JOURNAL OF COMPUTATIONAL PHYSICS 22, 215–228 (1976)

An Efficient Method for a Finite-Difference Solution of the Poisson Equation on the Surface of a Sphere

SAMUEL Y. K. YEE

Air Force Geophysics Laboratory, Bedford, Massachusetts 01731

Received September 8, 1975; revised November 17, 1975 and March 1, 1976

An efficient shooting method is presented for the numerical solution of a discrete Poisson equation on the surface of the sphere. The solution is computed via two-dimensional shooting in the physical domain while the "missing initial conditions" needed to start the shooting are obtained in a one-dimensional setting in the Fourier domain. For a computational grid with $J \times J$ grid points, the operational count of this method is of $O((J^2/m) \log_2 J)$, where m is the number of grid points within each shooting subrange. The actual count is, for most practical purposes, less than 26 arithmetic operations per grid point. Stability of the method is a problem; this problem, however, can be overcome by the use of the multiple shooting technique. Numerical examples are given to demonstrate the applicability of the procedure.

1. INTRODUCTION

Consider the Poisson equation on the surface of a sphere of radius r,

$$\frac{1}{r^2 \sin\theta}\left(\frac{\partial}{\partial\theta}\sin\theta\frac{\partial}{\partial\theta} + \frac{1}{\sin\theta}\frac{\partial^2}{\partial\lambda^2}\right) u = f(\theta, \lambda). \tag{1.1}$$

Here $0 \leqslant \theta \leqslant \pi$ is the colatitude and $0 \leqslant \lambda \leqslant 2\pi$ is the longitude. The following features of (1.1) may be noted. (a) It has coordinate singularities at the poles $\theta = 0$ and $\theta = \pi$. If local values of u are to be sought for all (θ, λ), these singularities will first have to be removed in some way. (b) Even with the singularities removed, unless one additional constraint is imposed, (1.1) does not have a unique solution. (c) If one additional piece of independent information is available, we may consider (1.1) to be a boundary value problem. A unique solution can then be had by ensuring that $f(\theta, \lambda)$ satisfies the *compatibility condition*

$$\int_S f(\theta, \lambda)\, ds = 0. \tag{1.2}$$

Here $ds = \sin\theta\, d\theta\, d\lambda$; and the integration is over the entire surface of the sphere S. As a result of observational or numerical errors, $f(\theta, \lambda)$ in practice often does not satisfy (1.2). In such cases, $f(\theta, \lambda)$ may be perturbed by a constant so that (1.2) holds. The resulting solution is then a least squares solution of the unperturbed system (1.1) (e.g., [4]).

The conventional direct method for the numerical finite-difference solution of (1.1) makes use of the Fourier transform to reduce the two-dimensional equation to a number of one-dimensional difference equations. These one-dimensional equations are solved by Gaussian elimination. The desired solution to (1.1) is then given by the inverse transforms of the solutions to these one-dimensional equations (e.g., [4, 5]). We shall describe here a procedure to solve (1.1), which makes use of a shooting method for two-point boundary value problems. The shooting itself is conducted in the physical domain on the surface of the sphere. Only the "missing initial conditions" are computed in the Fourier domain. For a computational domain with a large number of grid points, this method is more efficient than the traditional direct method because it reduces the number of data points to be Fourier transformed to that of the number of starting points. It has the disadvantage of being numerically unstable. This difficulty, however, can be alleviated by the use of multiple shooting.

We shall set the stage in Section 2 for the use of this method in one-dimensional problems. The application of the method for a finite difference solution of (1.1) is developed in Section 3. Computational details and sample numerical results are given in Section 4.

2. A Finite-difference Approximation

Since we are interested mainly in the application of the procedure, we shall exclude from considerations the coordinate singularities at the poles by adopting a discrete spherical grid proposed by Merilees [3]. In this grid, the poles are not grid points, and we do not consider explicitly values of u at the poles in terms of (1.1). For a computational domain with $2I$ and J grid points along the θ and λ coordinates, respectively, a five-point centered-difference operator for the Laplacian leads to the following difference equation for (1.1).

$$a_i u_{i-1,j} - (a_i + b_i)\, u_{i,j} + b_i u_{i+1,j} + c_i(u_{i,j-1} - 2u_{i,j} + u_{i,j+1}) = f_{i,j}\,. \quad (2.1)$$

Here, for $1 \leqslant i \leqslant 2I$, $1 \leqslant j \leqslant J$

$$\begin{aligned} u_{i,j} &= u(\theta_i, \lambda_j), & a_i &= \frac{\sin \theta_{i-(1/2)}}{\Delta\theta^2 \sin \theta_i}, \\ f_{i,j} &= f(\theta_i, \lambda_j), & & \\ \theta_i &= (i - \tfrac{1}{2})\,\Delta\theta, & b_i &= \frac{\sin \theta_{i+(1/2)}}{\Delta\theta^2 \sin \theta_i}, \\ \lambda_j &= j\,\Delta\lambda, & & \\ \Delta\theta &= \pi/2I, & c_i &= \frac{1}{\Delta\lambda^2 \sin^2 \theta_i}\,. \\ \Delta\lambda &= 2\pi/J, & & \end{aligned}$$

Together with the "*boundary conditions*" [3]

$$\begin{aligned} u_{0,j} &= u_{1,(J/2)+j}, & 1 &\leqslant j \leqslant J/2, \\ &= u_{1,j-(J/2)}, & (J/2)+1 &\leqslant j \leqslant J, \\ u_{2I+1,j} &= u_{2I,(J/2)+j}, & 1 &\leqslant j \leqslant J/2, \\ &= u_{2I,j-(J/2)}, & (J/2)+1 &\leqslant j \leqslant J, \end{aligned} \tag{2.1a}$$

and the *auxiliary conditions*

$$\left.\begin{aligned} u_{i,0} &= u_{i,J}, \\ u_{i,J+1} &= u_{i,1}, \end{aligned}\right. \quad 0 \leqslant i \leqslant 2I+1, \tag{2.1b}$$

the system (2.1) is then closed. This linear algebraic system is however, singular. We shall see later, in the discussion of (2.7), that a unique solution exists if one additional piece of information is available.

To solve (2.1), define

$$\mathbf{u}_i = \begin{pmatrix} u_{i,1} \\ \vdots \\ u_{i,j} \\ \vdots \\ u_{i,J} \end{pmatrix}, \qquad \mathbf{f}_i = \begin{pmatrix} f_{i,1} \\ \vdots \\ f_{i,j} \\ \vdots \\ f_{i,J} \end{pmatrix},$$

$$R = \begin{pmatrix} 2 & -1 & & -1 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ -1 & & -1 & 2 \end{pmatrix}.$$

Then (2.1) becomes, for $1 \leqslant i \leqslant 2I$,

$$a_i \mathbf{u}_{i-1} - (a_i + b_i)\,\mathbf{u}_i + b_i \mathbf{u}_{i+1} - c_i R \mathbf{u}_i = \mathbf{f}_i . \tag{2.2}$$

Note that although the boundary conditions $\mathbf{u}_0$ and $\mathbf{u}_{2I+1}$ are defined via (2.1a) as a function of $\mathbf{u}_1$ and $\mathbf{u}_{2I}$, respectively, we do not have to consider them because both a_1 and b_{2I} are equal to zero. Furthermore, the matrix R, although singular, is real symmetric and is therefore orthogonally simiar to a real diagonal matrix D,

$$D = P^{-1} R P.$$

Here the diagonal elements of D are the eigenvalues $d_{k,k}$ of R, and P is a $J \times J$ orthogonal matrix whose columns are the normalized eigenvectors associated with $d_{k,k}$, i.e.,

$$d_{k,k} = 2(1 - \cos \lambda_k), \qquad 1 \leqslant k \leqslant K,$$

$$p_{j,k} = \left(\frac{2}{J}\right)^{1/2} \begin{pmatrix} \cos j\lambda_k, & 1 \leqslant k \leqslant (K/2) - 1, \\ (\cos j\lambda_k)/2^{1/2}, & k = K/2, \\ -\sin j\lambda_k, & (K/2) + 1 \leqslant k \leqslant K - 1, \\ 1/2^{1/2}, & k = K, \end{pmatrix}, \qquad 1 \leqslant j \leqslant J,$$

where $K = J$.

If we perform a discrete Fourier transform on $\mathbf{u}_i$ and $\mathbf{f}_i$,

$$\mathbf{w}_i = P^{-1}\mathbf{u}_i, \qquad \mathbf{g}_i = P^{-1}\mathbf{f}_i, \tag{2.3}$$

where

$$\mathbf{w}_i = \begin{pmatrix} w_{i,1} \\ \vdots \\ w_{i,k} \\ \vdots \\ w_{i,K} \end{pmatrix}, \qquad \mathbf{g}_i = \begin{pmatrix} g_{i,1} \\ \vdots \\ g_{i,k} \\ \vdots \\ g_{i,K} \end{pmatrix}.$$

We may write (2.2) in a space-Fourier domain, for $1 \leqslant i \leqslant 2I$,

$$a_i \mathbf{w}_{i-1} - (a_i + b_i)\, \mathbf{w}_i + b_i \mathbf{w}_{i+1} - c_i D \mathbf{w}_i = \mathbf{g}_i. \tag{2.4}$$

Thus, at latitude θ_i, for example, the kth Fourier component equation of (2.4) is

$$a_i w_{i-1,k} - (a_i + b_i)\, w_{i,k} + b_i w_{i+1,k} - c_i d_{k,k} w_{i,k} = g_{i,k}. \tag{2.5}$$

Noting that $w_{i,k}$ is the kth discrete Fourier component of $\mathbf{u}_i$, we now group $w_{i,k}$ by components and define

$$\mathbf{w}_k = \begin{pmatrix} w_{1,k} \\ \vdots \\ w_{i,k} \\ \vdots \\ w_{2I,k} \end{pmatrix}, \qquad \mathbf{g}_k = \begin{pmatrix} g_{1,k} \\ \vdots \\ g_{i,k} \\ \vdots \\ g_{2I,k} \end{pmatrix}.$$

Equation (2.4) becomes, for each $1 \leqslant k \leqslant K$,

$$T_k \mathbf{w}_k = \mathbf{g}_k, \tag{2.6}$$

where

$$T_k = \begin{pmatrix} e_{1,k} & b_1 & & \\ a_2 & \ddots & \ddots & \\ & \ddots & \ddots & b_{2I-1} \\ & & a_{2I} & e_{2I,k} \end{pmatrix},$$

$$e_{i,k} = -(a_i + b_i + c_i d_{k,k}).$$

For $k < K$, T_k is nonsingular, and (2.6) can be solved efficiently by Gaussian elimination. For the case of $k = K$, since $d_{K,K} = 0$, the system

$$T_K \mathbf{w}_K = \mathbf{g}_K \tag{2.7}$$

is singular and thus has no unique solution. At this point, one may argue that since the solution for a Poisson equation on the surface of a sphere can be determined only to within an additive constant, we are free to pick arbitrarily a value for $w_{1,K}$ to solve (2.7).

Thus, the traditional direct method of computing a finite-difference solution for the Poisson equation is to Fourier-decompose along latitudes the forcing function $f_{i,j}$, solve by Gaussian elimination a tridiagonal system for each of the Fourier components, and finally Fourier-synthesize the solutions of these tridiagonal systems to yield the solution for (2.1). If the labor of computing $d_{k,k}$ and $p_{j,k}$ is ignored, such a method takes, for a domain of $2IJ$ gridpoints, approximately $2IJ(10 + 4 \log_2 J)$ arithmetic operations. We shall see in Section 3 that (2.1) can be solved more efficiently, though perhaps less accurately, by the use of a shooting method.

3. Method of Solution

Tridiagonal systems such as (2.6) can be solved easily by a shooting method. We shall develop here an algorithm for such a method. Since the method is to be applied for all k in (2.6), the subscript k will be dropped for brevity.

Pick an arbitrary value $\hat{w}_1$ and consider it to be the value of w_1 in (2.6). Define

$$\hat{\epsilon}_1 = \hat{w}_1 - w_1, \tag{3.1}$$

where $\hat{\epsilon}_1$ is the error committed in taking $\hat{w}_1$ as the correct value of w_1. $\hat{w}_2$ is then given by

$$\hat{w}_2 = (g_1 - e_1 \hat{w}_1)/b_1 .$$

In general, for $2 \leqslant i \leqslant 2I - 1$, we have from (2.6)

$$\hat{w}_{i+1} = (g_i - a_i\hat{w}_{i-1} - e_i\hat{w}_i)/b_i . \tag{3.2}$$

After $\hat{w}_{2I-1}$, $\hat{w}_{2I}$ have been computed, we may calculate

$$g_{2I}^* = a_{2I}\hat{w}_{2I-1} + e_{2I}\hat{w}_{2I} , \tag{3.3}$$

which is, in general, different from g_{2I}. Thus the quantity $\Delta g_{2I} = g_{2I}^* - g_{2I}$ represents the cumulative error due to $\hat{\epsilon}_1$. Since for a given coefficient matrix T in (2.6), the amplification rate of $\hat{\epsilon}_1$ is defined, we can compute $\hat{\epsilon}_1$ from a knowledge of Δg_{2I}. Once $\hat{\epsilon}_1$ is determined, the solution to (2.6) may be obtained by the application of (3.2) for the second time, starting with the correct value of $w_1 = \hat{w}_1 - \hat{\epsilon}_1$.

To compute $\hat{\epsilon}_1$, we first determine its amplification factor from a marching solution of the homogeneous equations to (2.6),

$$T\boldsymbol{\alpha} = 0, \tag{3.4}$$

where $\boldsymbol{\alpha}$ is a $2I \times 1$ vector. One solution to (3.4) can be had by setting $\alpha_1 = 1$, which permits us to compute $\alpha_2 = -e_1\alpha_1/b_1$. And in general, for $2 \leqslant i \leqslant 2I - 1$,

$$\alpha_{i+1} = -(a_i\alpha_{i-1} + e_i\alpha_i)/b_i . \tag{3.4a}$$

After α_{2I} has been computed, $\hat{\epsilon}_1$ is given by

$$\hat{\epsilon}_1 = \Delta g_{2I}/\alpha_{2I} . \tag{3.5}$$

And the problem is then essentially solved.

Unfortunately, the method given here, although efficient, is also numerically unstable for problems of more than one dimension because of the large amplification rate for $\hat{\epsilon}_1$. In such cases, it is often desirable to use the so-called multiple shooting technique, in which arbitrary values at more than one point are adopted to start the first shot. Such an extension of the method in essence divides the range over which the shooting is to be conducted to smaller subranges. Parallel shootings are then conducted within each subrange. In terms of algebra, this is nothing more than the partitioning of a system such as (2.6) into a number of subsystems and then the solving of these subsystems simultaneously by shooting. For pedagogic purposes, we shall develop here in detail a two-subinterval multiple shooting algorithm. Extension of the algorithm for a computational domain with a larger number of subintervals is straightforward.

Consider the case in which parallel shootings are to be conducted from both the north and south polar regions toward the equator. We may then partition (2.6) into two simultaneous systems.

$$\begin{pmatrix} e_1 & b_1 & & \\ a_2 & \ddots & \ddots & \\ & \ddots & \ddots & b_{I-1} \\ & & a_I & e_I \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_{I-1} \\ w_I \end{pmatrix} = \begin{pmatrix} g_1 \\ \vdots \\ g_{I-1} \\ g_I' \end{pmatrix}, \tag{3.6a}$$

$$\begin{pmatrix} e_{2I} & a_{2I} & & \\ b_{2I-1} & \ddots & \ddots & \\ & \ddots & \ddots & a_{I-1} \\ & & b_{I+1} & e_{I+1} \end{pmatrix} \begin{pmatrix} w_{2I} \\ \vdots \\ w_{I+2} \\ w_{I+1} \end{pmatrix} = \begin{pmatrix} g_{2I} \\ \vdots \\ g_{I+2} \\ g_{I+1}' \end{pmatrix}, \tag{3.6b}$$

where $g_I' \equiv g_I - b_I w_{I+1}^N$, $g_{I+1}' \equiv g_{I+1} - a_{I+1} w_I^S$. Notice that the artificial internal boundary conditions w_{I+1}^N and w_I^S must satisfy the *auxiliary conditions*

$$\begin{aligned} w_I^S &= w_I, \\ w_{I+1}^N &= w_{I+1}. \end{aligned} \tag{3.6c}$$

It is through these auxiliary conditions that (3.6a) and (3.6b) are coupled. This is because the boundary point of one subinterval is also an interior point of a neighboring subinterval. Furthermore, the coefficient matrices in (3.6a) and (3.6b) are identical because the partitioning of (2.6) into (3.6a) and (3.6b) effectively divides a sphere into halves at the equator. The homogeneous equations associated with these systems are identical. We thus have

$$T'\boldsymbol{\alpha} = 0, \tag{3.7}$$

where T' is the coefficient matrix in (3.6) and $\boldsymbol{\alpha}$ is now a $I \times 1$ vector. Our task is to solve (3.6) simultaneously using the shooting method.

To do this, pick arbitrary values $\hat{N}_1$ and $\hat{S}_{2I}$; and consider them to be the values of w_1 and w_{2I} in (3.6). Define

$$\begin{aligned} \hat{\epsilon}_1 &= \hat{N}_1 - w_1, \\ \hat{\epsilon}_{2I} &= \hat{S}_{2I} - w_{2I}. \end{aligned} \tag{3.8}$$

Thus $\hat{\epsilon}_1$ and $\hat{\epsilon}_{2I}$ are, respectively, the errors committed in taking $\hat{N}_1$ and $\hat{S}_{2I}$ as the correct values for w_1 and w_{2I}. With these arbitrarily picked $\hat{N}_1$ and $\hat{S}_{2I}$, we may now compute a marching solution to (3.6):

$$\begin{aligned} \hat{N}_{i+1} &= (g_i - a_i\hat{N}_{i-1} - e_i\hat{N}_i)/b_i, \qquad 2 \leqslant i \leqslant I, \\ \hat{S}_{i-1} &= (g_i - e_i\hat{S}_i - b_i\hat{S}_{i+1})/a_i, \qquad 2I - 1 \geqslant i \geqslant I + 1, \end{aligned} \tag{3.9}$$

where $\hat{N}_2 = (g_1 - e_1\hat{N}_1)/b_1$ and $\hat{S}_{2I-1} = (g_{2I} - e_{2I}\hat{S}_{2I})/a_{2I}$. Had the marching solution satisfied the auxiliary conditions (3.6c),

$$\hat{S}_I = \hat{N}_I,$$
$$\hat{N}_{I+1} = \hat{S}_{I+1},$$

we would have had the correct solution to (3.6), namely,

$$\begin{aligned} w_i &= \hat{N}_i, \qquad 1 \leqslant i \leqslant I, \\ &= \hat{S}_i, \qquad 2I \geqslant i \geqslant I+1. \end{aligned}$$

Since $\hat{N}_1$ and $\hat{S}_{2I}$, the starting values for w_1 and w_{2I}, have been chosen arbitrarily, conditions (3.6c) are usually not satisfied. We know, however, that in the process of marching for $\hat{N}_i$ and $\hat{S}_i$, the errors due to $\hat{\epsilon}_1$ and $\hat{\epsilon}_{2I}$ are amplified according to (3.7). Thus the errors in $\hat{N}_I$ and $\hat{S}_I$ at latitude θ_I due to $\hat{\epsilon}_1$ and $\hat{\epsilon}_{2I}$ are

$$\begin{aligned} \alpha_I\hat{\epsilon}_1 &= \hat{N}_I - w_I, \\ \alpha_{I+1}\hat{\epsilon}_{2I} &= \hat{S}_I - w_I, \end{aligned}$$

where $\alpha_{I+1} = -(a_I\alpha_{I-1} - e_I\alpha_I)/b_I$. On eliminating w_I, we have

$$\alpha_I\hat{\epsilon}_1 - \alpha_{I+1}\hat{\epsilon}_{2I} = \hat{N}_I - \hat{S}_I \equiv \Delta w_I. \tag{3.10a}$$

Similarly, $\hat{N}_{I+1}$ and $\hat{S}_{I+1}$ at latitude θ_{I+1} give a second equation for the unknowns $\hat{\epsilon}_1$ and $\hat{\epsilon}_{2I}$,

$$\alpha_{I+1}\hat{\epsilon}_1 - \alpha_I\hat{\epsilon}_{2I} = \hat{N}_{I+1} - \hat{S}_{I+1} \equiv \Delta w_{I+1}. \tag{3.10b}$$

Thus once the coefficients α_I and α_{I+1} have been determined via (3.7) in a manner similar to (3.4a), $\hat{\epsilon}_1$ and $\hat{\epsilon}_{2I}$ can be computed via the 2×2 system (3.10). In the cases where the number of subintervals is large, the corrections to the arbitrarily picked initial conditions may be computed more efficiently by solving a number of tridiagonal systems. For example, for the case of the two-subinterval shooting, we may compute $\hat{\epsilon}_1$ and $\hat{\epsilon}_{2I}$ by solving in sequence the systems

$$\begin{pmatrix} e_1 & b_1 & & \\ a_2 & \ddots & \ddots & \\ & \ddots & e_{I-1} & b_{I-1} \\ & & a_I & e_I' \end{pmatrix} \begin{pmatrix} \hat{\epsilon}_1 + \hat{\epsilon}_{2I} \\ \vdots \\ \hat{\epsilon}_{I-1} + \hat{\epsilon}_{I+2} \\ \hat{\epsilon}_I + \hat{\epsilon}_{I+1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ g_I^* + g_{I+1}^* \end{pmatrix}, \tag{3.11a}$$

$$\begin{pmatrix} e_1 & b_1 & & \\ a_2 & \ddots & \ddots & \\ & \ddots & e_{I-1} & b_{I-1} \\ & & a_I & e_I'' \end{pmatrix} \begin{pmatrix} \hat{\epsilon}_1 \\ \vdots \\ \hat{\epsilon}_{I-1} \\ \hat{\epsilon}_I \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ g_I^* - b_I(\hat{\epsilon}_I + \hat{\epsilon}_{I+1}) \end{pmatrix} \tag{3.11b}$$

Here

$$\begin{aligned} e_I' &= e_I + b_I, \\ e_I'' &= e_I - b_I, \\ g_I^* &= b_I(\hat{S}_{I+1} - \hat{N}_{I+1}), \\ g_{I+1}^* &= a_{I+1}(\hat{N}_I - \hat{S}_I). \end{aligned}$$

It should be emphasized that although the method has been developed here, for illustrative purposes, in a one-dimensional setting, it is not recommended for one-dimensional problems because it is no more efficient and yet may be less accurate than the standard Gaussian elimination. The virtues of the method vest in its ability to handle multi-dimensional problems efficiently.

4. Computational Details and Numerical Examples

The application of this technique to the discrete two-dimensional problem (2.1) makes use of the following observations. Since $a_1 = 0$ and $b_{2I} = 0$, if $u_{1,j}$ and $u_{2I,j}$ are known, we may then compute $u_{2,j}$ and $u_{2I-1,j}$ for all $1 \leqslant j \leqslant J$. In general, we may compute, for $1 \leqslant j \leqslant J$,

$$\begin{aligned} u_{i+1,j} &= [f_{i,j} - a_i u_{i-1,j} + (a_i + b_i + 2c_i)\, u_{i,j} - c_i(u_{i,j-1} + u_{i,j+1})]/b_i, \\ &\qquad\qquad 2 \leqslant i \leqslant I, \\ u_{i-1,j} &= [f_{i,j} + (a_i + b_i + 2c_i)\, u_{i,j} - b_i u_{i+1,j} - c_i(u_{i,j-1} + u_{i,j+1})]/a_i, \\ &\qquad\qquad 2I - 1 \geqslant i \geqslant I + 1. \end{aligned}$$

Thus the problem of solving (2.1) becomes that of obtaining correct values for $u_{1,j}$ and $u_{2I,j}$, $1 \leqslant j \leqslant J$. This may be accomplished by the algorithm

(a) Set $N_{1,j} = 0$, $S_{2I,j} = 0$, $0 \leqslant j \leqslant J + 1$.

(b)

$$\begin{aligned} N_{2,j} &= [f_{1,j} + (b_1 + 2c_1)\, N_{1,j} - c_1(N_{1,j-1} + N_{1,j+1})]/b_1, \qquad 1 \leqslant j \leqslant J, \\ N_{2,0} &= N_{2,J}, \\ N_{2,J+1} &= N_{2,1}. \end{aligned}$$

(b′)

$$\begin{aligned} S_{2I-1,j} &= [f_{2I,j} + (a_{2I} + 2c_{2I})\, S_{2I,j} - c_{2I}(S_{2I,j-1} + S_{2I,j+1})]/a_{2I}, \qquad 1 \leqslant j \leqslant J, \\ S_{2I-1,0} &= S_{2I-1,J}, \\ S_{2I-1,J+1} &= S_{2I-1,1}. \end{aligned}$$

(c) Compute, for $2 \leqslant i \leqslant I$,

$$N_{i+1,j} = [f_{i,j} - a_i N_{i-1,j} + (a_i + b_i + 2c_i) N_{i,j} - c_i(N_{i,j-1} + N_{i,j+1})]/b_i, \quad 1 \leqslant j \leqslant J,$$

$$N_{i+1,0} = N_{i+1,J},$$

$$N_{i+1,J+1} = N_{i+1,1}.$$

(c′) Compute, for $2I \geqslant i \geqslant I + 1$,

$$S_{i-1,j} = [f_{i,j} + (a_i + b_i + 2c_i) S_{i,j} - b_i S_{i+1,j} - c_i(S_{i,j-1} + S_{i,j+1})]/a_i, \quad 1 \leqslant j \leqslant J,$$

$$S_{i-1,0} = S_{i-1,J},$$

$$S_{i-1,J+1} = S_{i-1,1}.$$

(d) Compute, for $1 \leqslant j \leqslant J$,

$$\Delta u_{I,j} = N_{I,j} - S_{I,j},$$
$$\Delta u_{I+1,j} = N_{I+1,j} - S_{I+1,j}.$$

(e) Fourier decompose, via (2.3), $\Delta u_{I,j}$ and $\Delta u_{I+1,j}$ to give $\Delta w_{I,k}$ and $\Delta w_{I+1,k}$.

(f) Compute, for $1 \leqslant k \leqslant K - 1$, $\alpha_{I,k}$ via (3.7) as indicated in (3.4a).

(g) For $1 \leqslant k \leqslant K - 1$, compute $\hat{\epsilon}_{1,k}$ and $\hat{\epsilon}_{2I,k}$ via (3.10) or (3.11). For $k = K$, $\hat{\epsilon}_{1,K} = 0$, $\hat{\epsilon}_{2I,K} = -\Delta w_{I,K}$.

(h) Fourier synthesize, via (2.3), $\hat{\epsilon}_{1,k}$ and $\hat{\epsilon}_{2I,k}$ to give $\epsilon_{1,j}$ and $\epsilon_{2I,j}$.

(i) Compute, for $1 \leqslant j \leqslant J$, the missing initial conditions

$$u_{1,j} = N_{1,j} - \epsilon_{1,j},$$
$$u_{2I,j} = S_{2I,j} - \epsilon_{2I,j}.$$

(j) Shoot for the second time via steps (b) and (c). This time start with the correct values for $u_{1,j}$ and $u_{2I,j}$ from step (i) above.

Thus system (2.1) can be solved in two shots. The first misses the targets by $\Delta u_{I,j}$ at latitude θ_I and $\Delta u_{I+1,j}$ at θ_{I+1}. These missed "distances" enable us to obtain the correct $u_{1,j}$ and $u_{2I,j}$, to be used as the starting values for the second shot. It should be emphasized that this is not an iterative scheme in which an iterate at each iteration is altered to approach an asymptotic value. Instead, (2.1) is viewed here as an initial value problem, with initial conditions specified only implicitly. Steps (a) through (i), above, enable us to obtain explicitly the values of these "missing" initial conditions. With these initial conditions known, we then solve the initial value problem in step (j) by marching.

Notice that the Fourier decomposition in step (e) permits us to consider each Fourier component as a one-dimensional problem in the computation of $\hat{\epsilon}_{1,k}$

and $\hat{\epsilon}_{2I,k}$. The Fourier synthesis in step (h) permits us to conduct the actual shooting in a two-dimensional physical domain. The efforts made to reduce a two-dimensional problem to K one-dimensional problems, as discussed in Section 2, are thus justified. Such a formulation, while permitting us to discuss the shooting method in a one-dimensional setting in Section 3, nevertheless, allows us to conduct the shooting itself in a two-dimensional physical domain.

The order of arithmetic operations involved in each of the steps above is listed in Table I. Here the counts are given for a multiple shooting algorithm in which m

TABLE I

Order of Arithmetic Operations for Various Steps in the Algorithm

Step	Order of operations	Approximate number of operations
(c)	$O(J^2)$	$8J^2$
(d)	$O(J^2/m)$	J^2/m
(e)	$O((J^2/m)\log_2 J)$	$2(J^2/m)\log_2 J$
(f)	$O(J^2)$	$2J^2$
(g)	$O(J^2)$	$4J^2$
(h)	$O((J^2/m)\log_2 J)$	$2(J^2/m)\log_2 J$
(i)	$O(J^2/m)$	J^2/m
(j)	$O(J^2)$	$8J^2$

is the number of grid points wthin each shooting subrange. For convenience of comparison with other existing direct methods, the operation counts have been given for the case of a $J \times J$ square grid. It can be seen from this table that this may be considered as an $O((J^2/m)\log_2 J)$method since steps (e) and (h) both require $O((J^2/m)\log_2 J)$ operations. At the suggestion of one of the reviewers, the approximate number of arithmetic operations for each of the steps is also given in Table I. We see that the approximate total number of operations is

$$\text{total counts} = [22 + (4\log_2 J)/m + 2/m]\,J^2.$$

Note that for most practical purposes, the first term inside the square brackets usually dominates. For example, for $m = 8$, even with $J = 256$, the second term inside the brackets, $(4\log_2 J)/m$, has a numerical value of 4. This is only about 20 percent of the first term. Thus, we may say that for most practical purposes, this method requires a total of less than $26J^2$ arithmetic operations for a $J \times J$ grid. Compared with the approximate operation count of $(10 + 4\log_2 J)\,J^2$ in the traditional direct method outlined in Section 2, it is apparent that the multiple shooting method is more efficient for cases with $J > 16$. Bank and Rose [2] have recently given an $O(J^2)$ method for constant coefficient boundary value problems in two dimensions. For the linear algebraic system (2.1), their method requires, however, $O(J^2 \log_2(J/m))$ operations [1].

To gain some insight into the numerical properties of the procedure described in Section 3, test computations have been conducted for the solution of (2.1). Since we are interested here only in the effect of the accumulation of the machine round-off error and not the discretization error of the difference equation, we created a set of true solution for $u_{i,j}$ in (2.1) by computing $f_{i,j}$ from

$$f_{i,j} = a_i v_{i-1,j} - (a_i + b_i + 2c_i)\, v_{i,j} + b_i v_{i+1,j} + c_i(v_{i,j-1} + v_{i,j+1}).$$

Here values for $v_{i,j}$ are obtained from a random number generator and have been subjected to the constraint $\sum_i \sum_j v_{i,j} = 0$ so that $f_{i,j}$ satisfies the discrete form of the *compatibility condition* (1.2)

$$\sum_i \sin\theta_i \left(\sum_j f_{i,j}\right) = 0. \tag{4.1}$$

With the forcing function $f_{i,j}$ computed in this manner, a normalized error norm defined by

$$\| E \|_2 = \| u - v \|_2 / \| v \|_2 \,. \tag{4.2}$$

may then be considered as a measure of the accuracy of the numerical procedure. The number of digits of accuracy in u is then given by

$$Z = -\log_{10} \| E \|_2 \tag{4.3}$$

Several sets of computations have been made for various values of m. For the two-subinterval results reported here, we start the shooting from both poles and march toward the equatorial regions. Sample results are tabulated here in Table II

TABLE II

Numerical Error as a Function of the Length of the Shooting Subrange m

m	Z^a	$\| E \|_2$	E(max)
2	12.6	2.6×10^{-13}	4.9×10^{-13}
3	11.4	3.6×10^{-12}	6.1×10^{-12}
5	9.5	3.0×10^{-10}	4.2×10^{-10}
8	6.9	1.3×10^{-7}	1.9×10^{-7}
12	3.5	3.2×10^{-4}	7.2×10^{-4}
16	0.1	8.7×10^{-1}	1.7×10^{-0}
16D	1.9	1.2×10^{-2}	2.5×10^{-2}

[a] Z: Number of digits of accuracy in computed solution u.

in terms of Z, $\| E \|_2$ and $E(\max)$, the maximum error over the entire computational domain. It is obvious that the accuracy of this scheme depends critically on the length of the shooting subrange m. This is to be expected since, as may be seen from Table III, the maximum amplification factor for $\hat{\epsilon}$ (when $d_{k,k} = 4.0$) increases by 12 orders of magnitude over 16 grid points. Thus just as $\epsilon_{1,j}$ and $\epsilon_{2I,j}$ are amplified by this factor in a distance spanned by 16 grid points, so is the error due to

TABLE III

Approximate Values of the Maximum α_{I+1} (When $d_{k,k} = 4$) as a Function of I

I	$\mathrm{Max}(\alpha_{I+1})$
2	1×10^1
3	7×10^1
5	3×10^3
8	7×10^5
12	1×10^9
16	3×10^{12}

the machine round-off. It is of interest to note that the length of the shooting subrange m plus Z, the number of correct digits in u is roughly equal to the number of digits of accuracy of the machine used for the computation. Thus for example, for $m = 8$, we have $Z = 6.9$. (Our CDC 6600 has roughly a 15-digit accuracy.) In one experiment, values of the amplification factor α_i alone were computed for the case of $m = 16$ using double-precision arithmetics in the CDC 6600. The results are given in the last row of Table II. We see that this reduces the error norm only by two orders of magnitude. Thus, the possibility of increasing the accuracy of the scheme by computing accurately the amplification factors once and for all seems to have been ruled out. Perhaps the most effective way of ensuring accuracy is by the use of multiple shooting, choosing the subranges according to the accuracy requirement of the solution, and the numerical capability of the machine at hand.

Acknowledgment

I am indebted to Professor E. N. Lorenz of the Massachusetts Institute of Technology for suggesting this problem and for helpful conversations.

References

1. R. E. Bank, "Marching Algorithms for Elliptic Boundary Value Problems," TR12-15, Center for Research in Computing Technology, Harvard University, Cambridge, Mass., 1975.
2. R. E. Bank and D. J. Rose, *SIAM J. Numer. Anal.* **12** (1975), 529–540.

3. P. E. Merilees, *Atmosphere* **11** (1973), 13–20.
4. P. N. Swarztrauber, *J. Computational Phys.* **15** (1974), 46–54.
5. S. Y. Yee, "Noniterative Solution of a Boundary Value Problem of the Helmholtz Type," AFCRL 69-478, Air Force Cambridge Research Laboratories, Bedford, Mass., 1969.