JOURNAL OF COMPUTATIONAL PHYSICS 60, 187–207 (1983)

Rapid Solution of Integral Equations of Classical Potential Theory

V. ROKHLIN

Exxon Production Research Company, P. O. Box 2189, Houston, Texas 77001

Received December 20, 1983; revised October 29, 1984

An algorithm is described for rapid solution of classical boundary value problems (Dirichlet an Neumann) for the Laplace equation based on iteratively solving integral equations of potential theory. CPU time requirements for previously published algorithms of this type are proportional to n^2, where n is the number of nodes in the discretization of the boundary of the region. The CPU time requirements for the algorithm of the present paper are proportional to n, making it considerably more practical for large scale problems. © 1985 Academic Press, Inc.

I. INTRODUCTION

When a boundary value problem for the Laplace equation has to be solved numerically, one of two approaches is usually employed. The first approach calls for discretizing the region of interest by means of some finite differences or finite element type scheme with subsequent application of a "fast solver" to the resulting system of linear algebraic equations (see, for example, [5, 7, 11a, 13a]). The most efficient of fast solvers require an amount of storage and CPU time proportional to N, where N is the number of nodes in the discretization of the region. These methods are most suitable for rectangular regions; they are less efficient for regions of more complicated shapes. Algorithms discretizing the whole area of interest are not applicable to exterior boundary value problems, like the ones encountered in aerodynamical calculations (see, for example, [9]). For such problems, and also for interior problems in extremely complicated regions, algorithms based on boundary integrals are usually employed (see [2, 8, 15]). In most of these algorithms, the problem is reduced to a second kind integral equation on the boundary of the region by means of classical potential theory. Discretization of the resulting integral equations leads to large scale systems of linear algebraic equations whch are in turn solved by means of some iterative technique (see [1, 9, 13]). Most iterative schemes for solution of linear systems resulting from classical potential theory require application of the matrix of the system to a sequence of recursively generated vectors. Applying a dense matrix to a vector requires k^2 multiplications and about as many additions where k is the dimension of the system, and the dimension of the system is equal to the number of nodes n in the discretization of the boundary of the region. As a result, the whole process is at least of the order n^2; in a square-

0021-9991/85 $3.00

shaped region $n^2 = 16N$, and this estimate turns out to coincide with that for a conventional fast Poisson solver. In practical computations, however, fast Poisson solvers tend to be substantially more efficient whenever they are applicable.

The present paper describes an algorithm for rapid application of matrices resulting from discretizations of integral equations of classical potential theory. The algorithm requires an amount of work and storage proportional to n, and when it is combined with a generalized conjugate residual type algorithm (see [6]), the resulting process takes very few iterations to converge, resulting in an order n algorithm for the numerical solution of the original integral equation.

When the algorithm of the present paper is applied to a boundary value problem for the Laplace equation, the result of the calculation is a charge or dipole distribution on the boundary of the region. Evaluating the solution at a point inside the region involves additional order n operations resulting in the total CPU time estimate

$$An + Bmn \tag{1.1}$$

where m is the number of points at which the solution is to be evaluated, and A, B are coefficients determined by the physical situation, computer system, language, etc. When m is small, the estimate (1.1) is dominated by the first term, resulting in an extremely fast algorithm. For interior boundary value problems, m is often proportional to n^2, in which case (1.1) becomes substantially inferior to CPU time estimates for fast poisson solvers whenever the shape of the region permits the latter to be applied efficiently (see [5, 13a]).

II. Relevant Mathematical Facts

1. *Boundary Value Problems for the Laplace Equation*

We will be considering the situation depicted in Fig. 1. The open interior of the Jordan curve γ: $[0, L] \to R^2$ will be denoted by Ω. The image of γ will be denoted by Γ, and the closure of Ω will be denoted by $\bar{\Omega}$, so that $\partial\bar{\Omega} = \Gamma$. The curve γ is

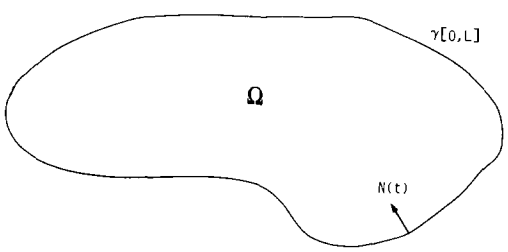

FIG. 1. A boundary value problem in R^2.

presumed to be at least c^2 (i.e., it has at least two continuous derivatives), and is parametrized by its length. The normalized internal normal to γ at $t \in [0, L]$ will be denoted by $N(t)$. For an integrable function f: $[0, L] \to R^1$, we will be solving one of the following four problems:

(A) Interior Dirichlet problem: find a ϕ: $\bar{\Omega} \to R^1$ such that

$$\begin{aligned} \nabla^2 \phi(x) &= 0 && \text{for} \quad x \in \Omega, \\ \phi(x) &= f(\gamma^{-1}(x)) && \text{for} \quad x \in \partial\bar{\Omega}. \end{aligned} \tag{2.1}$$

(B) Exterior Dirichlet problem: find a bounded ϕ: $R^2 \setminus \Omega \to R^1$ such that

$$\begin{aligned} \nabla^2 \phi(x) &= 0 && \text{for} \quad x \in R^2 \setminus \bar{\Omega}, \\ \phi(x) &= f(\gamma^{-1}(x)) && \text{for} \quad x \in \partial\bar{\Omega}. \end{aligned} \tag{2.2}$$

(C) Interior Neumann problem: find a ϕ: $\bar{\Omega} \to R^1$ such that

$$\begin{aligned} \nabla^2 \phi(x) &= 0 && \text{for} \quad x \in \Omega, \\ \frac{\partial \phi}{\partial N} &= f(\gamma^{-1}(x)) && \text{for} \quad x \in \partial\bar{\Omega}. \end{aligned} \tag{2.3}$$

(D) Exterior Neumann problem: find a bounded ϕ: $R^2 \setminus \Omega \to R^1$ such that

$$\begin{aligned} \nabla^2 \phi(x) &= 0 && \text{for} \quad x \in R^2 \setminus \bar{\Omega}, \\ \frac{\partial \phi}{\partial N} &= f(\gamma^{-1}(x)) && \text{for} \quad x \in \partial\bar{\Omega}. \end{aligned} \tag{2.4}$$

In the above problems, ϕ has to be twice differentiable in the interior of its domain, and contnuous on its closure. As is well known, problems (A), (B) have solutions for any integrable f, while the problems (C), (D) have solutions if and only if

$$\int_0^L f(t)\, dt = 0. \tag{2.5}$$

The solution of the problem (A) is unique, and the solution of the problem (B) is unique in the class of bounded functions. For the problem (C), the difference between any two solutions is a constant, and the same is true for the problem (D).

2. *Single and Double Layer Potentials*

The potential of a charge of unit intensity located at the point $x_o \in R^2$ is the function ϕ_{x_o}: $(R^2 \setminus \{x_0\}) \to R^1$ defined by the formula

$$\phi_{x_0}(x) = \log(\|x - x_0\|). \tag{2.6}$$

The potential of a unit strength dipole located at x_0 and oriented in the direction $h \in R^2$ ($\|h\| = 1$) is defined by the formula

$$\phi_{x_0,h}(x) = \frac{\partial}{\partial t}(\phi_{x_0}(x+th))|_{t=0} = \frac{(h(x-x_0))}{\|x-x_0\|^2}. \tag{2.7}$$

For an integrable function σ: $[0, L] \to R^1$, the potential of a single layer with density σ is

$$p_\sigma^0(x) = \int_0^L \phi_{\gamma(t)}(x)\,\sigma(t)\,dt, \tag{2.8}$$

and the potential of a double layer with the dipole density σ is

$$p_\sigma^1(x) = \int_0^L \phi_{\gamma(t),N(t)}(x)\,\sigma(t)\,dt. \tag{2.9}$$

3. *Integral Equations of Classical Potential Theory*

In classical potential theory, the problems (2.1), (2.2) are solved by representing ϕ as a double layer potential, and the problems (2.3), (2.4) are solved by representing ϕ as a single layer potential. Single and double layer potentials are singular on Γ, and the analysis of these singularities yields the following second kind integral equations:

(A1) Interior Dirichlet problem:

$$\pi\sigma(x) + p_\sigma^1(\gamma(x)) = f(x). \tag{2.10}$$

(A2) Exterior Dirichlet problem:

$$-\pi\sigma(x) + p_\sigma^1(\gamma(x)) = f(x). \tag{2.11}$$

(A3) Interior Neumann problem:

$$\pi\sigma(x) + \frac{\partial}{\partial N(x)} p_\sigma^0(\gamma(x)) = f(x). \tag{2.12}$$

(A4) Exterior Neumann problem:

$$-\pi\sigma(x) + \frac{\partial}{\partial N(x)} p_\sigma^0(\gamma(x)) = f(x). \tag{2.13}$$

Equations (2.10) and (2.12) have unique solutions for any integrable f. Equations (2.11), (2.13) have solutions if and only if certain conditions are met (see [12]), and these solutions are not unique. However, the null-spaces of the operators on the left-hand sides of these equations have dimension 1; adding one extra constraint eliminates the non-uniqueness (see, for example, [10]).

The rest of this paper is devoted to solving Eqs. (2.10)–(2.13) numerically.

Remark 2.1. It is easy to show that if the mapping γ has k ($k \geqslant 2$) continuous derivatives, then the kernels in the Eqs. (2.10)–(2.13) have $k-2$ continuous derivatives (see [4]). This observation will be important in the following section.

4. *Harmonic Expansions*

If a function $\phi: R^2 \to R^1$ is harmonic in the closed disk D with the origin at $x \in R^2$ and radius R then there exists a pair of sequences $\{\alpha_k\}$, $\{\beta_k\}$, $k = 0, 1, 2, \ldots$, such that for any $x \in D$,

$$\phi(x) = \sum_{k=0}^{\infty} \rho^k(\alpha_k \cos(k\theta) + \beta_k \sin(k\theta)). \tag{2.14}$$

In the above formula, $\rho = \|x - x_0\|$ and θ is the angle between the vector $x - x_0$ and the x axis. If ϕ is harmonic outside D and bounded at infinity then there exists a pair of sequences $\{\alpha_k\}$, $\{\beta_k\}$ such that for any $x \in R^2 \setminus D$,

$$\phi(x) = \sum_{k=0}^{\infty} \frac{1}{\rho^k}(\alpha_k \cos(k\theta) + \beta_k \sin(k\theta)). \tag{2.15}$$

A detailed derivation of expansions (2.14), (2.15) can be found, for example, in [12].

As is well known, real and imaginary parts of an analytic function $w = u + iv$ of the complex variable $z = x + iy$ are harmonic functions of variables x, y. Conversely, for any harmonic function $u: R^2 \to R^1$, there exists an analytic function $w: C^1 \to C^1$ such that $u(x, y) = \mathrm{Re}(w(x, y))$ (see, for example, [3]). This fact is often used to facilitate computations with harmonic functions, and for the rest of this paper we will make no distinction between points in R^2 and points in C^1. In complex terms, the expressions (2.6), (2.7) assume the form

$$\phi_{x_0}(x) = \mathrm{Re}(\log(x - x_0)), \tag{2.16}$$

$$\phi_{x_0,h}(x) = \mathrm{Re}\left(\frac{h}{x - x_0}\right), \tag{2.17}$$

and the expressions (2.14), (2.15) assume the form

$$\phi(x) = \mathrm{Re}\left(\sum_{k=0}^{\infty} a_k(x - x_0)^k\right), \tag{2.18}$$

$$\phi(x) = \mathrm{Re} \sum_{k=0}^{\infty} \frac{a_k}{(x - x_0)^k}, \tag{2.19}$$

with the sequences of complex numbers $\{a_k\}$, $k = 0, 1, \ldots$, replacing the sequences of real pairs $\{\alpha_k, \beta_k\}$.

III. Relevant Numerical Algorithms

1. *Trapezoidal Quadrature Rules for Periodic Functions*

We will define an n-point quadrature rule η on the interval $[0, L]$ as a finite sequence of pairs $\{x_i, w_i\}$, $i = 1, 2, ..., n$, where $x_i \in [0, L]$ for all $i \in [1, n]$. For a function ϕ: $[0, L] \to R^1$, we will look upon the sum

$$\eta(\phi) = \sum_{i=1}^{n} w_i \phi(x_i) \tag{3.1}$$

as an approximation to the integral

$$\int_0^L \phi(x)\, dx. \tag{3.2}$$

We will say that the family of quadrature formulae $\eta_n = \{x_{ni}, w_{ni}\}$, $i = 1, 2, ..., n$, has a rate of convergence m ($m \geqslant 1$) for the function ϕ: $[0, L] \to R^1$ if there exist such $A > 0$, $N > 0$ that

$$\left| \eta_n(\phi) - \int_0^L \phi(x)\, dx \right| < \frac{A}{n^m} \tag{3.3}$$

for all $n > N$. The n-point trapezoidal quadrature rule is defined by the formulae:

$$\begin{gathered} x_i = (i-1)\frac{L}{n} \qquad \text{for} \quad i = 1, 2, ..., n, \\ w_1 = w_n = \frac{L}{2n}, \end{gathered} \tag{3.4}$$

and $w_i = L/N$ for $i \in [2, n-1]$. As is well known, if ϕ has two continuous derivatives then the trapezoidal quadrature rule converges quadratically for ϕ, i.e., there exist such N, $A < 0$ that

$$\left| \int_0^L (x)\, dx - \eta_n(\phi) \right| < \frac{A}{n^2} \tag{3.5}$$

for any $n > N$.

The following theorem is less widely known. Its standard proof based on the Euler–Maclaurin formula can be found, for example, in [16].

THEOREM 3.1. *Suppose that ϕ: $[0, L] \to R^1$ has k continuous derivatives ($k \geqslant 1$). Suppose further that ϕ and its k derivatives are periodic with the period L. Then the order of convergence of the trapezoidal rule for ϕ is equal to $k + 1$.*

2. *Nyström Algorithm*

In order to solve the integral equation

$$\phi(x) + \int_0^L K(t, x)\, \phi(t)\, dt = f(x), \tag{3.6}$$

the Nyström algorithm associated with an n-point quadrature formula $\eta = \{x_i, w_i\}$, $i = 1, 2, \ldots, n$, replaces (3.6) with a system of linear equations

$$\phi_i + \sum_{j=1}^{n} w_j K(x_j, x_i)\, \phi_j = f(x_i) \tag{3.7}$$

with $i = 1, 2, \ldots, n$. We will denote the matrix of the system (3.7) by A_n. The solution $\phi_1, \phi_2, \ldots, \phi_n$ of this system will be looked upon as an approximation to the solution of (3.6) at the nodes x_i, $i = 1, 2, \ldots, n$. If (3.6) has a unique solution then for a wide class of quadrature formulae η_n the system (3.7) also has a unique solution, as long as n is sufficiently large. Furthermore, under fairly broad assumptions, the convergence rate of the Nyström algorithm is the same as the convergence rate of the quadrature formula it is based on (see [1]).

The following theorem is the principal justification for using the generalized conjugate residual algorithm (see next subsection) for the solution of the system (3.7). It can be found in [1].

THEOREM 3.2. *Suppose that* $K: [0, L] \times [0, L] \to R^1$ *is a* c^2*-function and the Eq.* (3.6) *has a unique solution. Suppose further that the system of linear equations* (3.7) *has been obtained by applying the Nyström algorithm based on the trapezoidal quadrature rule to* (3.6). *Then*

$$\lim_{n \to \infty} k(A_n) = a, \tag{3.8}$$

where $0 < a < \infty$ *is some real number and* $k(A_n)$ *denotes the condition number of the matrix* A_n.

3. *Iterative Solution of Linear Systems*

Discretization of Eqs. (2.10)–(2.13) by means of Nyström algorithm leads to large scale systems of linear algebraic equations. These systems are nonsparse and nonsymmetric, but have asymptotically limited condition numbers due to Theorem 3.2. For such systems, the generalized conjugate residual algorithm (GCRA) is known to converge rapidly (see [6] and [17]). In the present paper, we will need two facts concerning GCRA.

Suppose that GCRA is applied to the linear system

$$Ax = y. \tag{3.9}$$

Then

(A) The number of operations required for the first k steps of the GCRA is equal to

$$akN + bk^2N + ckQ, \tag{3.10}$$

where n is the dimension of the system (3.9), Q is the cost of applying the matrix A to a vector and a, b, c are coefficients determined by the computer system, particular implementation of the algorithm, language, etc.

B. On each step of the GCRA the residual in the approximation of the solution (i.e., $\|Ax - y\|_2$) decreases at least by the factor

$$\frac{1-k(A)}{1+k(A)}. \tag{3.11}$$

Combining (3.8) and (3.11), we obtain the following theorem.

THEOREM 3.3. *Suppose that* K: $[0, L] \times [0, L] \to R^1$ *is a* c^2*-function and the Eq.* (3.6) *has a unique solution. Suppose further that the system of linear equations* (3.7) *has been obtained by applying the Nyström algorithm based on the trapezoidal rule to* (3.6). *Then for every* $\varepsilon > 0$ *there exist* $N > 0$ *and* $M > 0$ *such that for any* $n \geqslant N$, *the GCRA will solve* (3.7) *to a relative accuracy* ε *in no more than* M *iterations.*

Remark 3.1. The estimate (3.11) is a very pessimistic one and usually, GCRA converges much faster, especially for linear systems resulting from discretizations of second kind integral equations. In [16a], a considerably stronger estimate for the conjugate gradient algorithm is proven. Probably, a similar result can be obtained for GCRA.

IV. AN ALGORITHM OF ORDER n^2

In this section, we will describe a standard algorithm for the solution of boundary value problems for the Laplace equation in two dimensions. In the following three sections, this algorithm will be drastically speeded up. We will be discussing the solution of (2.10). The algorithms dealing with (2.11), (2.12) and (2.13) are quite similar.

We will apply to (2.10) the Nyström algorithm based on the trapezoidal quadrature rule. Discretizing (2.10) at n equispaced nodes x_i, $i = 1, 2, \ldots, n$, we obtain the system of linear equations

$$\pi\sigma_i + \sum_{j=1}^{n} a_{ij}\sigma_j = f_i \tag{4.1}$$

with $i = 1, 2, \ldots, n$ and

$$f_i = f(t_i), \qquad h = \frac{L}{n},$$
$$a_{ij} = h\phi_{\gamma(t_j),N(t_j)}(t_i). \tag{4.2}$$

The linear system (4.1) is solved by means of a generalized conjugate residual algorithm, and after the solution $\{\sigma_i\}$ $(i = 1, 2, \ldots, n)$ is obtained, the field at any point $x \in \Omega$ is approximated by the sum

$$\phi(x) \sim h \sum_{i=1}^{n} \sigma_i \phi_{\gamma(t_i),N(t_i)}(x). \tag{4.3}$$

The system (4.1) is known to be well conditioned (see [1, 10]), and experience shows that the conjugate residual process converges for such systems extremely well: 6-digit precision is rarely achieved in more than 15 iterations. However, the matrix $\{a_{ij}\}$, $i, j = 1, 2, \ldots, n$, is dense, and one iteration of the conjugate residual process requires more than n^2 multiplications and about as many additions.

The following three sections are devoted to developing a fast algorithm for applying the matrix of the linear system (4.1) to a vector or, equivalently, for computing the field created on a curve by a dipole distribution on that curve.

V. Evaluation of Harmonic Expansions

The following four lemmas constitute the principal analytical tool of the present paper. Their proof consists of expanding the expressions (2.18), (2.19) into Taylor series with respect to the variables z, z_0.

Lemma 5.1. *Suppose that the expansion*

$$\phi(z) = \sum_{k=1}^{\infty} \frac{a_k}{(z - z_0)^k} \tag{5.1}$$

converges outside the circle D of radius R with the center at z_0. Then for z outside the circle D_1 of radius $R + |z_0|$ and the center at 0,

$$\phi(z) = \sum_{m=1}^{\infty} \frac{b_m}{z^m} \tag{5.2}$$

with

$$b_m = \sum_{k=1}^{m} a_k z_0^{m-k} \binom{m-1}{k-1}, \tag{5.3}$$

where $\binom{m}{k}$ *are binomial coefficients. Furthermore, there exists an* $\alpha > 0$ *such that*

$$\left| \phi(z) - \sum_{m=1}^{p} \frac{b_m}{z^m} \right| < \alpha \left(\frac{|z_0| + R}{z} \right)^{p+1} \tag{5.4}$$

for any $p \geqslant 1$.

LEMMA 5.2. *Suppose that the expansion* (5.1) *converges inside the circle* D_2 *with center at* 0 *and radius* $R_0 < |z_0|$. *Then inside* D_2,

$$\phi(z) = \sum_{m=0}^{\infty} b_m z^m \tag{5.5}$$

with b_m *defined by the formula*

$$b_m = \frac{1}{z_0^m} \sum_{k=0}^{\infty} \frac{a_k}{z_0^k} \binom{m-k-1}{k-1} (-1)^k. \tag{5.6}$$

Furthermore, there exists $\alpha > 0$ *such that*

$$\left| \phi(z) - \sum_{m=0}^{p} b_m z^m \right| < \alpha \left| \frac{z}{z_0} \right|^{p+1} \tag{5.7}$$

for any $p \geqslant 1$.

LEMMA 5.3. *If* $z_0, z \in C^1$ *and* $|z| > |z_0|$ *then*

$$\operatorname{Log}(z - z_0) = \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k} \left(\frac{z_0}{z} \right)^k \tag{5.8}$$

and

$$\frac{h}{z - z_0} = h \sum_{k=0}^{\infty} z_0^k \frac{1}{z^{k+1}}. \tag{5.9}$$

LEMMA 5.4. *For any complex* z_0, z *and* $\{a_k\}$, $k = 1, 2, \ldots, n$,

$$\sum_{k=0}^{n} a_k (z - z_0)^k = \sum_{k=0}^{n} a_k \sum_{m=0}^{k} \binom{k}{m} z_0^{k-m} z^k (-1)^{k-m}. \tag{5.10}$$

Lemmas 5.1–5.4 can be used for rapid approximate evaluation of the fields of distributions of charges and dipoles. In the following two sections, we will describe an algorithm for approximate evaluation of integral operators in (2.10)–(2.13) based on these lemmas.

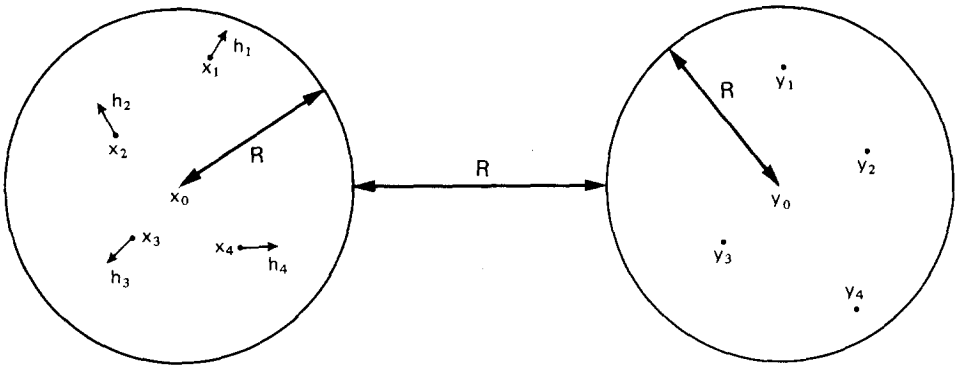

FIG. 2. Separated sets in R^2.

VI. HEURISTIC DESCRIPTION OF THE ALGORITHM

We will illustrate the idea of the algorithm of the present paper with a simple example. Suppose that dipoles of strengths $c_1, c_2,..., c_n$ $(n \geqslant 2)$ are located at the points $x_1, x_2,..., x_n \in R^2$ (see Fig. 2), and the dipole at the point x_i is oriented in the direction h_i, $i = 1, 2,..., n$. Suppose further that $y_1, y_2,..., y_{m-1}, y_m$ $(m \geqslant 2)$ is another set of points in R^2. We will say that the sets $\{x_i\}$ and $\{y_i\}$ are separated if there exist such $x_0, y_0 \in R^2$ and such real $r > 0$ that

$$\|x_i - x_0\| < r \qquad \text{for all } i = 1, 2,..., n, \tag{6.1}$$

$$\|y_j - y_0\| < r \qquad \text{for all } j = 1, 2,..., n, \tag{6.2}$$

$$\|x_0 - y_0\| > 3r. \tag{6.3}$$

Finally, suppose that we would like to evaluate the sums

$$\sum_{i=1}^{n} c_i \phi_{x_i, h_i}(y_j) \tag{6.4}$$

for all $j = 1, 2,..., m$. Clearly, this is an order nm process (evaluating n fields at m points). However, if we are interested in evaluating (6.4) with finite accuracy (which is always the case in actual computations), Lemmas 5.1–5.4 can be used to speed up the process.

Let us denote the circle of radius r with the center at x_0 by D_x and the circle of radius r with the center at y_0 by D_y. Due to the triangle inequality, if $y \in D_y$ then for any $i = 1, 2,..., n$,

$$\|y - x_0\| \geqslant 2 \|x_i - x_0\| \tag{6.5}$$

and, applying Lemma 5.1, we conclude that

$$\sum_{i=1}^{n} c_i \phi_{x_i,h_i}(y) = \mathrm{Re}\left(\sum_{j=0}^{\infty} \left(\frac{1}{(y-x_0)^{j+1}} \sum_{i=1}^{n} c_i h_i (x_i - x_0)^j \right) \right). \tag{6.6}$$

Furthermore, for any $k \geqslant 1$,

$$\left\| \sum_{i=1}^{n} c_i \phi_{x_i,h_i}(y) - \mathrm{Re}\left(\sum_{j=0}^{k} \left(\frac{1}{(y-x_0)^{j+1}} \sum_{i=1}^{n} c_i h_i (x_i - x_0)^j \right) \right) \right\| \leqslant \frac{1}{2^{k+1} \| y - x_0 \|} \sum_{i=1}^{n} |c_i|. \tag{6.7}$$

The expression

$$\sum_{j=0}^{k} \frac{1}{(y-x_0)^{j+1}} \sum_{i=1}^{n} c_i h_i (x_i - x_0)^j \tag{6.8}$$

can be viewed as a polynomial of order $k+1$ with respect to $1/(y-x_0)$. For a fixed k, evaluating its coefficients requires a number of operations proportional to nk. If we look upon (6.8) as an approximation to (6.4), then in order to obtain a relative precision ε, k would have to be of the order

$$-\log_2 \varepsilon. \tag{6.9}$$

Now, by first evaluating the coefficients of (6.8) and then applying this expansion at the points $y_1, \ldots, y_m$, we have reduced the amount of computations to the order

$$akn + bkm \sim -\log(\varepsilon)(an + bm) \tag{6.10}$$

which for large m and n is significantly smaller than mn.

An alternative approach would be to evaluate the coefficients of the expansion (6.8) (order kn operations), using Lemma 5.2, convert them into an expansion of the form

$$\sum_{j=1}^{k} \alpha_j (y - y_0)^j \tag{6.11}$$

(order k^2 operations) and then evaluate the expansion (6.11) at the points $y_1, \ldots, y_m$ (order km operations).

When the field of charges (or dipoles) located at the nodes of a curve has to be evaluated at these same nodes, the above approach cannot be applied in such straightforward manner. In this case, the charges are not separated from the points where the field is to be evaluated, and Lemmas 5.1–5.4 have to be used in a more subtle way. This is done in the following section, resulting in an order n algorithm for the solution of (4.1).

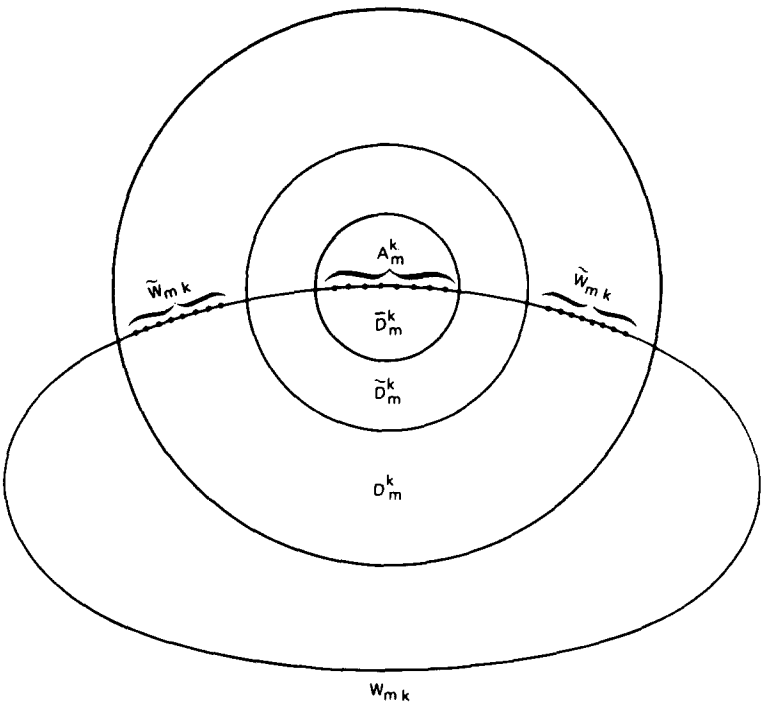

FIG. 3. Definition of A_m^k, D_m^k, $\bar{D}_m^k$, $\tilde{D}_m^k$, W_m^k and $\tilde{W}_m^k$.

VII. DETAILED DESCRIPTION OF AN ORDER N ALGORITHM

1. *Notation*

We will consider the situation depicted in Fig. 3. The curve Γ is discretized into $N=2^n$ equispaced nodes $x_1, x_2,..., x_N$, and we will denote the spacing between adjacent nodes (i.e., $\|x_i - x_{i-1}\|$) by h, while the internal normal to Γ at x_i will be denoted by N_i. Suppose that for each $i=1, 2,..., N$, a normally oriented dipole of strength σ_i is located at the node x_i. In this section, we will describe an order 2^n algorithm for rapid calculation of approximations g_i, $i=1, 2,..., 2^n$, to the sums

$$G_\sigma(x_i) = \sum_{\substack{j=1 \\ j\neq i}}^{N} \sigma_j \phi_{x_j, N_j}(x_i) \tag{7.1}$$

for $i=1, 2,..., N$.

We will denote by A the set of $N=2^n$ nodes x_i, $i=1, 2,..., N$, and by S the set of all dipoles located at these nodes. For $m=1, 2,..., n$ and $k=1, 2,..., 2^{n-m}$, we will denote by A_m^k the subset of A consisting of the nodes x_i with $i=2^m(k-1)+1$, $2^m(k-1)+2,..., 2^m k-1$, $2^m k$. The subset of S consisting of dipoles located at the

nodes in A_m^k will be denoted by S_m^k. For each A_m^k, we will denote by C_{mk} the point in R^2 defined by the formula

$$C_{mk} = \tfrac{1}{2}(x_{2^m(k-1)+2^{m-1}} + x_{2^m(k-1)+2^{m-1}+1}).$$

(Conceptually, C_{mk} is the "center" of A_m^k.) D_m^k will denote a circle with the center at C_{mk} and radius $h2^{m+1}$, $\tilde{D}_m^k$ will denote a circle with the center as C_{mk} and radius $h2^m$, and $\bar{D}_m^k$ will denote a circle with the center at C_{mk} and radius $h2^{m-1}$. Obviously, $A_m^k \subset \bar{D}_m^k \subset \tilde{D}_m^k \subset D_m^k$.

For $m = 1, 2, \ldots, n$, we will denote by W_m the set of all S_m^k with $k = 1, 2, \ldots, 2^{n-m}$. For $k = 1, 2, \ldots, 2^{n-m}$, we will denote by $\tilde{W}_{m,k}$ the subset of W_m consisting of all S_m^i such that $A_m^i \subset D_m^k$ and $A_m^i \not\subset \tilde{D}_m^k$. Finally, we will denote by W_{mk} the subset of W_m consisting of all S_m^i such that $A_m^i \not\subset D_m^k$.

For each of A_m^k and $p \geqslant 1$, we will define the mapping $\phi_{mp}^k : C^1 \setminus A_m^k \to C^1$ by the formula

$$\phi_{mp}^k(x) = \sum_{i=1}^{p} \frac{a_{mi}^k}{(x - C_{mk})^i} \tag{7.2}$$

with

$$a_{mi}^k = \sum_{x_j \in A_m^k} \sigma_j N_j (C_{mk} - x_j)^{i-1}. \tag{7.3}$$

For each A_m^k and $p \geqslant 1$, we will define the mapping $\psi_{mp}^k : C^1 \to C^1$ by the formula

$$\psi_{mp}^k(x) = \sum_{i=0}^{p} b_{mi}^k (x - C_{mk})^i \tag{7.4}$$

with

$$b_{mi}^k = \sum_{x_j \in \tilde{W}_{mk}} \frac{\sigma_j N_j}{(x_j - C_{mk})^{i+1}}. \tag{7.5}$$

Finally, we will denote by χ_{mp}^k the mapping $C^1 \to C^1$ defined by the formula

$$\chi_{mp}^k(x) = \sum_{i=0}^{p} c_{mi}^k (x - C_{mk})^i \tag{7.6}$$

with

$$c_{mi}^k = \sum_{x_j \in W_{mk}} \frac{\sigma_j N_j}{(x_j - C_{mk})^{i+1}}. \tag{7.7}$$

Remark 7.1. Due to Lemma 5.3, the function ϕ_{mp}^k can be viewed as an approximation to

$$\sum_{x_j \in A_m^k} \sigma_j \phi_{x_j, N_j}(x) \tag{7.8}$$

for x outside $\tilde{D}_m^k$. Due to Lemma 5.2, ψ_{mp}^k can be viewed as an approximation to

$$\sum_{x_j \in \tilde{W}_{mk}} \sigma_j \phi_{x_j, N_j}(x) \tag{7.9}$$

for x inside $\bar{D}_m^k$. Finally, χ_{mp}^k can be looked upon as an approximation to

$$\sum_{x_j \in W_{mk}} \sigma_j \phi_{x_j, N_j}(x) \tag{7.10}$$

for x inside $\tilde{D}_m^k$ due to Lemma 5.2.

Furthermore, the relative errors of all these approximations are proportional to $c2^{-p}$, where c is a constant determined by the points $x_1, x_2, \ldots, x_N$.

2. *Description of the Algorithm*

The algorithm consists of four stages. During the first stage the coefficients of expansions ϕ_{mp}^k are evaluated for all A_m^k and some sufficiently large p (in most applications, $p \in [10, 20]$) by means of Lemmas 5.1 and 5.3. During the second stage, expansions ϕ_{mp}^i with appropriately chosen i are used to evaluate the coefficients of expansions ψ_{mp}^k by means of Lemma 5.2. During the third stage, expansions ψ_{mp}^i are combined to obtain the coefficients of expansions χ_{mp}^i by means of Lemma 5.4. Finally, during the fourth stage the expansions χ_{1p}^i are evaluated at x_j, $j = 1, 2, \ldots, N$, giving the desired approximation to (7.1). In the formal description of the algorithm below, the left column contains the operations to be performed. In the right column, CPU time estimates for operations (Stages, Steps, and Do Loops) are given. The coefficients $a_1, a_2, \ldots, a_{11}$ in these estimates are determined by the computer system, implementation, etc.

Stage 1	$(a_1 p + a_3 p^2) 2^n$
Step 1	$a_1 p 2^n$
for $m = 2$ Step 1 until 2^{m-1} do (Using Lemma 5.3, create the coefficients of expansions ϕ_{1p}^k from the fields of dipoles at the nodes x_{2k-1}, x_{2k})	$a_1 p 2^n$
End of Step 1	
Step 2	$a_3 2^n$
for $m = 2$ Step 1 until $n-1$ do	$\sum_{m=2}^{n-1} a_2 p^2 2^{n-m} = a_3 2^n$
BEGIN	
for $k = 1$ Step 1 until 2^{n-m} do (Using Lemma 5.1, combine expansions ϕ_{mp}^{2k-1} and ϕ_{mp}^{2k} obtaining $\phi_{m+1,p}^k$)	$a_2 p^2 2^{n-m}$
END	
End of Step 2	

Stage 2	$A_5 p^2 2^n + a_7 p^2 2^n = a_8 p^2 2^n$
Step 1	$A_5 p^2 2^n$
for $m = 2$ Step 1 until $n-1$ do	$\sum_{m=2}^{n-1} a_4 p^2 2^{n-m} = a_5 p^2 2^n$
BEGIN	
for $k = 1$ Step 1 until 2^{n-m} do (Using Lemma 5.2, combine expansions ϕ^j_{mp} with appropriately chosen j to obtain ψ^k_{mp})	$2^{n-m} a_4 p^2$
END	
End of Step 1	
Step 2	$a_7 p^2 2^n$
for $m = n-1$ Step (-1) until 2 do	$\sum_{m=2}^{n-1} a_6 p^2 2^{n-m} = a_7 p^2 2^n$
BEGIN	
for $k = 1$ Step 1 until 2^{n-m} do (Using Lemma 5.4, add expansion χ^k_{mp} to expansions $\psi^{2k-1}_{m-1,p}$ and $\psi^{2k}_{m-1,p}$ obtaining expansions $\chi^{2k-1}_{m-1,p}$ and $\chi^{2k}_{m-1,p}$)	$a_6 p^2 2^{n-m}$
END	
End of Step 2	
Stage 3	$a_{10} 2^n$
Step 1	$a_{10} 2^n$
for $k = 1$ Step 1 until 2^{n-1} do (evaluate the expansion χ^k_{1p} at the nodes x_{2k-1}, x_{2k} obtaining the field χ^h_{1p} at these nodes).	$a_9 p 2^{n-1} = a_{10} p 2^n$
End of Step 1	
Stage 4	$a_{11} 2^n p$
Step 1	$a_{11} 2^n p$
for $k = 1$ Step 1 until 2^{n-1} do	$a_{11} 2^n p$
BEGIN	
for $i = 2k-1$, $2k$ do	$a_{11} 2^n p$
(Evaluate an approximation g_i to $G_\sigma(x_i)$ according to the formula	

$$g_i = \chi^k_{n-1,p}(x_i) + \sum_{\substack{x_j \in A^k_{1p} \\ j \neq i}} \sigma_j \phi_{x_j, N_j}(x_i)) \tag{7.11}$$

END	
End of Step 1	

Remark 7.2. Adding up the CPU time estimates for stages 1 through 4, we

obtain the following estimate for approximately applying the matrix A_n of the system (3.7) to a vector:

$$T_A = (P_0 + P_1 p + P_2 p^2) N. \tag{7.12}$$

Since the relative accuracy of the expansions ϕ_{mp}^k, ψ_{mp}^k, χ_{mp}^k is proportional to $1/2^p$ (see Remark 7.1), (7.12) can be rewritten as

$$T_A = (Q_0 + Q_1 \log(\varepsilon) + Q_2(\log(\varepsilon))^2) N, \tag{7.13}$$

where ε is the relative accuracy of the evaluation of the operator A_N.

3. *Rapid Solution of the System* (3.7)

In the preceding subsection, we described in some detail an algorithm for rapid application of the matrix A_n of the system (3.7) to arbitrary vectors. A standard algorithm for the solution of the original equation (2.1) utilizes a GCRA to solve the system (3.7) and the GCRA involves applying the matrix A_n to a sequence of recursively generated vectors. If, instead of applying A_n to these vectors directly, we utilize the algorithm of the preceeding subsection, then by combining the estimates (3.10), (3.11), (7.13), we obtain the following CPU time estimate for solving (3.7):

$$\begin{aligned} T_{\text{solve}} \leqslant\ & (a \log(\varepsilon_2) + b(\log(\varepsilon_2))^2) N \\ & + (Q_0 + Q_1 \log(\varepsilon_1) + Q_2(\log(\varepsilon_1))^2) \log(\varepsilon_2) N, \end{aligned} \tag{7.14}$$

where ε_1 is the relative accuracy of the approximation $\tilde{A}_N$ of the matrix A_N, ε_2 is the relative accuracy in the solution of this approximate linear system, and the coefficients a, b, Q_0, Q_1, Q_2 are determined by the computer system, language, etc. When $\varepsilon_1 = \varepsilon_2 = \varepsilon$ (which is often a reasonable convention), (7.14) becomes

$$T_{\text{solve}} = (R_1 + R_2 \log(\varepsilon) + R_3(\log(\varepsilon))^2) N \log(\varepsilon) \tag{7.15}$$

with R_1, R_2, R_3 determined by the language, computer system, etc.

VIII. Implementation and Numerical Results

The algorithm of the present paper has been implemented for both Dirichlet and Neumann problems in two dimensions, and in this section we present three numerical examples illustrating its performance.

1. *Dirichlet Problem Inside a Square*

The problem (2.1) was solved with Ω a square with the corners at the points $(-10, 10)$, $(-10, -10)$, $(10, -10)$, $(10, 10)$ and the right hand side f equal to the field of a charge B of unity intensity located at the point $(13, 5)$ (see Fig. 4). In this case, the solution of (2.10) is equal to the field of the charge B inside Ω (see [11]),

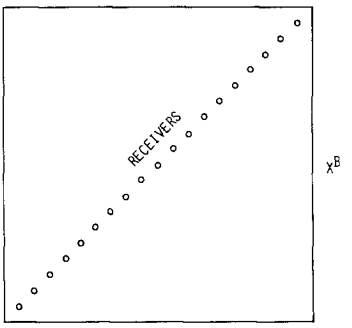

FIG. 4. Dirichlet problem inside a square.

which provides a convenient way to verify the accuracy of the solution. The solution obtained by the algorithm of this paper was calculated at 19 points inside Ω with the coordinates $(-9, -9)$, $(-8, -8)$,..., $(-1, -1)$, $(0, 0)$, $(1, 1)$,..., $(8, 8)$, $(9, 9)$, and in Table I, the results for various discretizations of Γ are compared to the analytical solution at these points. Table I illustrates the following three observations characteristic of algorithms based on second kind integral equations.

(a) The number of iterations required by GCRA to converge to a given accuracy is almost independent of the number of nodes in the discretization of Γ.

b. The computed convergence rate of the algorithm is asymptotically quadratic as expected.

c. The effect of truncation errors on the accuracy of the solution is virtually independent of the number of nodes in the discretization of Γ since the system of linear equations being solved is well-conditioned.

In Table II, the CPU times required by the algorithm of the present paper are compared to the CPU times required by a fast Poisson solver written at the National Center for Atmospheric Research (See [5, 13a]) to solve a similar

TABLE I

Number of nodes on the boundary of the square	Number of Iterations of GCRA	Resulting accuracy (mean square error at 19 receivers)	CPU time on IBM-3033
64	11	0.832*D*-2	0.34
128	12	0.903*D*-4	0.65
256	14	0.897*D*-5	1.16
512	14	0.209*D*-5	1.98
1024	14	0.373*D*-6	3.51
2048	14	0.568*D*-7	6.52

TABLE II

Number of nodes on the boundary of the square	CPU time on IBM-3033	
	NCAR Fast Poisson Solver	Algorithm of the present paper
128	0.07	0.65
265	0.36	1.16
512	1.71	1.98
1024	6.84 (estimate)	3.51
2048	27.36 (estimate)	6.52

problem. Both algorithms were run on an IBM-3033 with the algorithm of this paper converging to 10^{-7} and the solution evaluated at 19 points inside the square. It is clear from Table II that for relatively small scale interior problems in rectangular regions, the NCAR algorithm is obviously superior. For large scale problems, the algorithm of the present paper is faster, and its performance is practically independent of the shape of the region.

Remark 8.1. It should be noted that in the above example, the algorithm of the present paper was used to evaluate the solution at 19 points inside the square, while the NCAR program produced the solution at $n^2/16$ points. Therefore, the comparison between these algorithms that can be inferred from Table II is only valid if the solution is required at a small number of points.

The algorithm of the present paper has been applied to several other problems, both exterior and interior. Figure 5 depicts the lines of a horizontal air current scattering from a car-shaped obstacle (exterior Neumann problem). In this case, the number of nodes in the discretization of the boundary was 1024, the problem was solved to 7 digits, and it took 5.11 seconds on an IBM 3033. Figure 6 depicts the equipotential lines of a field generated by 5 point electrodes inside a grounded conducting cavity of a complicated shape (interior Dirichlet problem). In this case, the number of nodes on the boundary was 2048, and the algorithm took 8.32 seconds of IBM 3033 CPU time to converge to 6 digits.

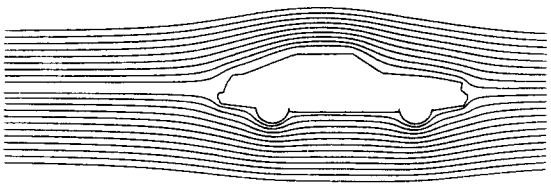

FIG. 5. Lines of current for an exterior Neumann problem.

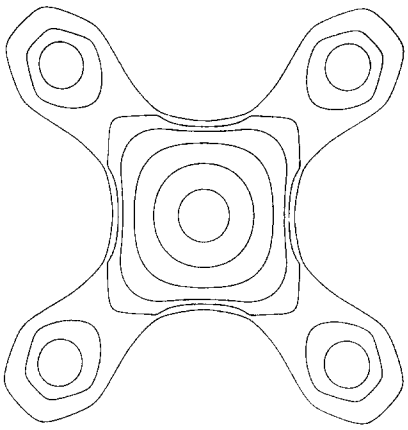

FIG. 6. Equipotential lines for an interior dirichlet problem.

IX. GENERALIZATIONS AND CONCLUSIONS

The algorithm of the present paper appears to be the most efficient of presently available tools for the solution of large scale boundary value problems whenever the solution needs to be evaluated at a limited number of points. For small to moderate scale interior problems, and whenever the solution is required at a large number of points, classical fast solvers are likely to be more efficient.

In Section VIII, we assumed that the number of the nodes in the discretization of the boundary of the region is a power of 2. Clearly, this requirement is not an essential one, and it can be eliminated by obvious changes in the logic.

The algorithm of this paper has an obvious 3-dimensional analogue. In the 3-dimensional version, the sines and cosines in expressions (2.14)–(2.15) are replaced by spherical harmonics (see [11]), one-dimensional quadrature formulae are replaced by two-dimensional ones, and the programming becomes more involved. Otherwise, the transition is fairly straightforward.

In many applications, the Nyström method is not the algorithm of choice as a tool for discretizing the integral equation. In aerodynamical calculations, for example, the powerful and versatile method of panels (see [8, 18]) has been extremely successful. Like the Nyström algorithm, it leads to repeated evaluation of a field of a distribution of charges and/or dipoles on the boundary of the region. The algorithm of Section VII can be applied in the context of the method of panels just as easily as in the context of the Nyström method. This would result in a method of panels with the operation count proportional to n (where n is the number of nodes in the discretization of the boundary of the region) as opposed to the usual estimate of n^2.

ACKNOWLEDGMENTS

It is the author's pleasure to thank J. E. Dennis and J. B. Bell for their interest and support and E. Zhits for his help in the implementation of the algorithm. The author would also like to thank a referee for attracting his attention to the Refs. [11a, 13a, 16a], and for several useful suggestions.

REFERENCES

1. K. E. ATKINSON, "A Survey of Numerical Methods for the Solution of Fredholm Integral Equations of the Second Kind," SIAM, Philadelphia, Pa, 1976.
2. J. L. BLUE, Boundary integral solutions of Laplace's equation, Computing Science Technical Report #60, Bell Laboratories, Murray Hills, N.J., 1977.
3. R. V. CHURCHILL, "Complex Variables and Applications," McGraw–Hill, New York, 1960.
4. R. COURANT AND D. HILBERT, "Methods of Mathematical Physics," Interscience, New York, 1966.
5. I.S. DUFF, Recent developments in the solution of large sparse linear equations, *in* "INRIA Fourth International Symposium on Computing Methods in Applied Science and Engineering, Versailles, 1979."
6. S. C. EISENSTAT, H. C. ELMAN, AND M. H. SCHULTZ, Variational iterative methods for nonsymmetric systems of linear equations, *SIAM J. Numer. Anal.* (1983), 345–357.
7. I. GLADWELL AND R. WAIT, Eds., "A Survey of Numerical Methods for Partial Differential Equations, Oxford Univ. Press (Clarendon), Oxford, 1979.
8. J. K. HESS, A higher order panel method for three-dimensional potential flow, Douglas Aircraft Co., Report No. MDC J8519, 1979.
9. J. L. HESS AND A. M. O. SMITH, Calculation of potential flow about arbitrary bodies, *Prog. Aero. Sci.* **8** (1967), 1–138.
10. L. V. KANTOROVICH AND V. I. KRYLOV, "Approximate Methods of Higher Analysis," Wiley, New York, 1964.
11. N. S. KOSHLIAKOV, M. M. SMIRNOV, AND E. B. GLINER, "Differential Equations of Mathematical Physics," North-Holland, Amsterdam, 1964.

11a. A. MAYO, The fast solution of Poisson's and biharmonic equations in irregular regions, *SIAM J. Numer. Anal.* **21**, No. 2 (1984).

12. I. G. PETROVSKY, "Lectures on Partial Differential Equations," Interscience, London, 1954.
13. V. ROKHLIN, Solution of acoustic scattering problems by means of second kind integral equations, *Wave Motion* **5** (1983), 257–272.

13a. P. N. SCHWARTZTRAUBER AND R. A. SWEET, Algorithm 541, *ACM Trans. Math. Software* **5** (1979), 352.

14. R. S. STEPLEMAN, Some refinements of method of moments for the solution of the charged lamina problems in 3-D, *in* "Advances in Computer Methods for Partial Differential Equations—II" (R. Vichnevetsky, Ed.), pp. 34–40, IMACS, New Brunswick, NJ, 1977.
15. R. S. STEPLEMAN, personal communication, 1981.
16. J. STOER AND R. BULIRSCH, "Introduction to Numerical Analysis," Springer-Verlag, New York, 1980.

16a. R. WINTHNER, Some superlinear convergence results for the conjugate gradient method, *SIAM J. Numer. Anal.* **17**, No. 1 (1981).

17. A. WOUK, "A Course of Applied Functional Analysis," Wiley, New York, 1979.
18. D. P. YOUNG *et al.*, *SIAM J. Statist. Sci. Comput.*, in press.