JOURNAL OF COMPUTATIONAL PHYSICS 57, 403–414 (1985)

Direct Solution of Poisson's Equation in Cylindrically Symmetric Geometry: A Fast Algorithm

E. E. KUNHARDT AND P. F. WILLIAMS

Departments of Electrical Engineering and Physics, Texas Tech University, Lubbock, Texas 79409

Received June 29, 1983; revised December 6, 1983

We describe a new algorithm for directly solving Poisson's equation in cylindrically symmetric geometries. It is based on the use of fast Fourier transforms for the axial solution, and a novel expansion in cubic splines for the radial solution. Error and stability analyses of the algorithm are included, and the results of test calculations are presented. The algorithm has proved stable, reasonably fast, and able to handle very stiff driving functions.

I. INTRODUCTION

A number of direct methods for solving the discrete Poisson equation in rectangular coordinates have been developed [1–4]. For further studies on such methods see Wilhelmson and Ericksen [3] and James [4].

In this paper, we present a novel combination of a numerical transform technique (FFT) and an expansion in cubic splines [5] for the solution of Poisson's equation in cylindrically symmetric geometry [6]. This combination may also be applicable to other separable coordinate systems. The transform is used in the z-direction (the axis of symmetry), thus reducing the problem to an ordinary differential equation in the radial coordinate. To solve this equation, we have approximated the solution between each set of grid points by a cubic spline, chosen to satisfy the equation and to provide continuity through the second derivative at the endpoints. This technique belongs to the general class of approximation schemes termed finite element procedures. The technique allows for non-uniform grid spacing in r, a property we have found to be important in a number of applications. The use of the sine transform in the z-direction, however, restricts the algorithm to applications involving uniform axial grid spacing.

Restrictions of this approach arise from requirements on the boundary conditions. It can be applied to problems in which Dirichlet or Neumann (or mixed) boundary conditions are specified. Although the technique could probably be

0021-9991/85 $3.00

applied to more general cases, the boundary surfaces should lie on coordinate surfaces.

This method was developed to satisfy the need for a fast solution to Poisson's equation under conditions where the source may be a rapidly varying function of space, such as arise, for example, in the study of the evolution of an electron avalanche taking into account the electric field generated by the avalanche space charge. The solution to Poisson's equation in this application has presented great difficulty. The problem is stiff and requires speed in the computation of the field in order to carry out the simulation of the evolution of an avalanche for a significant length of time. The algorithm described in this paper has been successfully applied to this problem [7, 8].

In Section II, the algorithm is discussed. Analysis of the stability and truncation error is given in Section III, and a discussion of the performance of a Fortran program that implements the algorithm and example calculations are given in the last two sections.

II. Description of the Algorithm

We wish to solve the two-dimensional Poisson equation

$$\nabla^2\phi = \rho \tag{1}$$

for the function $\phi(r, z)$ in terms of the driving function $\rho(r, z)$ in the cylindrical domain: $0 \leqslant z \leqslant a$, $0 \leqslant r \leqslant R$ with $\phi(r, 0) = \phi(r, a) = 0$. ϕ may be expanded as

$$\phi(r, z) = \sum_{n=1}^{N_z} \phi_n(r) \sin \frac{n\pi}{a} z \tag{2}$$

where N_z is the total number of grid points in the z-direction. Putting Eq. (2) into a z-discretized form of (1) and using the orthogonality properties of the basis functions, we obtain

$$\frac{\partial^2\phi_n}{\partial r^2} + \frac{1}{r}\frac{\partial\phi_n}{\partial r} + \frac{2}{h_z^2}\left(\cos\frac{n\pi}{N_z+1} - 1\right)\phi_n = \rho_n(r) \tag{3}$$

where

$$\rho(r, z) = \sum_{n=1}^{N_z} \rho_n(r) \sin \frac{n\pi}{a} z \tag{4}$$

and h_z is the grid spacing in the z-direction. Fast Fourier transform algorithms may be used to obtain $\rho_n(r)$ from $\rho(r, z)$, and to obtain $\phi(r, z)$ from $\phi_n(r)$. Thus the problem reduces to numerically solving Eq. (3) in an efficient and stable manner.

For this purpose we use a technique utilizing a set of cubic polynomial spline functions and find it to be convenient and accurate in solving this equation even for cases where the variation of $\rho_n(r)$ is relatively stiff [7].

The r-axis is partitioned into a set of N_r points $\{r_i\}$, with separation h_i between r_i and r_{i+1}. A set of N_r cubic spline functions

$$S_i(r) = \sum_{j=0}^{3} \frac{1}{j!} T_{i,j}(r - r_i)^j \tag{5}$$

is introduced, and the transformed potential $\phi_n(r)$ is approximated by $S_i(r)$ for $r_{i-1} < r < r_i$. To determine the coefficients $\{T_{i,j}\}$, ϕ_n, ϕ_n', and ϕ_n'' are required to be continuous at each grid point. We also require S_i to satisfy the differential equation, Eq. (3), at each grid point. In this context, the endpoints may be regarded as collocation points. These conditions imply, respectively,

$$T_{i,0} = T_{i-1,0} + h_{i-1} T_{i-1,1} + \tfrac{1}{2} h_{i-1}^2 T_{i-1,2} + \tfrac{1}{6} h_{i-1}^3 T_{i-1,3} \tag{6a}$$

$$T_{i,1} = T_{i-1,1} + h_{i-1} T_{i-1,2} + \tfrac{1}{2} h_{i-1}^2 T_{i-1,3} \tag{6b}$$

$$T_{i,2} = T_{i-1,2} + h_{i-1} T_{i-1,3} \tag{6c}$$

$$T_{i,2} + f_i T_{i,1} + g_i T_{i,0} = \rho_n(r_i) \tag{6d}$$

where

$$f_i = \frac{1}{r_i} \qquad \text{and} \qquad g_i = \frac{2}{h_z^2}\left(\cos\frac{n\pi}{N_z + 1} - 1\right).$$

These equations, along with two boundary conditions, are sufficient to specify the coefficients, $T_{i,j}$, and an approximation to the function and its first three derivatives is thereby determined.

Through introduction of the splines, the solution of Eq. (3) has been reduced to the numerical solution of $4N_r$ simultaneous, linear equations in $4N_r$ unknowns. After tedious algebraic manipulation, the coefficients of all but one of the powers of $(r - r_i)$, usually either $\{T_{i,0}\}$ or $\{T_{i,1}\}$, can be eliminated. The result is a set of equations in tridiagonal form (see Appendix). Assuming we keep the equations for the set $\{T_{i,0}\}$, we obtain

$$\sum_j Q_{i,j} T_{j,0} = \sigma_i \tag{7}$$

where the σ_i are related to the original driving function, $\rho_n(r_i)$, and the matrix $\underline{\underline{Q}}$ is in tridiagonal form. Specific formulae for $\underline{\underline{Q}}$ and $\underline{\sigma}$ are derived in the Appendix for the boundary condition $\phi(R)$ given.

Equation (7) may then be solved to determine the spline coefficients, $\{T_{i,0}\}$. The remaining coefficients may be determined from

$$T_{i,1} = W_i + E_i T_{i-1,0} + F_i T_{i,0} \tag{8a}$$

$$T_{i,2} = \rho_i - g_i T_{i,0} - f_i T_{i,1} \tag{8b}$$

$$T_{i,3} = \frac{1}{h_i}(T_{i+1,2} - T_{i,2}) \tag{8c}$$

(see Appendix for values of constants).

In order to avoid problems with an exponentially growing round-off error, we used the following technique to solve Eq. (7) [9]. $\underline{\underline{Q}}$ is first factored into two matrices, $\underline{\underline{Q}} = \underline{\underline{U}} \cdot \underline{\underline{L}}$, where

$$U_{i,k} = U_{i,i+1}\delta_{i,k-1} + U_{i,i}\delta_{i,k} \tag{9}$$

and

$$L_{i,k} = L_{i,i-1}\delta_{i,k+1} + L_{i,i}\delta_{i,k}.$$

There is some freedom in choosing this factorization, and we choose $U_{i,i} = 1$ for convenience. The tridiagonal equations then are

$$\underline{\underline{U}} \cdot \underline{\underline{L}} \cdot \underline{T}_0 = \underline{\sigma} \tag{10}$$

where

$$U_{i,j} = \delta_{i,j} + u_i \delta_{i,j-1}$$

$$L_{i,j} = d_i \delta_{i,j} + l_i \delta_{i,j+1}$$

and

$$l_i = Q_{i,i-1}$$

$$d_i = Q_{ii} - u_i Q_{i+1,i}$$

$$u_i = \frac{Q_{i,i+1}}{d_{i+1}}.$$

Defining $\underline{\underline{L}} \cdot \underline{T}_0 = \underline{G}$, Eq. (9) may be written as two equations:

$$\underline{\underline{U}} \cdot \underline{G} = \underline{\sigma} \tag{11a}$$

and

$$\underline{\underline{L}} \cdot \underline{T}_0 = \underline{G}. \tag{11b}$$

Equation (11a) is readily solved for $\underline{G}$, and then (11b) is solved for $\underline{T}_0$. In these solutions, Eq. (11a) must be iterated inwards, and (11b) iterated outwards in order to avoid numerical round-off error problems.

III. Round-Off Error Stability and Truncation Error Analysis

In order to analyze the stability of this algorithm to round-off errors, we will assume a uniformly spaced partition so that $h_i = h$, $f_i = 1/ih$, and

$$g_i = \frac{2}{h^2}\left(\cos\frac{n\pi}{N_z+1} - 1\right) = g.$$

In the limit of large i, our driving function will always approach zero, and Eq. (7) reduces in this limit to

$$\left(1+\frac{h^2}{6}g\right)\Delta^2 T_{i,0} + \frac{1}{h(i-1/3)}\Delta T_{i,0} + gT_{i,0} = 0. \tag{12}$$

Neglecting the term in ΔT, the general solution to this equation is

$$T_{i,0} = A\xi^i + B\xi^{-i} \tag{13}$$

where

$$\xi = \frac{1}{1+\frac{h^2}{6}g}\left[1-\frac{h^2}{3}g + \sqrt{-h^2 g\left(1-\frac{h^2}{12}g\right)}\right]. \tag{14}$$

This solution consists of exponentially growing and decaying terms. In an analytical solution, the growing term would be discarded due to the boundary condition at $r \to \infty$. For a numerical solution, however, small round-off errors can introduce a small amount of this growing solution, resulting in large errors for large i. This problem is eliminated by factoring the tridiagonal, 2nd-order finite difference equation, Eq. (7), into two 1st-order finite difference equations, each having only one homogeneous solution, providing that we can always choose our direction of integration such that the homogeneous solution is decaying.

Choosing the factorization given in Eqs. (11), for $0 > h^2 g > -12$, the elements of $\underline{\underline{U}}$ and $\underline{\underline{L}}$, l_i, d_i, and u_i, approach for large i

$$l_i \to -\left(1+\frac{h^2}{6}g\right)$$

$$d_i \to 1 + \frac{1}{6}\sqrt{-h^2 g(h^2 g+12)} = d$$

$$u_i \to \frac{-\left(1+\frac{h^2}{6}g\right)}{d}.$$

The vector $\underline{G}$ satisfies

$$G_i + u_i G_{i+1} = \sigma_i,$$

which in the limit of large i becomes

$$G_i - \frac{1 + \frac{h^2 g}{6}}{d} G_{i+1} = 0$$

with solution

$$G_i = A \left(\frac{d}{1 + \frac{h^2 g}{6}} \right)^i.$$

For $-12 \leqslant h^2 g \leqslant 0$, this solution grows exponentially with i, and therefore Eq. (11a) must be solved by starting at $i = N_r - 1$ and iterating inwards.

Similarly, the equation for $\underline{T}_0$ is

$$d_i T_{i,0} + l_i T_{i-1,0} = G_i$$

with the asymptotic homogeneous equation

$$d_i T_{i,0} - \left(1 + \frac{h^2}{6}\right) T_{i-1,0} = 0$$

and solution

$$T_{i,0} = \left(\frac{1 + \frac{h^2}{6} g}{d} \right)^i.$$

This is an exponentially decaying solution, so the equation must be solved by starting at $i = 0$ and iterating outwards.

By factoring the tridiagonal matrix and then solving for $\underline{T}_0$ in this way, we were able to essentially eliminate inaccuracies due to round-off error, however, the preceding discussion shows that the truncation error with this algorithm may be significant. In the limit of large i, the tridiagonal Eq. (7) reduces to Eq. (12), with solutions given by Eqs. (13) and (14). For $|h^2 g| \ll 1$, these solutions reduce to the expected exponential solution of the original different equation. For larger values of $|h^2 g|$, the discrete solution deviates significantly from the continuous solution, and for $h^2 g < -6$, $\xi < 0$ so that the sign of the discrete solution alternates at consecutive points.

The quantity h^2g is given by

$$h^2g = 2\left(\frac{h_r}{h_z}\right)^2\left(\cos\frac{n\pi}{N_z+1}-1\right)$$

where h_r and h_z are the r-axis and z-axis partition widths, respectively. If we take $h_r \leqslant h_z$, $h^2g > -4$ and the discrete solution is guaranteed to be well behaved.

IV. Fortran Implementation

The numerical procedure outlined above has been written into a Fortran program and run on a VAX11/780 computer without floating point accelerator. The program completed the calculation of the potential and both components of the field on a spatial grid, containing $(N_r \times N_z) = 51 \times 32$ mesh points, in approximately 10 seconds. Although care was taken to write efficient Fortran code, no serious effort was made to further optimize the speed of execution, and significant improvements are probably possible here. In the program, for each value of r, $\rho(r, z)$ in Eq. (1) is Fourier transformed in the z-dimension yielding the functions $\rho_n(r)$ in Eq. (3). For each value of n, this equation is solved for the $\phi_n(r)$, using the cubic spline differential equation solver. For each n, Eq. (8a) is used to determine the coefficients $\{T_{i,1}\}$ which are just the values of

$$\frac{\partial \phi_n}{\partial r}$$

at each point, r_i. The z-component of the field is determined by fast inverse cosine-transforming the set

$$\frac{2}{h_z}\sin\left(\frac{n\pi h}{2a}\right)\phi_n$$

and the r-component is obtained by the fast inverse sine-transforming the set

$$\frac{\partial \phi_n}{\partial r}.$$

The algorithm was tested at two levels. First, the solution to Eq. (3) was obtained using the spline expansion and compared to analytical solutions obtained for given source functions ρ_n. To test for round-off errors, source functions leading to cubic polynomial solutions were tested. The algorithm produced results which agreed with the analytic solution to within machine round-off error. Fast variations in ρ_n with r were also tested giving very satisfactory results. The second test was on the complete algorithm, i.e., the numerical solution to Eq. (1). For this test, a test

charge density consisting of a set of spheres, each of uniform charge density, placed between two conducting planes was used. The position, radius, and charge density of the spheres were independently varied. The field from a charge distribution of this type can also be calculated readily by the method of images, although many images are needed to ensure proper convergence. A separate program was written to calculate the field using this method, and the results were compared with our algorithm. The results of these tests are given in the last section.

The algorithm described in this paper is easily implemented and is suitable for use as a subroutine in simulation problems requiring the repeated solution of Poisson's equation. We currently use a subroutine based on the algorithm in several separate applications related to numerical studies of space-charge-dominated transport in gases, and have found it to perform reliably and accurately. Although the error analysis we have described and the test results we present were carried out assuming a uniform radial grid spacing, the algorithm is readily applied to the problems with non-uniform grid spacing in the radial direction. In our applications we typically take the radial grid spacing to be uniform for the innermost radial points, and then to be exponentially increasing for the rest. A complete code may be obtained from the authors.

V. Test Results

The results we obtained in these tests depended on the charge density used, but they were generally good. The most completely studied charge distribution was that of a single uniform sphere of charge. In most tests, h_r/h_z was approximately one so that significant truncation errors may be expected for high-frequency Fourier components. Nevertheless, as long as the diameter of the sphere was at least several times the partition spacing, the spline algorithm produced a potential field which agreed with the image charge routine to within 2%. The error in the calculated electric field component was of similar magnitude, although at a few isolated points, associated with the sharp edge of the charge distribution, the error was as much as 10%. Figure 1 shows the input charge distribution and the calculated fields and associated errors. From Fig. 1a it is evident that the sampling error associated with this grid spacing is significant, and we believe most of the errors in the calculated fields are due to this factor.

In a more stringent test, we checked the accuracy of the program using an input charge density consisting of two equally but oppositely charged spheres of equal radii, but displaced by $(5/2)\,h_z$, where h_z is the partition dimension in the z-direction. These results are shown in Fig. 2. From Fig. 2a it is evident that the charge density sampling error is substantial in this case. We found the error in the calculated fields to be generally of the order of a few percent, but at several isolated points the error was of the order of 25%. These points are located near the sharp discontinuities in ρ, consistent with the sampling error in ρ. Other tests were run with both greater and smaller displacements of the two spheres of charge and we

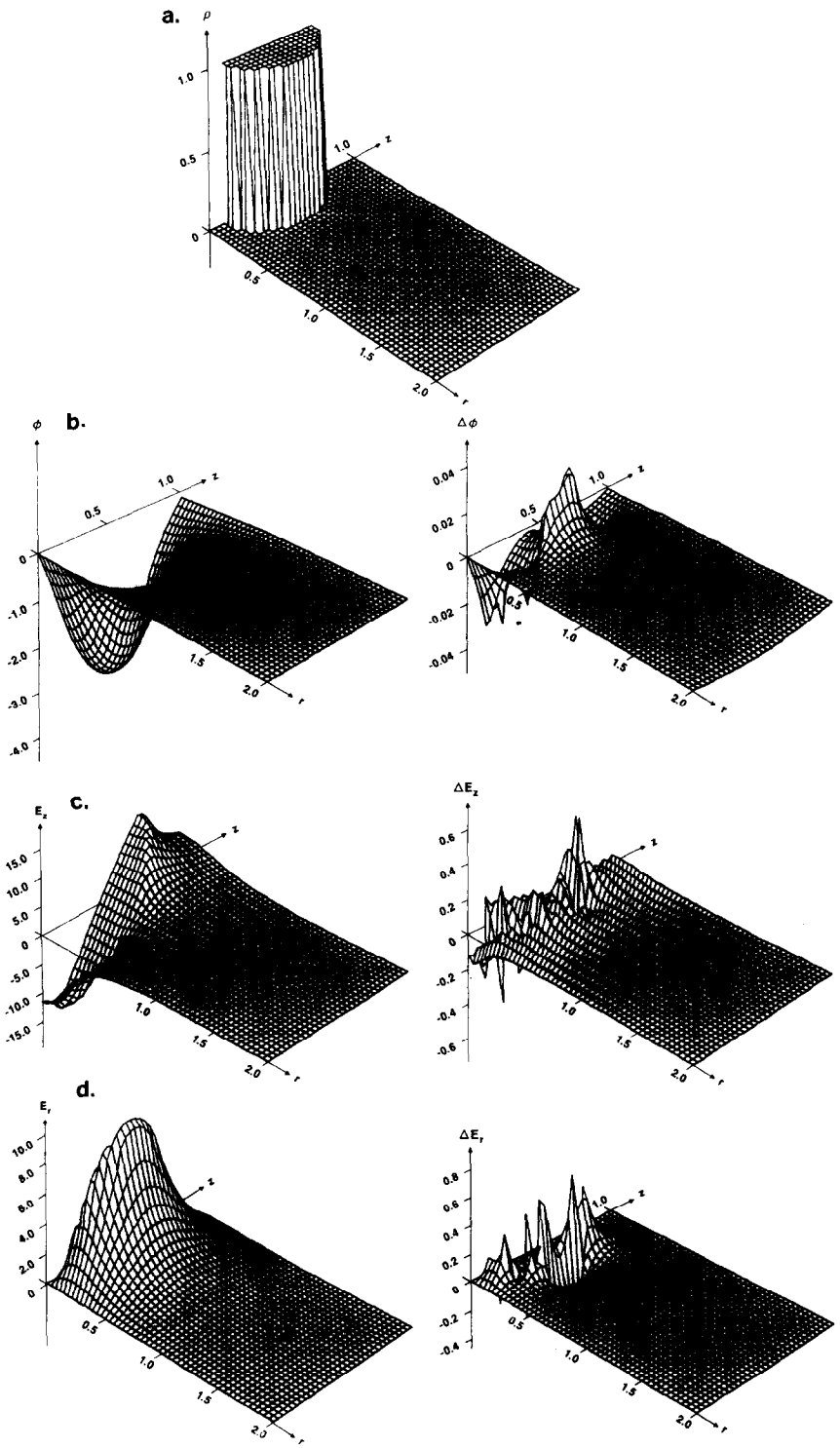

FIG. 1. Results of the algorithm for a single, uniform sphere of charge density. (a) Input charge density; (b) calculated potential and error; (c) calculated z-component of the electric field and error; (d) calculated r-component of the electric field and error.

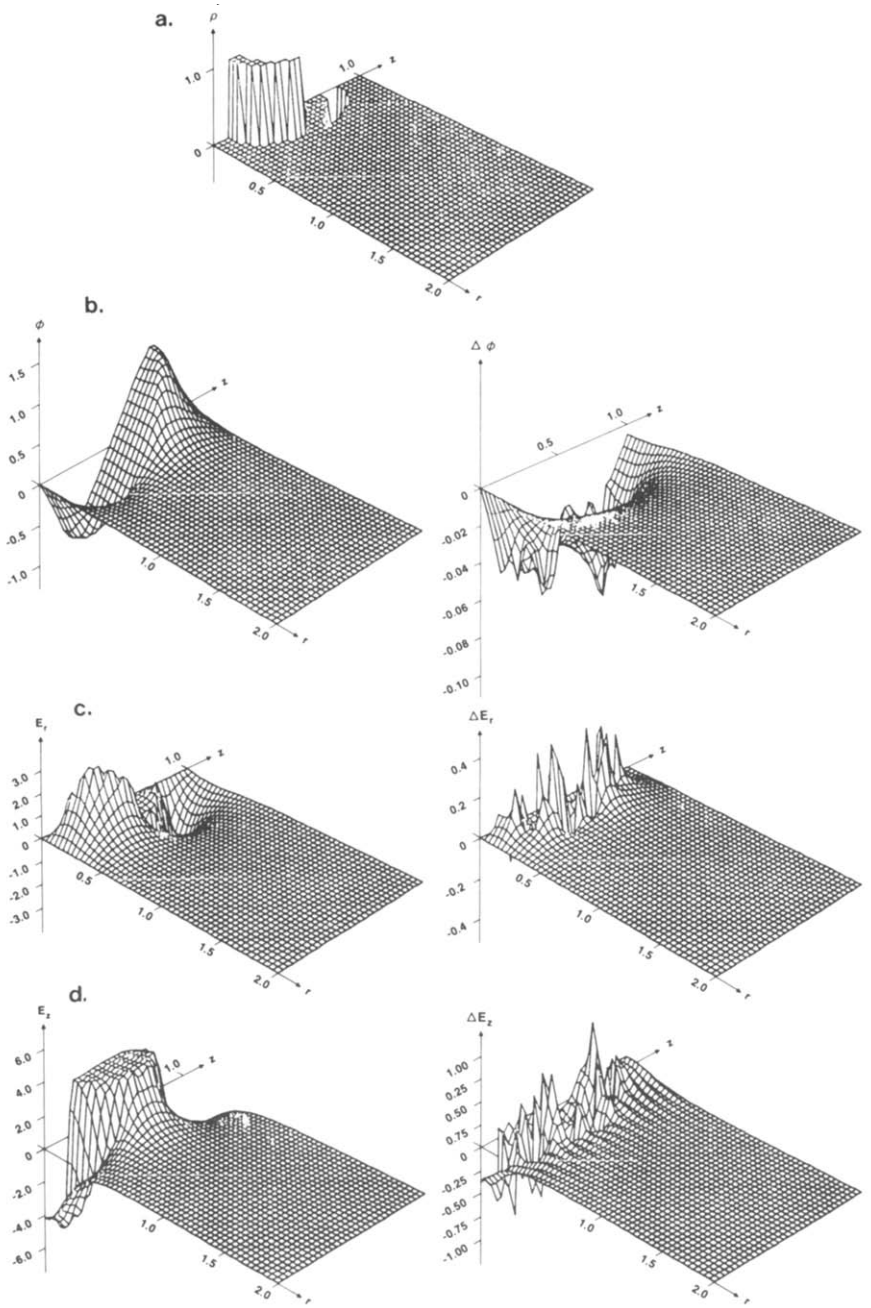

FIG. 2. Results of the algorithm for two uniform spheres of charge, equally but oppositely charged and of equal radii, and axially displaced from each other. (a) Input charge density; (b) calculated potential and error; (c) calculated r-component of the electric field and error; (d) calculated z-component of the electric field and error.

found the maximum error to vary with separation in a way consistent with this interpretation.

For these two-sphere test charge densities, the Fourier transformed density has strong high-frequency components. According to the discussion in Section III, significant truncation error may be expected for these cases. It is difficult to separate truncation from sampling errors, but asymptotically we expect truncation errors to produce a radially oscillating solution, which we do not observe in these tests. For this reason we believe that the error analysis in Section III is overly pessimistic in the non-asymptotic region, at least for the class of driving functions we have tested.

APPENDIX: EXPLICIT FORMULAE FOR $\underline{\underline{U}}$ AND $\underline{\sigma}$

The explicit representation of Eq. (7) follows. For $2 < i < N_r - 1$, and defining $f_i = 1/r_i$, and

$$g_i = \frac{2}{h_z^2}\left(\cos\frac{n\pi}{N_z+1} - 1\right),$$

the tridiagonal matrix $\underline{\underline{Q}}$ is given by

$$\begin{aligned} Q_{i,j} = {} & A_{i,1}E_i\delta_{i,j+1} + (A_{i,0} + A_{i,1}F_i - B_{i,1}E_{i+1})\,\delta_{i,j} \\ & - (B_{i,0} + B_{i,1}F_{i+1})\,\delta_{i,j-1} \end{aligned} \tag{A1}$$

and the driving vector $\underline{\sigma}$ by

$$\sigma_i = X_i + B_{i,1}W_{i+1} - A_{i,1}W_i \tag{A2}$$

where

$$\begin{aligned} A_{i,0} &= 1 - \tfrac{1}{3}h_i^2 g_i, & A_{i,1} &= h_i(1 - \tfrac{1}{3}h_i f_i) \\ B_{i,0} &= 1 + \tfrac{1}{6}h_i^2 g_{i+1}, & B_{i,1} &= \tfrac{1}{6}h_i^2 f_{i+1} \\ C_{i,0} &= -\tfrac{1}{2}h_i g_i & C_{i,1} &= 1 - \tfrac{1}{2}h_i f_i \\ D_{i,0} &= \tfrac{1}{2}h_i g_{i+1} & D_{i,1} &= 1 + \tfrac{1}{2}h_i f_{i+1} \\ X_i &= -\tfrac{1}{6}h_i^2(\rho_{i+1} + 2\rho_i), & Y_i &= -\tfrac{1}{2}h_i(\rho_{i+1} + \rho_i) \\ h_i &= r_{i+1} - r_i \end{aligned} \tag{A3}$$

and

$$\begin{aligned} W_{i+1} &= (A_{i,1}Y_i - C_{i,1}X_i)/\Delta_{i+1} \\ E_{i+1} &= (C_{i,1}A_{i,0} - A_{i,1}C_{i,0})/\Delta_{i+1} \\ F_{i+1} &= (A_{i,1}D_{i,0} - C_{i,1}B_{i,0})/\Delta_{i+1} \\ \Delta_{i+1} &= C_{i,1}B_{i,1} - A_{i,1}D_{i,1}. \end{aligned} \tag{A4}$$

Because of the $(1/r)(\partial\phi_n/\partial r)$ term in Eq. (3), the $r=0$ point must be treated specially. Expanding the derivative in a Taylor series, and keeping the lowest non-vanishing term, we obtain, in place of Eq. (6d),

$$2T_{0,2}+g_0T_{0,0}=\rho_n(0) \tag{A5}$$

$$Q_{0,0}=D_{0,1}A_{0,0}-B_{0,1}C_{0,0}, \qquad Q_{1,0}=A_{1,1}E_1$$

$$Q_{0,1}=B_{0,1}D_{0,0}-D_{0,1}B_{0,0}, \qquad Q_{1,1}=A_{1,0}+A_{1,1}F_1-B_{1,1}E_2 \tag{A6}$$

$$\sigma_0=D_{0,1}X_0-B_{0,1}Y_0, \qquad \sigma_1=X_1+B_{1,1}W_2-A_{1,1}W_1 \tag{A7}$$

where

$$\begin{aligned} A_{0,0}&=1-\tfrac{1}{6}h_0^2g_0, & A_{0,1}&=h_0\\ B_{0,0}&=1+\tfrac{1}{6}h_0^2g_1, & B_{0,1}&=\tfrac{1}{6}h_0^2f_1\\ C_{0,0}&=-\tfrac{1}{4}h_0g_0, & C_{0,1}&=1\\ D_{0,0}&=\tfrac{1}{2}h_0g_1, & D_{0,1}&=1+\tfrac{1}{2}h_0f_1\\ X_0&=-\tfrac{1}{6}h_0^2(\rho_0+\rho_1), & Y_0&=-\tfrac{1}{4}h_0(\rho_0+2\rho_1) \end{aligned} \tag{A8}$$

and for the boundary condition $\phi_n(R)$ given,

$$\begin{aligned} Q_{N,N-1}&=0, & Q_{N,N}&=1\\ \sigma_N&=\phi_n(R). \end{aligned} \tag{A9}$$

ACKNOWLEDGMENTS

This work was supported in part by the Naval Surface Weapons Center (Contract N60921-82-C-A185) and by the Air Force Office of Scientific Research (Contract F49620-79-C-0191).

REFERENCES

1. R. W. HOCKNEY, *J. Assoc. Comput. Mach.* **12** (1965), 95.
2. R. C. LE BAIL, *J. Comput. Phys.* **9** (1972), 440.
3. R. B. WILHELMSON AND J. H. ERICKSEN, *J. Comput. Phys.* **25** (1977), 319.
4. R. A. JAMES, *J. Comput. Phys.* **25** (1977), 71.
5. D. J. FYLE, *Comput. J.* **12** (1969), 188.
6. A quite different algorithm utilizing a spline interpolation of the electric field has been described by G. KNORR, G. JOYCE, AND A. J. MARCUS, *J. Comput. Phys.* **38** (1980), 227.
7. Y. TZENG AND E. E. KUNHARDT, *Bull. Amer. Phys. Soc.* **28** (1983), 180.
8. S. K. DHALI AND P. F. WILLIAMS, *in* "Proceedings, 4th IEEE Pulse Power Conference (1983)," in press.
9. See, for example, D. POTTER, "Computational Physics," pp. 99–101, Wiley, New York, 1977.