive definite systems. Accordingly, CG is applied to $A^TA\mathbf{x} = A^T\mathbf{b}$ instead of $A\mathbf{x} = \mathbf{b}$. Note that the product A^TA is never actually formed—this would ruin the bound on the running time. Instead, whenever a product $A^TA\mathbf{y}$ is needed by CG, we carry out the product as two matrix-vector multiplications $A^T(A\mathbf{y})$. This remark about matrix products also applies to the case described below in which preconditioners are inserted into the problem. CG applied to the normal equations is known as CGNR.

It is known (see Golub and Van Loan [10]) that the convergence rate of CGNR can be bounded in terms of the condition number of the matrix of coefficients. The convergence rate for GMRES is determined by the positions of the eigenvalues or pseudoeigenvalues (see Nachtigal, Reddy, and Trefethen [18]). There is no simple formula known for the convergence of CGS.

Suppose that we have a nonsingular matrix P such that PA has a lower condition number than A alone. Then the system of equations $PA\mathbf{x} = P\mathbf{b}$ has the same solution as $A\mathbf{x} = \mathbf{b}$. Moreover, an iterative method applied to $PA\mathbf{x} = P\mathbf{b}$ might converge faster because of the lower condition number. Such a matrix P is called a *left preconditioner*. Ideally, we want to have PA close to the identity matrix.

Each iteration of conjugate gradient requires one matrix-vector multiply operation. Since we are working on the normal equations, we need two matrix-vector multiplications. Finally, if we use preconditioning then the system becomes A^TP^TPA so we need four matrix-vector multiplications.

Each iteration of CGS requires two matrix-vector multiplications. Each iteration

of GMRES requires one matrix-vector multiplication. The time required for the kth iteration of GMRES is proportional to $O(kn + n^2)$, in contrast to CGNR and CGS, which require only $O(n^2)$ steps per iteration.

3. Preconditioners for BIEM. In this section we introduce three classes of preconditioners tailored to the BIEM problem described in the introduction. All three of our preconditioners are based on the same idea, which is as follows. The rows of preconditioner P are generated independently and can be done in parallel. Let the ith column of P^T be denoted as $\mathbf{p}_i$. As mentioned in the last section, we would ideally like to have $A^T\mathbf{p}_i = \mathbf{e}_i$, where $\mathbf{e}_i$ is the ith column of the identity matrix.

For our preconditioners we take the following approach. From the matrix A and other data we determine some small list L of indices drawn from $\{1, \cdots, n\}$ such that the variables and constraints in L have the most impact on variable i. Then we solve the small system of equations $\bar{A}^T\bar{\mathbf{p}}_i = \bar{\mathbf{e}}_i$ where the bars over the variables indicate that we are deleting all the rows and columns except for those in L. Once this solution is known, we expand it back to the entries of P. This is done for all rows of P; the rows can be generated in parallel.

The brief description in the last paragraph completely explains how to compute our classes of preconditioners. All that remains is the explanation of how L is chosen and how to expand the solution of the small system back to the row of P. In all cases the number of flops to compute the preconditioner is bounded by $O(k^3n)$ where k is the maximum size of the small system solved for each node.

Preconditioner MN. Preconditioner MN stands for "mesh neighbor." Recall that each variable of the linear system arising in a BIEM problem corresponds to a node on the surface of the region. We say that two nodes are *neighbors* if they border on a common element. The choice of L for node i is as follows: We let L be i together with all the indices of neighbors of node i. After the small system involving these rows and columns is solved, then we scatter the entries of the solution vector $\bar{\mathbf{p}}_i$ back to their original coordinate positions in $\mathbf{p}_i$, and we fill in the remaining positions of $\mathbf{p}_i$ with zeros.

The motivation for preconditioner MN is as follows. The matrix coefficient relating node i and node j comes up roughly from a formula like $1/\|x_i - x_j\|$. Therefore, the further apart two nodes are, the less impact we would expect a change at one node to have on the other. Since neighboring nodes are the most interrelated, we put them in our preconditioner.

Note that the preconditioner generated in this manner will be sparse, and in particular, its sparsity pattern will mirror the connectivity of the mesh. In general, sparsity is considered a very desirable feature of a preconditioner. See, for example, Manteuffel [17] or Kolotilina and Yeremin [15].

For our application, sparsity is an added benefit but not a major goal. The reason is that A is already dense, so we have to carry out a dense $n \times n$ matrix-vector multiply operation in each iteration. Therefore, the additional multiplication involving P is not too expensive, relatively speaking, even if P is dense.

Preconditioner ME. Preconditioner ME stands for "matrix entries." The way to choose L for a variable i for this preconditioner is as follows. If a_{ij}, a_{ji} satisfy $|a_{ij}a_{ji}| \geq t|a_{ii}a_{jj}|$ then include j in L. Here, t denotes some user-specified tolerance (our choice was $t = 0.1$). Note that this preconditioner does not depend on any BIEM structure and hence could be applied to an arbitrary matrix.

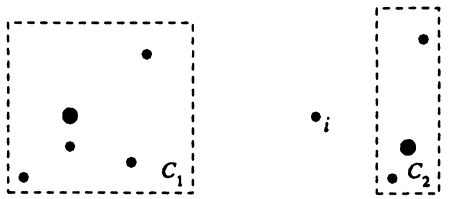

FIG. 1. *C_1 is acceptable; C_2 is not acceptable.*

After the small system is solved, the solution vector is scattered back into P as in preconditioner MN.

This preconditioner is similar to the local matrix idea of Benson and Frederickson [2].

Preconditioner HC. Preconditioner HC stands for "hierarchical clustering." Hierarchical clustering, an idea due to Rokhlin [20], classifies nodes based on how far away they lie from node i. This is the most complicated of the three preconditioners and requires a more extensive explanation.

A *cluster* refers to a set of nodes that are intended to be near one another. The first step in constructing HC is to make a hierarchy of clusters. The top cluster C_0 of the hierarchy contains all the nodes. The next level contains two clusters, each with about half the nodes. These clusters are then recursively subdivided, so that the kth level of the hierarchy consists of 2^k mutually disjoint clusters of about equal size whose union is the entire set of nodes.

There are many algorithms for clustering geometric objects (see, for example, Feder and Greene [9] and Greengard [11]). We have chosen the following simple approach, which seems to work well in practice. The top level cluster C_0 is divided in two by splitting the points according to the median x_1 coordinate. The clusters at the next level are split according to their median x_2 coordinates. The next level is split according to median x_3 coordinates. The further levels cycle through the three coordinates.

Once the cluster hierarchy is built, we next identify a *center* for each cluster. This is the node in the cluster such that the maximum distance to other nodes in the cluster is minimized. In our current implementation we find the exact center of each cluster in the hierarchy; this fairly expensive computation ($O(n^2)$ steps total) could be replaced by a heuristic. The *radius* of a cluster is defined to be the maximum distance from its center to other nodes in the cluster.

Once the clusters, centers, and radii are computed, we next address the actual preconditioner. For each node i, we carry out the following steps to produce the list L of nodes related to i. We say that a cluster C is *acceptable* to node i if the distance from node i to the center of C is at least tr, where r is the radius of C and $t > 1$ is a user-specified parameter. In Fig. 1 we show a sample node i and an acceptable and unacceptable cluster, with $t = 2$. The centers of the two clusters are enlarged.

The algorithm for constructing list L for node i is the following recursive procedure, whose initial arguments are (i, C_0).

function **make-list**(i, C)
    if C is acceptable to i then
        make-list := {center of C}.
    else
        Let C_1, C_2 be the children of C in
          the hierarchy.
        make-list := **make-list**$(i, C_1) \cup$ **make-list**(i, C_2).
    end

Note that this recursive procedure is guaranteed to terminate, since the bottom level of the hierarchy consists of clusters made up of individual nodes. These clusters have radius zero and hence are acceptable to all nodes.

The motivation is as follows. Suppose C is acceptable to i. The coupling in matrix A between the nodes of cluster C and node i will not depend very much on which node of C is selected. Therefore, we could summarize all the nodes of C with a single node, namely, the center. This motivation follows Rokhlin [20] and Greengard [11], although their algorithms for determining the representation of a cluster are considerably more sophisticated since they are interested in getting strong bounds on the difference between the "cluster" approximation and the true effect of all nodes of the cluster.

The list L is generated in this manner, and then the small system for i is solved. In preconditioners MN and ME the elements of the small system solution were scattered throughout $\mathbf{p}_i$ and the remaining elements were filled with zeros. Preconditioner HC will be dense; the entry at position m of $\mathbf{p}_i$ is taken to be equal to the entry in $\bar{\mathbf{p}}_i$ corresponding to the representative of m's cluster, scaled by the cluster size.

For the parameter t in the definition of "acceptable," we used $t = 1.5$ in our experiments. This seemed to give reasonable preconditioning results. It should be noted that larger values of t give more reliable clusters because the "clustering" effect is expected to be more apparent, but they also result in significantly more expensive preconditioners since the lists L become larger. Rokhlin [20] settles on a choice for t based on careful analysis of the clustering effect, which we have not been able to carry out in the setting of preconditioners.

It is also not clear whether it makes sense to force the cluster boundaries to respect nondifferentiable edges and vertices of the region or boundaries between regions with different types of boundary conditions (we did not). The theory for boundary element methods with nonsmooth boundaries and mixed boundary conditions is not understood well enough to make definite statements in this regard.

An important question to ask about our preconditioners is whether they can be stably computed in all cases. The methods break down if one of the small linear systems is singular. We have the following theorem in this regard.

THEOREM 3.1. *If A is either symmetric positive definite or strictly diagonally dominant, then all three preconditioners can be computed stably.*

This theorem is immediately obvious from the fact that the small systems that are solved in all cases are principal submatrices of A. Therefore, these submatrices are symmetric positive definite if A is, and they are strictly diagonally dominant if A is.

Of course, the matrices of interest to us are neither symmetric positive definite nor strictly diagonally dominant. We have not encountered any instabilities in our test runs (such instabilities would most likely manifest themselves as unexpectedly large elements of the preconditioner). There is no theory at present to predict whether this

could happen.

The primary value of the above theorem is to show that our preconditioning ideas may be applicable to a broad variety of dense or even sparse problems. By way of comparison, the well-known incomplete Cholesky preconditioners (see Manteuffel [17] and Elman [8]) can be stably computed for only a subset of symmetric positive definite matrices.

4. Computational experience. We tried all three preconditioners out on three different test problems. For each problem we tried CGS, GMRES, and CGNR. We also noted the condition numbers of the system with and without preconditioners. All experiments were carried out with MATLAB on an Ardent Titan. MATLAB, an interactive language for numerical computation, is a trademark of The Mathworks, Inc. The results are summarized in Tables 1–3. The numbers in the six middle columns are number of iterations and number of millions of floating point operations for the iterative methods as computed by MATLAB. The last column is the 2-condition number. An illustration of the domains of the test problems are in Figs. 2–4.

In all of the domains we used mixed boundary conditions. The surfaces are divided into triangular elements, and the functions are assumed to be piecewise linear on the triangular elements. The collocation points are identical to the nodal points of the discretization.

Dense matrix-vector multiplication was used to compute $PA\mathbf{x}$ in the iterative routines, even when P was sparse (including the case $P = I$). Accordingly, the floating point operation counts are higher in many cases than would be expected from the number of iterations.

The floating point count does not include the time to compute the preconditioner. In the cases of MN and ME, the operations to compute the preconditioner were negligible. In the case of HC, the number of operations to compute the preconditioner was substantial—on the same order as the number of operations of the iterative method. In the domain for Table 1, 0.69 million operations were required to compute HC. The numbers for Tables 2–3 are 2.42 million and 2.20 million, respectively.

Our convergence criterion was as follows. In all three routines we stopped when the residual computed by the routine dropped below $\epsilon|\hat{\mathbf{b}}|$, where ϵ denotes the unit roundoff (around $2.2 \cdot 10^{-16}$ on our computer) and $\hat{\mathbf{b}}$ denotes the right-hand side of the actual system being solved. In GMRES we stopped also when the residual norm did not improve by a factor smaller than 0.9 over two consecutive iterations. This was necessary in GMRES because we noticed its tendency to stop converging when the residual was slightly larger than $\epsilon|\hat{\mathbf{b}}|$. Such a test was not possible in CGS or CGNR because of oscillation in the computed residual.

In most cases the residual $\|\mathbf{b} - A\mathbf{x}\|$ computed a posteriori was on the order of 10^{-14} or 10^{-15}; in a few cases it was as large as 10^{-12}. These anomalous residuals always occurred in trials with CGS, where we discovered a divergence between the a posteriori residual and the computed residual $\|\mathbf{r}\|$ in the algorithm.

In each trial the right-hand side was generated randomly (each component was selected uniformly at random between -0.5 and 0.5). The same right-hand side was used for all trials in a table.

The most striking feature of these tables is the drastic reduction in the number of iterations when comparing preconditioned systems to the case with no preconditioner. In some cases the reduction is over a factor of 8.

The three preconditioners themselves seemed to have similar properties. We notice that preconditioner ME generally did slightly worse than the other two. Precon-

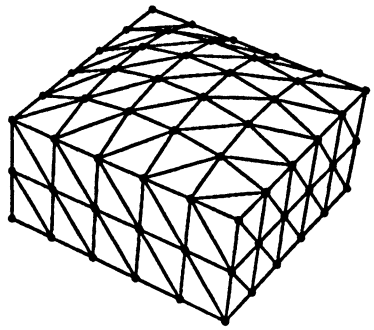

FIG. 2. *Domain D_1 with Dirichlet conditions on the top surface, Neumann conditions on the other surfaces.*

TABLE 1
Domain D_1 computational results ($n = 92$).

Precond.	CGS		GMRES		CGNR		κ_2
	iter.	m.flop.	iter.	m.flop.	iter.	m.flop.	
I	106	8.25	92	4.85	189	13.16	45.7
MN	12	0.88	20	0.82	33	2.28	3.2
HC	13	0.96	22	0.91	41	2.84	4.6
ME	13	0.96	22	0.91	38	2.63	4.3

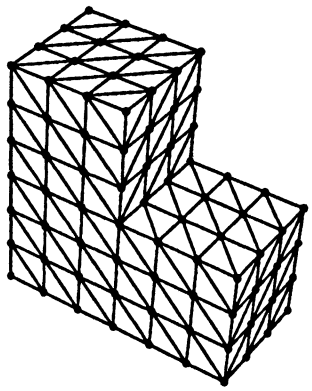

FIG. 3. *Domain D_2 with Dirichlet conditions on the two small end surfaces, Neumann conditions elsewhere.*

TABLE 2
Domain D_2 computational results ($n = 128$).

Precond.	CGS		GMRES		CGNR		κ_2
	iter.	m.flop.	iter.	m.flop.	iter.	m.flop.	
I	98	14.65	96	8.91	243	32.50	70.6
MN	13	1.84	21	1.62	46	6.12	5.3
HC	13	1.84	22	1.69	50	6.66	5.6
ME	14	2.00	23	1.77	55	7.33	6.6

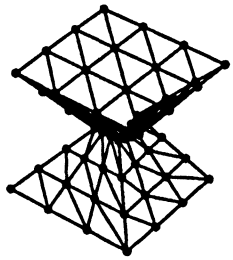

FIG. 4. *Domain D_3 with Dirichlet conditions on all surfaces except the bottom.*

TABLE 3
Domain D_3 computational results ($n = 92$).

Precond.	CGS		GMRES		CGNR		κ_2
	iter.	m.flop.	iter.	m.flop.	iter.	m.flop.	
I	118	9.20	64	3.05	198	13.79	75.1
MN	14	1.04	24	1.00	46	3.19	6.0
HC	14	1.04	24	1.00	38	2.63	3.9
ME	17	1.27	26	1.08	52	3.61	8.4

ditioner MN did slightly better than the other two. Algorithm GMRES marginally outperformed CGS in all trials when comparing the measured number of floating point operations; CGNR was a distant third.

In order to understand in practice how the preconditioners affect the convergence rates, we have plotted a sample of the pseudoeigenvalues of A and PA in Fig. 5, where P denotes preconditioner MN. These tests were done for an L-shaped region similar to domain D_2 but with fewer nodes. A sample of the pseudoeigenvalues of X (where $X = A$ or $X = PA$) was computed by plotting the eigenvalues of 20 complex perturbations of X. The entries of the perturbation were chosen uniformly at random in the disk in $\mathbb{C}$ of radius δ, where δ was taken to be 0.01, 0.005, and 0.0025. A rough rule is that convergence of GMRES is better when the pseudoeigenvalues are clustered away from zero; see Nachtigal, Reddy, and Trefethen [18]. We notice that the clustering seems to be greatly improved in the case of the preconditioned matrix.

To be precise, [18] has the following bound, based on earlier work by Greenbaum, Trefethen [25], and others:

$$\|\mathbf{b} - A\mathbf{x}^{(k)}\| \leq \|\mathbf{b} - A\mathbf{x}^{(0)}\| \cdot \inf\{\sup\{|p(z)| : z \in \Lambda_\epsilon\} : p \in \Pi_k\} \cdot L/(2\pi\epsilon).$$

In this formula, $\mathbf{x}^{(k)}$ denotes the kth iterate of GMRES; Λ_ϵ is the set of ϵ-pseudoeigenvalues of A (a subset of $\mathbb{C}$); L is the length of the boundary of Λ_ϵ; and Π_k is the set of degree-k complex polynomials $p(z)$ such that $p(0) = 1$. Thus, if the pseudospectrum is clustered away from the origin, it is easier to find a low-degree polynomial that nearly vanishes on the pseudospectrum and that is 1 at the origin, indicating that the inf factor in the preceding bound will be small. Conversely, a bad case for GMRES is when the eigenvalues or pseudoeigenvalues are clustered around the origin.

In the plots of Fig. 5, the pseudoeigenvalues of A are shown side by side with the pseudoeigenvalues of PA, for $\delta = 0.01, 0.005, 0.0025, 0$. Note that when $\delta = 0$, we are plotting the exact eigenvalues. Another way to bound the convergence of GMRES is the following formula in terms of exact eigenvalues:

$$\|\mathbf{b} - A\mathbf{x}^{(k)}\| \leq \|\mathbf{b} - A\mathbf{x}^{(0)}\| \cdot \kappa_2(V) \cdot \inf\{\sup\{|p(z)| : z \in \Lambda_0\} : p \in \Pi_k\}.$$

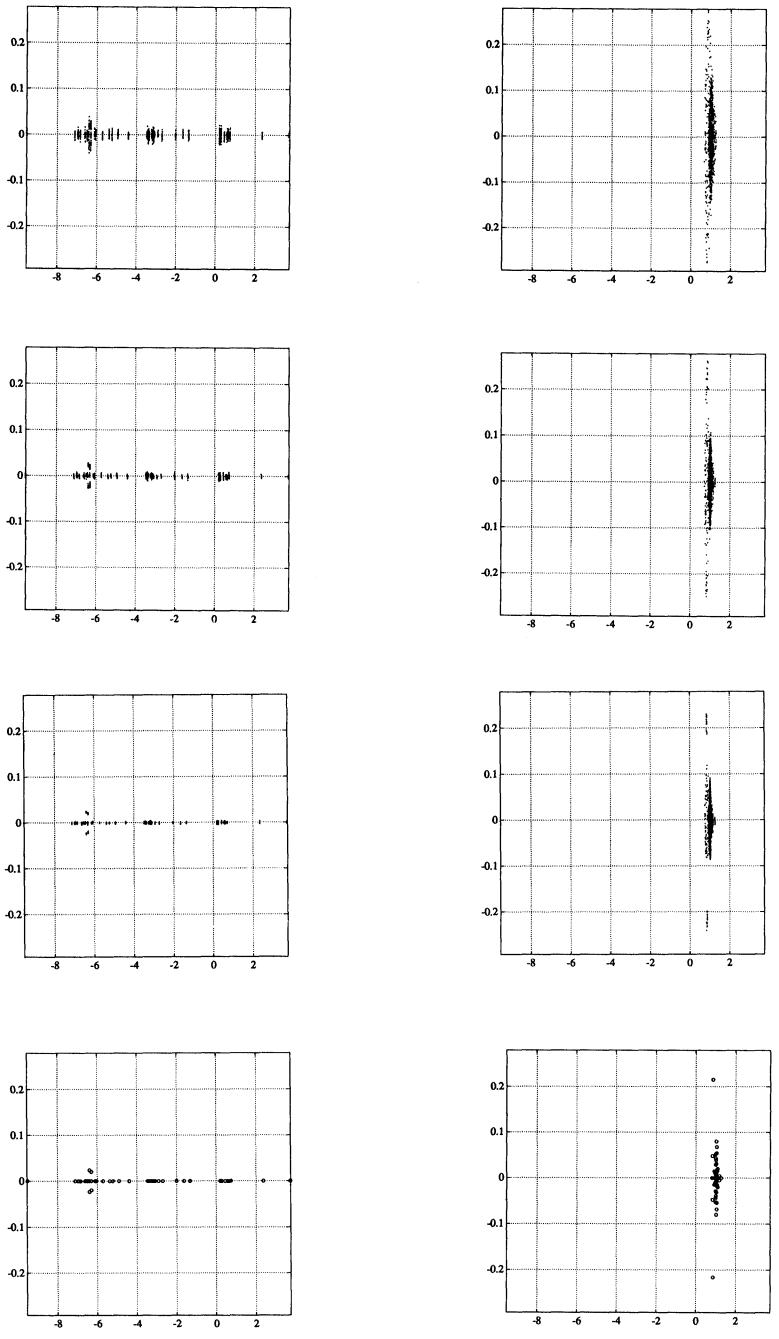

FIG. 5. *A sample of the pseudoeigenvalues of A (left) and PA (right) for L-shaped domain with* $n = 58$. *First row is* $\delta = 0.01$, *second row is* $\delta = 0.005$, *third row is* $\delta = 0.0025$, *fourth row is* $\delta = 0$. *The x and y axes (scaled differently) are the real and imaginary parts, respectively.*

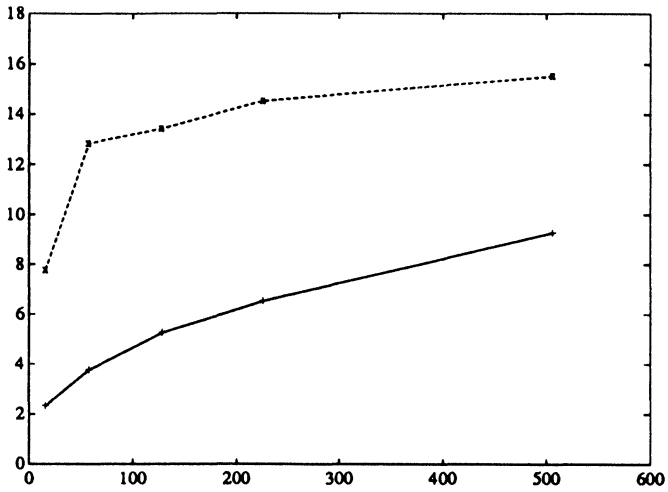

FIG. 6. *Solid line indicates $\kappa_2(PA)$ plotted against n; dashed line indicates $\kappa_2(A)/\kappa_2(PA)$ plotted against n.*

In this formula, Λ_0 denotes the spectrum of A, V denotes the matrix of eigenvectors of A, and $\kappa_2(V)$ denotes the condition number of V. Thus, this formula predicts better convergence when the exact eigenvalues are clustered away from the origin. As illustrated in the figure, the clustering is improved for PA. On the other hand, the factor $\kappa_2(V)$ is worse for PA: for this example, the condition numbers of the eigenvectors for A and PA are 6.19 and 328.37, respectively. Thus, the pseudoeigenvalue bound for the convergence of GMRES seems to be more predictive than the eigenvalue bound for this example.

A final experiment carried out was to compute $\kappa_2(PA)$ and $\kappa_2(A)/\kappa_2(PA)$ for the same shape domain (domain D_2) but varying numbers of nodes. Here, P is preconditioner MN. The results are illustrated in Fig. 6. We notice that the factor improvement in condition number increases as n increases, but so does the condition number of $\kappa_2(PA)$.

5. Analysis of a simple model problem. Let $\Omega \subset \mathbb{R}^2$ be the disk of radius ρ. We assume $\rho \neq 1$ (when $\rho = 1$, the method breaks down). We will try to solve Laplace's equation with Dirichlet boundary data on this region. We provide an asymptotic analysis of the condition number of the dense matrix and of preconditioner MN. We will argue that there is an asymptotic reduction of $\sqrt{\ln n}$ in the condition number. This reduction appears to be pessimistic compared to our three-dimensional computational experiments reported in the last section.

We define three operators on the space of real- or complex-valued functions on $\partial\Omega$, namely, **A**, **B**, and **I**. These operators can be defined for certain normed vector spaces, but the details of the particular function spaces are not important for the analysis in this section.

The three operators are as follows. Let the $\partial\Omega$ be parameterized by

$$\gamma(\theta) = \rho(\cos\theta, \sin\theta).$$

Let f be a complex-valued function on $\partial\Omega$. The function $\mathbf{A}(f)$ evaluated at point $\gamma(t)$ is defined to be

$$\mathbf{A}(f)(\gamma(t)) = -\frac{\rho}{4\pi}\int_0^{2\pi} \ln(\|\gamma(t)-\gamma(s)\|^2)\cdot f(\gamma(s))\,ds. \tag{3}$$

The kernel of this integral operator is the Green's function for a two-dimensional Laplace equation.

Let $\mathbf{B}$ be the following integral operator:

$$\mathbf{B}(f)(\gamma(t)) = \frac{\rho}{2\pi}\int_0^{2\pi} \frac{\langle \gamma(t)-\gamma(s), (\cos s, \sin s)\rangle}{\|\gamma(t)-\gamma(s)\|^2}\cdot f(\gamma(s))\,ds.$$

The kernel here is the normal derivative of Green's function. The notation $\langle\cdot,\cdot\rangle$ denote the inner product on $\mathbb{R}^2$.

Finally, $\mathbf{I}$ denotes the identity operator. Then the following result holds if f is the restriction to $\partial\Omega$ of a solution to Laplace's equation on Ω:

$$(\mathbf{I}+\mathbf{B})(f) = \mathbf{A}\left(\frac{\partial f}{\partial n}\right). \tag{4}$$

Here, $\partial f/\partial n$ denotes the derivative of the function extended to Ω. This identity is the basis for the boundary integral equation method in two dimensions.

In the version of the the boundary element equation method described earlier, functions on $\partial\Omega$ are approximated as a piecewise linear function with n equally spaced breakpoints around $\partial\Omega$. We assume the breakpoints are $v_0,\cdots,v_{n-1}$, where

$$v_j = \gamma(2\pi j/n).$$

A matrix A is created as a discretized representation of $\mathbf{A}$. The discretization procedure is as follows. Operator $\mathbf{A}$ is applied to the basis of piecewise linear "hat" functions. Denote this basis $b_0,\cdots,b_{n-1}$. These basis functions have the property that $b_k(\gamma(\theta))$ is a piecewise linear function of θ. In addition, the b_k's are continuous and satisfy

$$b_k(v_j) = \delta_{jk},$$

where the right-hand side is the Kronecker δ notation. The (j,k) entry of matrix A is then defined to be the function $\mathbf{A}(b_k)$ evaluated at v_j. In general, the entries of A are obtained by numerical integration and so would not be exactly equal to $\mathbf{A}(b_k)(v_j)$, but we assume that a sufficiently high-order quadrature method is used that the truncation error can be ignored.

We can conclude that A will be circulant and symmetric. It is circulant because $\mathbf{A}(b_k)(v_j)$ is the same as $\mathbf{A}(b_{k+1})(v_{j+1})$ (where b_n, v_n are identified with b_0, v_0); this follows from (3). The symmetry of A follows because $\mathbf{A}(b_k)(v_j) = \mathbf{A}(b_j)(v_k)$.

In the boundary element method for Ω given Dirichlet data, we have to solve a system involving matrix A. Accordingly, the goal of this section is to determine how the preconditioner affects the condition number of A.

Since A is circulant, the eigenvectors of A are the discrete Fourier vectors, that is, vectors $\mathbf{w}_0,\cdots,\mathbf{w}_{n-1}$, where the kth component of $\mathbf{w}_j$ is $\exp(2\pi ijk/n)$. We number the components of $\mathbf{w}_j$ with subscripts $0,\cdots,n-1$.

We next attempt to determine the eigenvalue of A corresponding to $\mathbf{w}_j$. Let ψ_j be the piecewise linear function on $\partial\Omega$ corresponding to $\mathbf{w}_j$, that is,

$$\psi_j = \sum_{k=0}^{n-1} w_{jk} b_k.$$

The function $\psi_j(\gamma(\theta))$ is a piecewise linear approximation to the Fourier function $\exp(ij\theta)$ defined on $\partial\Omega$ and has been studied in the literature. In particular, Chandler and Sloan [7] have provided a Fourier series expansion for ψ_j for $j \neq 0$:

$$\psi_j(\gamma(\theta)) = a_j \sum_{m \equiv j} \frac{\sin(\pi m/n)^2 \exp(im\theta)}{(\pi m/n)^2}.$$

Here, the sum is over integers m congruent to j mod n. The number a_j is a scaling constant. Their results also cover higher-order splines.

Note that the eigenvalue for $\mathbf{w}_j$ is equal to the ratio of $\mathbf{A}(\psi_j)$ evaluated at a collocation point divided by ψ_j evaluated at the same collocation point (provided ψ_j is not zero at the collocation point). Since $\mathbf{A}$ is linear, we have:

$$\mathbf{A}(\psi_j) = a_j \sum_{m \equiv j} \frac{\sin(\pi m/n)^2 \mathbf{A}(e_m)}{(\pi m/n)^2},$$

where e_m is the Fourier function $\exp(im\theta)$ defined on the circle of radius ρ.

Thus, the next task is to evaluate $\mathbf{A}(e_m)$. We first treat the case in which $m \neq 0$. We notice that

$$\mathbf{A}(e_m)(\gamma(t)) = -\frac{\rho}{4\pi} \int_0^{2\pi} \ln(\|\gamma(t) - \gamma(s)\|^2) \cdot \exp(ims)\, ds. \tag{5}$$

This integral appears to be difficult to analyze directly, so we instead approach it via identity (4). Focus on the case in which m is a positive integer. We notice that $\exp(ims)$ on the circle of radius ρ can be extended to all of $\mathbb{R}^2$ as the complex analytic function $f(z) = z^m/\rho^m$. Here, we are identifying $\mathbb{R}^2$ with $\mathbb{C}$. Notice also that f, since it is analytic, is a Laplace solution. The normal derivative of f is given by $\partial f/\partial n = me_m/\rho$ for points on the circle of radius ρ. Thus, from (4), we have

$$\mathbf{A}(e_m) = (\rho/m)(\mathbf{I} + \mathbf{B})(e_m).$$

Let us now evaluate the right-hand side:

$$\begin{aligned}
\mathbf{B}(e_m)(\gamma(t)) &= \frac{\rho}{2\pi} \int_0^{2\pi} \frac{\rho(\cos t - \cos s)\cos s + \rho(\sin t - \sin s)\sin s}{\rho^2(\cos t - \cos s)^2 + \rho^2(\sin t - \sin s)^2} \cdot e^{ims}\, ds \\
&= \frac{1}{2\pi} \int_0^{2\pi} \frac{\cos(s-t) - 1}{2 - 2\cos(s-t)} \cdot e^{ims}\, ds \\
&= \frac{1}{2\pi} \int_0^{2\pi} \left(-\tfrac{1}{2}\right) e^{ims}\, ds \\
&= 0
\end{aligned}$$

as long as $m \neq 0$. This is because the real and imaginary parts of e^{ims} go through an integer number of periods over the range of integration. Thus, for positive integers m, we have $\mathbf{A}(e_m) = (\rho/m)\mathbf{I}(e_m)$, i.e.,

$$\mathbf{A}(e_m) = (\rho/m)e_m. \tag{6}$$

The preceding analysis fails for negative integers because z^m is not analytic at zero if $m < 0$. The eigenvalue of e_m for $m < 0$ is seen to be $|m|$; this follows by taking the complex conjugate of (6).

Thus, we conclude for $m \neq 0$ that $\mathbf{A}(e_m) = (\rho/|m|)e_m$. This fact appears also in [7] and is well known in the literature.

The last case to analyze is $m = 0$:

$$\begin{aligned}\mathbf{A}(e_0)(\gamma(t)) &= -\frac{\rho}{4\pi}\int_0^{2\pi} \ln(\rho^2(2-2\cos(s-t)))\,ds \\ &= -\frac{\rho}{4\pi}\int_0^{2\pi} \ln(\rho^2)\,ds - \frac{\rho}{4\pi}\int_0^{2\pi} \ln(2-2\cos s)\,ds \\ &= -\rho\ln(\rho^2)/2.\end{aligned}$$

The second integral on the right-hand side of the second line vanishes, as can be verified with Fourier expansion. Notice that the answer is a constant (independent of t).

Now, we return to the analysis of matrix A. We see now that

$$\mathbf{A}(\psi_j)(\gamma(t)) = a_j \sum_{m\equiv j} \frac{\sin(\pi m/n)^2 e^{imt}\rho}{|m|\cdot(\pi m/n)^2}$$

for $j = 1, \cdots, n-1$. We return to the case when $j = 0$ below.

Recall that the eigenvalue of A corresponding to $\mathbf{w}_j$ is equal to the ratio

$$\mathbf{A}(\psi_j)(v_0)/\psi_j(v_0)$$

provided that $\psi_j(v_0) \neq 0$. The numerator $\mathbf{A}(\psi_j)(v_0)$ is found using the formula in the last paragraph:

$$a_j \sum_{m\equiv j} \frac{\sin(\pi m/n)^2\rho}{|m|\cdot(\pi m/n)^2},$$

which simplifies to

$$\frac{a_j \sin(\pi j/n)^2 \rho n^2}{\pi^2} \sum_{m\equiv j} (1/|m|^3). \tag{7}$$

The preceding summation can be analyzed by splitting it into positive and negative indices, and dividing each summation by its leading term. The result is

$$\sum_{m\equiv j} \frac{1}{|m|^3} = \tau_j \left(\frac{1}{j^3} + \frac{1}{(n-j)^3}\right),$$

where τ_j lies between 1 and 1.3.

The formula for $\psi_j(v_0)$ is

$$a_j \sum_{m\equiv j} \frac{\sin(\pi m/n)^2}{(\pi m/n)^2}.$$

This simplifies to

$$\frac{a_j \sin(\pi j/n)^2 n^2}{\pi^2} \sum_{m\equiv j} (1/m^2). \tag{8}$$

TABLE 4
Condition numbers of A for a disk with $\rho = 2$.

n	$\kappa_2(A)$
10	8.89
20	17.69
40	35.31
80	70.54

The summation in the previous expression can be written

$$\tau_j' \left(\frac{1}{j^2} + \frac{1}{(n-j)^2} \right),$$

where τ_j' lies between 1 and 1.7.

Thus, dividing (7) by (8) gives the eigenvalue of $\mathbf{w}_j$. Simplifying gives

$$A\mathbf{w}_j = \frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1 - 3z + 3z^2}{z(1-z)(1-2z+2z^2)} \mathbf{w}_j, \tag{9}$$

where $z = j/n$.

We next determine the eigenvalue of A for $\mathbf{w}_0$. Notice that $\mathbf{w}_0$ is the vector of all 1s. The value of $A\mathbf{w}_0$ is the collocated values of $\mathbf{A}(\psi_0)$, which is the same as $\mathbf{A}(e_0)$. We showed earlier that $\mathbf{A}(e_0)$ is the constant function with value $-\rho \ln(\rho^2)/2$. Thus we see that the eigenvalue of A for $\mathbf{w}_0$ is independent of n and equal to this constant.

We now have enough information to determine the asymptotic condition number of A. Since A is symmetric, its condition number is the ratio of the extreme magnitudes of its eigenvalues. The second fraction in (9) may be written as

$$\frac{1 - 3y}{y(1-2y)}, \tag{10}$$

where $y = z(1-z)$. Formula (10) is seen to be positive and decreasing for $0 < y < 0.25$ (since z is between zero and 1, y must be between zero and 0.25). Thus (10) is maximized when y is small, i.e., when z is close to zero or 1, i.e., when j is 1 or $n-1$. In this case the second fraction of (9) behaves asymptotically like n for large n. The first fraction behaves like $1/n$. Thus, the largest eigenvalue is for $\mathbf{w}_1$ or $\mathbf{w}_{n-1}$ and has value that is a constant. The smallest eigenvalue occurs when y in (10) is largest, i.e., when z is close to $\frac{1}{2}$, i.e., when j is close to $n/2$. Assume n is even so that $n/2$ is an integer. Then we see that the second fraction of (9) is 2, so the eigenvalue is proportional to $1/n$.

Thus, we see that the largest eigenvalue of A is at $\mathbf{w}_0$ or $\mathbf{w}_1$ and is asymptotically bounded above and below by constants independent of n. The smallest eigenvalue of A behaves like $1/n$. Thus, the condition number of A is expected to be $O(n)$ asymptotically. In fact, this condition number result was established more generally by Richter [19] in the context of Galerkin-type discretizations.

These assertions are borne out by our computational experience; Table 4 indicates the condition number of A in the case $\rho = 2$ for varying values of n.

Next, we turn to preconditioner MN. Let us call the preconditioning matrix P. In the problem on the disk, each node has two nearest neighbors; therefore, each row of P will have three nonzero entries. Each row is determined by solving a 3×3 linear system.

This linear system will be the same for each node $v_0, \cdots, v_{n-1}$ by symmetry, so P will also be circulant with three nonzero bands clustered around the diagonal. In addition, the 3×3 system is symmetric since A is symmetric.

Let the 3×3 system be denoted by $\bar{A}$; it is the principal submatrix of A formed by taking three consecutive rows and columns of A. Thus, it has the form

$$\bar{A} = \begin{pmatrix} \mu & \nu & \xi \\ \nu & \mu & \nu \\ \xi & \nu & \mu \end{pmatrix}.$$

The next task is to determine the values of μ, ν, and ξ. This is equivalent to determining formulas for the entries of A in positions $(1,1)$, $(1,2)$, and $(1,3)$. Let $f(z) = \mathbf{A}(b_0)(z)$; then our problem is to determine $f(v_0), f(v_1), f(v_2)$ (recall that b_0 denotes the piecewise linear hat function nonzero at v_0, and v_0, v_1, v_2 are breakpoints).

We have the following calculation:

$$\begin{aligned} \mathbf{A}(b_0)(v_p) &= -\frac{\rho}{4\pi} \int_{-\pi}^{\pi} \ln(\|v_p - \gamma(s)\|^2) \cdot b_0(\gamma(s))\, ds \\ &= -\frac{\rho}{4\pi} \int_{-2\pi/n}^{2\pi/n} \ln\left(\rho \,\|(\cos(2\pi p/n), \sin(2\pi p/n)) - (\cos s, \sin s)\|\right)^2 \\ &\quad \times b_0(\gamma(s))\, ds. \end{aligned}$$

In this formula, $p = 0, 1, 2$. The second line follows from the fact that $b_0(\gamma(s))$ is nonzero only for $s \in [-2\pi/n, 2\pi/n]$.

Next, notice that

$$\|(\cos(2\pi p/n), \sin(2\pi p/n)) - (\cos s, \sin s)\| = |2\pi p/n - s| \cdot \chi_p(s),$$

where $\chi_p(s) \in [1 - O(1/n^3), 1 + O(1/n^3)]$; this follows because $(\cos(2\pi p/n), \sin(2\pi p/n))$ and $(\cos s, \sin s)$ are two nearby points on the unit circle (recall $p = 0, 1, 2$). Thus we have

$$\begin{aligned} \mathbf{A}(b_0)(v_p) &= -\frac{\rho}{4\pi} \int_{-2\pi/n}^{2\pi/n} \ln\left(\rho \chi_p(s) \cdot |2\pi p/n - s|\right)^2 \cdot b_0(\gamma(s))\, ds \\ &= -\frac{\rho}{2\pi} \int_{-2\pi/n}^{2\pi/n} (\ln \rho + \ln \chi_p(s) + \ln |2\pi p/n - s|) \cdot b_0(\gamma(s))\, ds \\ &= -\frac{\rho}{2\pi}(i_1 + i_2). \end{aligned}$$

Here,

$$\begin{aligned} i_1 &= \int_{-2\pi/n}^{2\pi/n} (\ln \rho + \ln \chi_p(s)) b_0(\gamma(s))\, ds \\ &= (\ln \rho + O(1/n^3)) \frac{2\pi}{n}. \end{aligned}$$

The other term is

$$i_2 = \int_{-2\pi/n}^{2\pi/n} \ln |2\pi p/n - s| \cdot b_0(\gamma(s))\, ds.$$

The integral in the previous equation is the product of a logarithm and a piecewise linear function, which can be integrated analytically. The result is that $i_2 = \delta \ln \delta + c_p \delta$, where $\delta = 2\pi/n$ and $c_0 = -1.5$, $c_1 \approx -0.1137$, and $c_2 \approx 0.6712$.

Thus

$$\mathbf{A}(b_0)(v_p) = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_p + O(1/n^3)).$$

This gives us a formula for the entries of $\bar{A}$. In particular,

$$\mu = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_0 + O(1/n^3)),$$
$$\nu = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_1 + O(1/n^3)),$$
$$\xi = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_2 + O(1/n^3)).$$

The linear system we solve to recover the entries of a row of P is $\bar{A}\mathbf{x} = \mathbf{e}_2$, where $\mathbf{e}_2 = (0,1,0)^T$. We solve this with Cramer's rule. Let d be equal to $1/\det(\bar{A})$; then the solution to $\bar{A}\mathbf{x} = \mathbf{e}_2$ is $(x_1, x_2, x_3)^T$, where

$$x_1 = d(\xi\nu - \mu\nu),$$
$$x_2 = d(\mu^2 - \xi^2),$$
$$x_3 = d(\xi\nu - \mu\nu).$$

We can simplify these expressions; let $d' = (\rho\delta/(2\pi))^2$ and $d'' = \ln\rho + \ln\delta$; then

$$x_1 = x_3 = dd'(d''(c_2 - c_0) + c_1(c_2 - c_0) + d''O(1/n^3))$$

and

$$x_2 = dd'(2d''(c_0 - c_2) + c_0^2 - c_2^2 + d''O(1/n^3)).$$

This gives the entries of P. Notice that

$$x_2/x_1 = -2 + \frac{-c_0 - c_2 + 2c_1 + d''O(1/n^3)}{d'' + c_1 + d''O(1/n^3)}.$$

Recall that $d'' = \ln(\delta) + O(1) = -\ln(n) + O(1)$. Notice also that $-c_0 - c_2 + 2c_1$ is a positive constant approximately 0.6014; call this constant $d^{(3)}$. Thus,

$$x_2/x_1 = -2 - \frac{d^{(3)} + O(\ln(n)/n^3)}{\ln(n) + O(1)}$$
$$= -2 - \frac{d^{(3)}}{\ln n} + O((\ln n)^{-2}).$$

Thus, P/x_1 is a circulant symmetric matrix with $-2 - d^{(3)}/(\ln n) + O((\ln n)^{-2})$ on the diagonal and 1 on the positions adjacent to the diagonal (and in the $(n,1)$ and $(1,n)$ positions).

The next task is to compute the condition number of PA. This is the same as the condition number of PA/x_1, so we work with PA/x_1 instead. Since P is circulant and symmetric, it has $\mathbf{w}_0, \cdots, \mathbf{w}_{n-1}$ as eigenvectors. Therefore, PA/x_1 is also circulant and symmetric and also has $\mathbf{w}_0, \cdots, \mathbf{w}_{n-1}$ as eigenvectors. Accordingly, we now attempt to compute the eigenvalue of $\mathbf{w}_j$ in PA/x_1. This is the same as the product of the eigenvalue of $\mathbf{w}_j$ in P/x_1 multiplied by the eigenvalue in A.

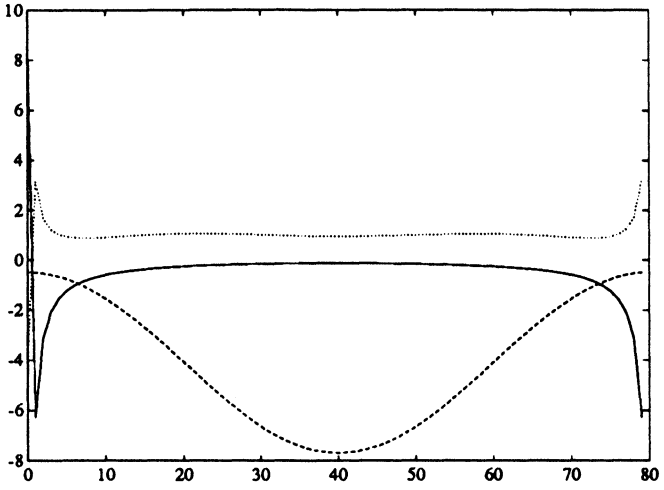

FIG. 7. *The eigenvalues of A, P, and PA plotted against frequency.*

The kth component of $P\mathbf{w}_j/x_1$ (counting from zero) is equal to

$$\begin{aligned}(P\mathbf{w}_j/x_1)_k &= e^{2\pi ij(k-1)/n} + (-2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n} + e^{2\pi ij(k+1)/n} \\ &= (e^{2\pi ij/n} + e^{-2\pi ij/n} - 2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n} \\ &= (2\cos(2\pi j/n) - 2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n} \\ &= (-4\sin^2(\pi j/n) - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n}.\end{aligned}$$

Thus, the eigenvalue of P/x_1 for $\mathbf{w}_j$ is

$$-4\sin^2(\pi j/n) - d^{(3)}/(\ln n) + O((\ln n)^{-2}).$$

We can rewrite the first term; in particular, there exists a number τ_j'' lying between π and 4 such that

$$\sin(\pi j/n) = \tau_j''(j/n)(1 - j/n)$$

for $j = 0, \cdots, n-1$. Thus the eigenvalue can be written

$$-4\tau_j''^2 z^2(1-z)^2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}), \tag{11}$$

where $z = j/n$.

Figure 7 shows a plot of the magnitude of the eigenvalue versus j produced by MATLAB. There are three sets of eigenvalues in the plot. The solid line is the eigenvalue in A; the dashed line is the eigenvalue in P; and the dotted line is the eigenvalue in PA. Notice how the eigenvalues of P are of large magnitude roughly in the places where the eigenvalues of A are small. This is the desired feature of the preconditioner. Note that these plots include some scaling factors not described above.

Continuing our analysis, the eigenvalue of $\mathbf{w}_j$ for PA/x_1 for $1 \le j \le n-1$ is given by the product of the eigenvalues in (9) and (11). This is

$$\lambda_j = \frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1 - 3z + 3z^2}{z(1-z)(1-2z+2z^2)} \cdot \left(-4\tau_j''^2 z^2(1-z)^2 - \frac{d^{(3)}}{\ln n} + O((\ln n)^{-2})\right),$$

where $z = j/n$. As above, we can substitute $y = z(1-z)$. Also, let $q_n = d^{(3)}/(\ln n) + O((\ln n)^{-2})$. We obtain

$$\lambda_j = \frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1-3y}{y(1-2y)} \cdot \left(-4\tau_j''^2 y^2 - q_n\right). \tag{12}$$

For n large enough, q_n is positive so the whole expression is negative. Therefore, it suffices to estimate the largest and smallest eigenvalues in order to estimate the condition number. We break (12) into two terms; the first term is

$$\frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1-3y}{1-2y} \cdot (-4\tau_j''^2 y).$$

This number is seen to lie between $\sigma_1 y/n$ and $\sigma_2 y/n$, where σ_1, σ_2 are two negative constants, provided y is between zero and 0.25 (so that $(1-3y)/(1-2y)$ is between $\frac{1}{2}$ and 1).

The second term is

$$\frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1-3y}{y(1-2y)} \cdot (-q_n).$$

Reasoning as in the previous paragraph, we conclude that this lies between $\sigma_3 q_n/(ny)$ and $\sigma_4 q_n/(ny)$ where σ_3, σ_4 are negative constants.

Thus, it suffices to establish upper and lower bounds on functions of the form

$$\sigma' y/n + \sigma'' q_n/(ny), \tag{13}$$

where σ', σ'' can lie between fixed negative bounds. Assume a fixed choice of σ', σ''. Formula (13) is a concave function of y. This means its lower bound is achieved at one of the endpoints. The first endpoint occurs when y is as small as possible, i.e., $j = 1$ or $j = (n-1)$, giving $z = 1/n$, and $z = (n-1)/n$, giving $y = (n-1)/n^2$, which is asymptotically like $1/n$. Plugging $1/n$ into (13) gives an eigenvalue asymptotically like $\sigma'' q_n$ for large n. The other endpoint occurs when y is as large as possible, i.e., $j = n/2$, in which case $z = \frac{1}{2}$ and $y = \frac{1}{4}$. Plugging this in gives an eigenvalue asymptotically like $\sigma'/(4n)$, which is larger than the eigenvalue at 1. (Recall σ', σ'' are negative and $q_n = O(1/(\ln n))$.) Thus, the smallest eigenvalue has the form $\sigma'' q_n$.

The largest value of (13) occurs where the derivative with respect to y is zero. Thus, the largest eigenvalue comes at

$$y_0 = \sqrt{\frac{\sigma'' q_n}{\sigma'}}.$$

We remark that for n large enough, y_0 will lie between zero and 0.25 and hence is achieved asymptotically for some value of j. At y_0, formula (13) has the value $-2\sqrt{\sigma'\sigma'' q_n}/n$. Therefore, this is the largest eigenvalue of PA/x_1 up to the choice of σ', σ''.

This means that the eigenvalues run from $O(1/(\ln n))$ down to $O(1/(n\sqrt{\ln n}))$ in absolute value. We have not yet addressed the eigenvalue for $\mathbf{w}_0$. This turns out to be $O(1/(\ln n))$ also.

Thus we obtain the condition number of PA/x_1 by dividing the largest eigenvalue in absolute value by the smallest. This yields a condition number of $O(n/\sqrt{\ln n})$.

Since the condition number of A was $O(n)$, this concludes our proof that for a simple model problem, the condition number of A is reduced by $O(\sqrt{\ln n})$, a factor growing faster than any constant.

6. Complexity issues. Suppose we could establish a constant upper bound on the number of iterations required by a preconditioned iterative method for BIEM. This would imply a running time of $O(n^2)$ to solve the dense system of equations, clearly an improvement over the running time $O(n^3)$ of Gaussian elimination.

In fact, there are several complexity results in this direction. Rokhlin gets an $O(n)$ running time for solving the linear system using multipole expansion. Brandt and Lubrecht [4] achieve an $O(n \log n)$ running time with multigrid. Earlier, Schippers [22] had achieved an $O(n^2)$ running time, also with multigrid. Bramble, Pasciak, and Xu [3] have proposed a multilevel preconditioner for boundary element methods.

It seems to us, however, that none of these methods apply to (1) with mixed boundary conditions, because solving (1) with mixed boundary conditions yields a system of equations that is a mixture of first- and second-kind integral equations. All of the above-mentioned methods assume that the integral equation is homogeneous.

Accordingly, it remains an interesting open problem to establish asymptotic complexity results on how long it takes to solve the BIEM system of linear equations arising from (1).

Acknowledgments. The author thanks Professor Kendall Atkinson for providing some of the background material for this paper and Professor L. N. Trefethen for explaining some properties of unsymmetric iterations.

REFERENCES

[1] K. E. Atkinson, *Piecewise polynomial collocation for integral equations on surfaces in three dimensions*, J. Integral Equations, 9 (1985), pp. 25–48.

[2] M. W. Benson and P. O. Frederickson, *Iterative solution of large sparse linear systems arising in certain multidimensional approximation problems*, Utilitas Math., 22 (1982), pp. 127–140.

[3] J. H. Bramble, J. E. Pasciak, and J. Xu, private communication, 1991. See also *Parallel Multilevel Computation*, Math. Comp., 55 (1990), pp. 1–22, by the same authors.

[4] A. Brandt and A. A. Lubrecht, *Multilevel matrix multiplication and fast solution of integral equations*, J. Comput. Phys., 90 (1990), pp. 348–370.

[5] C. A. Brebbia (ed.), *Topics in Boundary Element Research*, Springer-Verlag, Berlin, 1987.

[6] F. X. Canning, *Sparse approximation for solving integral equations with oscillatory kernels*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 71–87.

[7] G. A. Chandler and I. H. Sloan, *Spline qualocation methods for boundary integral equations*, Numer. Math., 58 (1990), pp. 537–567.

[8] H. C. Elman, *A stability analysis of incomplete LU factorizations*, Math. Comp., 47 (1986), pp. 191–217.

[9] T. Feder and D. H. Greene, *Optimal algorithms for approximate clustering*, Proc. 20th Annual ACM Symposium Theory of Computing, Chicago, IL, 1988, pp. 434–444.

[10] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[11] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press, Cambridge, MA, 1988.

[12] J. L. Hess and A. M. O. Smith, *Calculation of potential flow about arbitrary bodies*, in Progress in Aeronautical Sciences, Vol. 8, D. Küchemann, ed., Pergamon Press, London, 1967, pp. 1–138.

[13] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Stand., 49 (1952), pp. 409–436.

[14] M. A. Jaswon, *Integral equation methods in potential theory*, I, Proc. Royal Soc. Ser. A, 275 (1963), pp. 23–32.

[15] L. Yu. Kolotilina and A. Yu. Yeremin, *On a family of two-level preconditionings of the incomplete block factorization type*, Sov. J. Numer. Anal. Math. Model., 4 (1986), pp. 293–320.

[16] P. L.-F. Liu, H.-W. Hsu, M. H. Lean, and S. A. Vavasis, *A boundary element method for three-dimensional free surface flows*, Report X9100283, Webster Research Center, Xerox Corporation, North Tarrytown, NY, 1991.

[17] T. A. Manteuffel, *Shifted incomplete Cholesky factorization,* in Sparse Matrix Proceedings, 1978, I. S. Duff and G. W. Stewart, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979, pp. 41–61.

[18] N. M. Nachtigal, S. C. Reddy, and L. N. Trefethen, *How fast are nonsymmetric matrix iterations?* SIAM J. Matrix Anal. Appl., this issue, pp. 778–795.

[19] G. R. Richter, *Numerical solution of integral equations of the first kind with unsmooth kernels*, SIAM J. Numer. Anal., 15 (1978), pp. 511–522.

[20] V. Rokhlin, *Rapid solution of integral equations of classical potential theory*, J. Comput. Phys., 60 (1985), pp. 187–207.

[21] Y. Saad and M. H. Schultz, GMRES: *A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[22] H. Schippers, *Theoretical and practical aspects of multigrid methods in boundary element calculations*, in Topics in Boundary Element Research, Vol. 3, Computational Aspects, C. A. Brebbia, ed., Springer-Verlag, Berlin, 1987.

[23] P. Sonneveld, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[24] G. T. Symm, *Integral equation methods potential theory*, II, Proc. Royal. Soc. Ser. A, 275 (1963), pp. 33–46.

[25] L. N. Trefethen, *Pseudospectra of matrices*, Report 91/10, Oxford University Computing Laboratory, Numerical Analysis Group, Oxford, England, August 1991.