A Fourth-order Accurate Fourier Method for the Helmholtz Equation in Three Dimensions *

Ronald F. Boisvert
Center for Applied Mathematics
National Bureau of Standards
Gaithersburg, MD 20899

December 1985

Abstract

We present fourth-order-accurate compact discretizations of the Helmholtz equation on rectangular domains in two and three dimensions with any combination of Dirichlet, Neumann, or periodic boundary conditions. The resulting systems of linear algebraic equations have the same block-tridiagonal structure as traditional central differences, and hence may be solved efficiently using the Fourier method. The performance of the method for a variety of test cases is presented, including problems with non-smooth solutions. The method is seen to be roughly twice as fast as deferred corrections, and in many cases results in a smaller discretization error.

Categories and Subject Descriptors: G.1.8 [Numerical Analysis] Partial differential equations — elliptic equations; G.4 [Mathematics of Computing] Mathematical Software — algorithm analysis

General Terms: Algorithms, Theory

Additional Keywords and Phrases: fast-direct method, finite differences, Fourier method, Helmholtz equation, high-order accuracy, HODIE method

1 Introduction

Fast direct methods have proven to be very effective techniques for solving separable elliptic boundary value problems. These include cyclic reduction, Fourier analysis,

*Contribution of the National Bureau of Standards, not subject to copyright in the United States. This paper was published as *ACM Trans. Math. Softw.* 13 (1987) pp. 221–234.

FACR(l), and tensor product matrix decomposition [6, 25]. Each exploits, in some way, the tensor-product nature of the discrete problem obtained by applying standard second-order accurate finite differences on a rectangular domain. Problems on nonrectangular domains may be solved using fast-direct methods after suitable preprocessing to extend them to an enclosing domain [21, 17].

Recently, there has been much interest in combining fast-direct methods with high-order-accurate discretizations to produce software which is both very fast and very accurate. There are a variety of techniques for attaining high order accuracy while maintaining the special form of discrete problem required for fast-direct solution techniques. The accuracy of standard second-order finite differences can be extended using deferred corrections [19]. This is done for two-dimensional problems in the FISHPAK subprograms SEPELI and SEPX4 [1] to attain fourth-order accuracy. In [20] the Dirichlet problem for Laplace's equation is solved for nonrectangular planar regions to fourth-order accuracy using the capacitance matrix method and deferred corrections. A similar technique is employed to obtain sixth-order accuracy in [22].

An alternative to deferred corrections is the use of high-order-accurate compact finite differences. This was done in [10] to attain fourth and sixth order accuracy for the two-dimensional Helmholtz equation with Dirichlet boundary conditions. Similar techniques have also been applied to the three-dimensional Poisson equation [12, 13, 18]. Finite-element discretizations based upon tensor-products of one-dimensional spline functions also yield discrete problems to which fast-direct methods can be applied [9, 11].

In this paper I describe new fourth-order accurate compact discretizations of the two- and three-dimensional Helmholtz equations. I demonstrate how such compact discretizations can be extended to maintain fourth-order accuracy at boundary points where Neumann boundary conditions are specified. These discretizations are the basis for a suite of Fortran subprograms which solve the Helmholtz equation on rectangular domains in two and three dimensions with any combination of Dirichlet, Neumann, or periodic boundary conditions [5]. The discrete problem is solved using the Fourier method. This software differs from [10], [12], and [18] in the variety of boundary conditions handled, as well as in the details of the direct solution. The software described here is roughly twice as fast as fourth-order solvers based on deferred corrections, and does not require significantly more working storage.

2 HODIE Discretizations

We consider the problem

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} + \lambda u = g(x, y, z) \tag{1}$$

($\lambda < 0$ constant) on an open rectangular domain R in three dimensions; on the boundary $\partial R = \partial R_1 \cup \partial R_2 \cup \partial R_3$ we have

- u prescribed (Dirichlet condition) on ∂R_1
- u_n prescribed (Neumann condition) on ∂R_2
- the solution is periodic on ∂R_3

(∂R_1, ∂R_2, and ∂R_3 are each unions of sides of R.) To discretize the problem we first place a uniform grid of mesh width h over R (with NX by NY by NZ grid lines), and define $R_{ijk} = \{(x,y,z)|(x,y,z) \in [x_{i-1}, x_{i+1}] \times [y_{j-1}, y_{j+1}] \times [z_{k-1}, z_{k+1}]\}$.

2.1 Interior Point Discretizations

At grid points (x_i, y_j, z_k) in R we use the following discrete analogue of (1).

$$L_{ijk}U = h^2 I_{ijk} g \tag{2}$$

where

$$\begin{aligned} L_{ijk}U &= \sum_\ell \alpha_\ell U(p_\ell) \\ I_{ijk}g &= \sum_\ell \beta_\ell g(q_\ell) \end{aligned}$$

The set $\{p_\ell\}$ is called the discretization point set and the set $\{q_\ell\}$ is called the evaluation point set. Both $\{p_\ell\}$ and $\{q_\ell\}$ are subsets of R_{ijk}; $\{p_\ell\}$ contains only grid points, whereas $\{q_\ell\}$ does not have this restriction. The number and location of the evaluation points may be chosen to decrease the truncation error of the difference equations. Once an appropriate set of evaluation points has been selected one may obtain $\mathcal{O}(h^N)$ accuracy by choosing the coefficients in (2) so that the truncation error $L_{ijk}v - h^2 I_{ijk} L v = 0$ for all $v \in P^{N+1}$, the space of polynomials of degree $N+1$ or less. Discretizations of this form are studied in [2] and [14] where they are called HODIE methods; they are related to Mehrstellenverfahren [8] and operator compact implicit (OCI) methods [7]. Alternate methods for deriving such formulae are considered in [16].

Our discretization of (1) at the $(i,j,k)^{\text{th}}$ grid point uses operators L_{ijk} and I_{ijk} of the form

$$\begin{aligned} L_{ijk}U = \; & aU_{ijk} \\ & +b[U_{i+1,j,k} + U_{i,j+1,k} + U_{i,j,k+1} + U_{i-1,j,k} + U_{i,j-1,k} + U_{i,j,k-1}] \qquad (3) \\ & +c[U_{i+1,j,k+1} + U_{i,j+1,k+1} + U_{i+1,j+1,k} + U_{i+1,j,k-1} \\ & +U_{i,j+1,k-1} + U_{i+1,j-1,k} + U_{i-1,j,k+1} + U_{i,j-1,k+1} \\ & +U_{i-1,j+1,k} + U_{i-1,j,k-1} + U_{i,j-1,k-1} + U_{i-1,j-1,k}] \\ & +d[U_{i+1,j+1,k+1} + U_{i+1,j+1,k-1} + U_{i+1,j-1,k+1} + U_{i+1,j-1,k-1} \\ & +U_{i-1,j+1,k+1} + U_{i-1,j+1,k-1} + U_{i-1,j-1,k+1} + U_{i-1,j-1,k-1}] \end{aligned}$$

$$
\begin{aligned}
I_{ijk}g \;=\;& \beta_0 g_{ijk} \\
&+\beta_1[g_{i+1/2,j+1/2,k+1/2} + g_{i-1/2,j+1/2,k+1/2} + g_{i+1/2,j-1/2,k+1/2} \\
&+g_{i-1/2,j-1/2,k+1/2} + g_{i+1/2,j+1/2,k-1/2} + g_{i-1/2,j+1/2,k-1/2} \\
&+g_{i+1/2,j-1/2,k-1/2} + g_{i-1/2,j-1/2,k-1/2}] \\
&+\beta_2[g_{i+1,j,k} + g_{i-1,j,k} + g_{i,j+1,k} + g_{i,j-1,k} + g_{i,j,k+1} + g_{i,j,k-1}]
\end{aligned} \tag{4}
$$

with the coefficients $a = -24 + 5\Lambda - \Lambda^2/4$, $b = 2 - \Lambda/24 + \Lambda^2/48$, $c = 1 + 5\Lambda/48$, $d = 0$, $\beta_0 = 2 - \Lambda/4$, $\beta_1 = 1/2$, and $\beta_2 = \Lambda/48$, where $\Lambda = h^2\lambda$. This discretization has truncation error $\mathcal{O}(h^4)$ and is of positive type [8], hence for problems with Dirichlet and/or periodic boundary conditions it follows that the resulting global error is at most $\mathcal{O}(h^4)$. The discretization (3–4) reduces to the standard second-order accurate 7-point discretization of the Helmholtz equation with the choices $a = -6 + \Lambda$, $b = \beta_0 = 1$, $c = d = \beta_1 = \beta_2 = 0$. An alternate second-order discretization is obtained with the choices $a = -24 + 6\Lambda$, $b = 2$, $c = 1$, $\beta_0 = 6$, $d = \beta_1 = \beta_2 = 0$.

2.2 Discretization of Neumann Boundary Conditions

At grid points (x_i, y_j, z_k) in ∂R_2. we augment the difference equation to obtain

$$L_{ijk}U = h^2 I_{ijk}g + hJ_{ijk}u_n \tag{5}$$

where the functionals L_{ijk} and I_{ijk} are defined as in (2) with the point sets $\{p_\ell\}$ and $\{q_\ell\}$ restricted to $R_{ijk} \cup R$. The functional J_{ijk} is defined by

$$J_{ijk}s = \sum_\ell \gamma_\ell s(r_\ell)$$

where the boundary evaluation points r_ℓ are taken on ∂R_2 near (x_i, y_j, z_k). The evaluation points and coefficients are chosen so that the truncation error $L_{ijk}v - I_{ijk}Lv - J_{ijk}v_n = 0$ for all $v \in P^N$. Difference approximations of the general form (5) are described in more detail in [3].

The specific discretization considered here uses the L_{ijk} and I_{ijk} of (3–4), except that values of U and g outside the domain are defined by reflection through the boundary. A similar reflection symmetry is present in the standard second-order case obtained by applying (2) at a boundary point and then eliminating unknowns outside R using central difference approximations to the normal derivative. Not every choice of evaluation points in (2) leads to a boundary point discretization (5) with the proper reflection symmetry, as the following simple Theorem shows.

Theorem 1 *No boundary discretization of the form (5) with L_{ijk} of the form (3) with reflection symmetry and $\{q_\ell\} = \{p_\ell\}$ can be exact on P^4 unless $2b + 8c + 8d = 0$.*

proof: It is sufficient to consider the case $(x_i, y_j, z_k) = (0, 0, 0)$ with $x = x_{\mathrm{NX}} =$ and $u_n = u_x = s$. In this case we seek a discretization with $L_{ijk}u - h^2 I_{ijk}g - hJ_{ijk}s = 0$ for all $u \in P^4$ where

$$
\begin{aligned}
L_{ijk}u &= au(0,0,0) \\
&+b[2u(-h,0,0)+u(0,h,0)+u(0,-h,0)+u(0,0,h)+u(0,0,-h)] \\
&+c[2u(-h,h,0)+2u(-h,-h,0)+2u(-h,0,h)+2u(-h,0,-h) \\
&+u(0,h,h)+u(0,h,-h)+u(0,-h,h)+u(0,-h,-h)] \\
&+d[2u(-h,h,h)+2u(-h,h,-h)+2u(-h,-h,h)+2u(-h,-h,-h)]
\end{aligned}
$$

$$
\begin{aligned}
I_{ijk}g &= \beta_0 g(0,0,0)+\beta_1 g(0,h,0)+\beta_2 g(-h,0,0)+\beta_3 g(0,-h,0) \\
&+\beta_4 g(0,0,h)+\beta_5 g(0,0,-h)+\beta_6 g(-h,h,0)+\beta_7 g(-h,-h,0) \\
&+\beta_8 g(0,h,h)+\beta_9 g(-h,0,h)+\beta_{10} g(0,-h,h)+\beta_{11} g(0,h,-h) \\
&+\beta_{12} g(-h,0,-h)+\beta_{13} g(0,-h,-h)+\beta_{14} g(-h,h,h)+\beta_{15} g(-h,-h,h) \\
&+\beta_{16} g(-h,h,-h)+\beta_{17} g(-h,-h,-h)
\end{aligned}
$$

and $J_{ijk}s$ is any linear combination of evaluations of s along $x=0$. Let $\beta=\beta_2+\beta_6+\beta_7+\beta_9+\beta_{12}+\beta_{14}+\beta_{15}+\beta_{16}+\beta_{17}$. Taking $u=x^3$ and $u=x^4$ in turn yield the requirements that $\beta=2(b+4c+4d)/(6+h^2\lambda)$ and $\beta=2(b+4c+4d)/(12+h^2\lambda)$ for h sufficiently small. These cannot both be satisfied unless $b+4c+4d=0$. QED

Since discretizations of (1) of positive type have b, c, and d of one sign, the theorem implies that useful boundary discretizations consistent with (3) cannot have fourth-order accuracy unless evaluation points which are not grid points are included.

The exact form of the functional J_{ijk} which we use depends upon whether the point (x_i, y_j, z_k) is on a side, an edge, or a corner of the domain. At a point on the side $x=x_{\mathrm{NX}}$ (not an edge or corner) where $u_n=u_x=s$, J_{ijk} takes the form

$$
J_{\mathrm{NX},j,k}u_n=\gamma_0 s_{\mathrm{NX},j,k}+\gamma_1[s_{\mathrm{NX},j+1,k}+s_{\mathrm{NX},j-1,k}+s_{\mathrm{NX},j,k+1}+s_{\mathrm{NX},j,k-1}] \tag{6}
$$

At a point on the edge with $x=x_{\mathrm{NX}}$ and $y=y_{\mathrm{NY}}$ (not a corner) where $u_n=u_x=s$ for $x=x_{\mathrm{NX}}$ and $u_n=u_y=t$ for $y=y_{\mathrm{NY}}$, J_{ijk} takes the form

$$
\begin{aligned}
J_{\mathrm{NX},\mathrm{NY},k}s &= \delta_0[s_{\mathrm{NX},\mathrm{NY},k}+t_{\mathrm{NX},\mathrm{NY},k}] \\
&+\delta_1[s_{\mathrm{NX},\mathrm{NY}-1,k}+t_{\mathrm{NX}-1,\mathrm{NY},k}]+\delta_2[s_{\mathrm{NX},\mathrm{NY}-2,k}+t_{\mathrm{NX}-2,\mathrm{NY},k}] \\
&+\delta_3[s_{\mathrm{NX},\mathrm{NY}-3,k}+t_{\mathrm{NX}-3,\mathrm{NY},k}] \\
&+\delta_4[s_{\mathrm{NX},\mathrm{NY},k+1}+t_{\mathrm{NX},\mathrm{NY},k+1}+s_{\mathrm{NX},\mathrm{NY},k-1}+t_{\mathrm{NX},\mathrm{NY},k-1}] \\
&+\delta_5[s_{\mathrm{NX},\mathrm{NY}-1,k+1}+t_{\mathrm{NX}-1,\mathrm{NY},k+1}+s_{\mathrm{NX},\mathrm{NY}-1,k-1}+t_{\mathrm{NX}-1,\mathrm{NY},k-1}]
\end{aligned} \tag{7}
$$

At a point at the corner with $x=x_{\mathrm{NX}}$, $y=y_{\mathrm{NY}}$, and $z=z_{\mathrm{NZ}}$, where $u_n=u_x=s$ for $x=x_{\mathrm{NX}}$, $u_n=u_y=t$ for $y=y_{\mathrm{NY}}$, and $u_n=u_z=w$ for $z=z_{\mathrm{NZ}}$, J_{ijk} takes the form

$$
J_{\mathrm{NX},\mathrm{NY},\mathrm{NZ}}s = \epsilon_0[s_{\mathrm{NX},\mathrm{NY},\mathrm{NZ}}+t_{\mathrm{NX},\mathrm{NY},\mathrm{NZ}}+w_{\mathrm{NX},\mathrm{NY},\mathrm{NZ}}]
$$

$$
\begin{aligned}
&+\epsilon_1[s_{\mathrm{NX,NY-1,NZ}} + t_{\mathrm{NX-1,NY,NZ}} + w_{\mathrm{NX-1,NY,NZ}} + \\
&s_{\mathrm{NX,NY,NZ-1}} + t_{\mathrm{NX,NY,NZ-1}} + w_{\mathrm{NX,NY-1,NZ}}] \\
&+\epsilon_2[s_{\mathrm{NX,NY-2,NZ}} + t_{\mathrm{NX-2,NY,NZ}} + w_{\mathrm{NX-2,NY,NZ}} + \\
&s_{\mathrm{NX,NY,NZ-2}} + t_{\mathrm{NX,NY,NZ-2}} + w_{\mathrm{NX,NY-2,NZ}}] \\
&+\epsilon_3[s_{\mathrm{NX,NY-3,NZ}} + t_{\mathrm{NX-3,NY,NZ}} + w_{\mathrm{NX-3,NY,NZ}} + \\
&s_{\mathrm{NX,NY,NZ-3}} + t_{\mathrm{NX,NY,NZ-3}} + w_{\mathrm{NX,NY-3,NZ}}] \\
&+\epsilon_4[s_{\mathrm{NX,NY-1,NZ-1}} + t_{\mathrm{NX-1,NY,NZ-1}} + w_{\mathrm{NX-1,NY-1,NZ}}] \qquad (8) \\
&+\epsilon_5[s_{\mathrm{NX,NY-1,NZ-2}} + t_{\mathrm{NX-1,NY,NZ-2}} + w_{\mathrm{NX-1,NY-2,NZ}} + \\
&s_{\mathrm{NX,NY-2,NZ-1}} + t_{\mathrm{NX-2,NY,NZ-1}} + w_{\mathrm{NX-2,NY-1,NZ}}] \\
&+\epsilon_6[s_{\mathrm{NX,NY-2,NZ-2}} + t_{\mathrm{NX-2,NY,NZ-2}} + w_{\mathrm{NX-2,NY-2,NZ}}]
\end{aligned}
$$

Discretizations for all other boundary points may be obtained from these by symmetry. Values of coefficients yielding $\mathcal{O}(h^4)$ and $\mathcal{O}(h^2)$ truncation error are listed in Table 1 as Method A and Method B, respectively.

Since the operators L_{ijk} and I_{ijk} are obtained in each case from (3–4) by reflection symmetry we have that all difference equations are of positive type, which implies that the global error is the same as the truncation error in each case. Thus, methods A and B of Table 1 have order of accuracy $\mathcal{O}(h^4)$ and $\mathcal{O}(h^2)$ respectively.

2.3 Matrix Formulation

After incorporating Dirichlet and periodic boundary conditions in the usual way and using the natural ordering of grid points one obtains a linear system $Mu = g$. The nonzero elements of given a row in M correspond to the coefficients of the associated functional L_{ijk}. The right-hand side contains the contributions from the operators I_{ijk}, J_{ijk}, as well as terms eliminated from L_{ijk} to account for Dirichlet boundary data.

To describe the structure of the matrix M we first develop a notation for a certain class of nearly-tridiagonal matrices. Define

$$
T(a,b;\mu,\nu,\rho;n) = \begin{bmatrix}
a & \mu b & & & & & \rho b \\
b & a & b & & & & \\
 & b & a & b & & & \\
 & & \ddots & \ddots & \ddots & & \\
 & & & \ddots & \ddots & \ddots & \\
 & & & & b & a & b \\
\rho b & & & & & \nu b & a
\end{bmatrix}_{n\times n}
$$

In tensor-product notation [15] this may be written as

$$
T(a,b;\mu,\nu,\rho;n) = I_n \otimes a + J_n(\mu,\nu,\rho) \otimes b
$$

where

$$J_n(\mu,\nu,\rho) = \begin{bmatrix} 0 & \mu & & & & & \rho \\ 1 & 0 & 1 & & & & \\ & 1 & 0 & 1 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & \cdot \\ & & & & 1 & 0 & 1 \\ \rho & & & & & \nu & 0 \end{bmatrix}_{n\times n}$$

and I_n is the identity matrix of order n. In terms of these we define matrices A, B, C, J_y, and J_z by

$$\begin{aligned} A &= T(a,b;\mu_x,\nu_x,\rho_x;n) \\ B &= T(b,c;\mu_x,\nu_x,\rho_x;n) \\ C &= T(c,d;\mu_x,\nu_x,\rho_x;n) \\ J_y &= J_m(\mu_y,\nu_y,\rho_y) \\ J_z &= J_l(\mu_z,\nu_z,\rho_z) \end{aligned}$$

The scalars μ_x, ν_x, ρ_x, and n depend upon the boundary conditions in x, μ_y, ν_y, ρ_y, and m depend upon the boundary conditions in y, and μ_z, ν_z, ρ_z, and l depend upon the boundary conditions in z. Table 2 gives this dependence explicity. The scalars a, b, c, and d are from (3–4). In terms of these quantities, the matrix M may finally be written as

$$M = I_l \otimes (I_m \otimes A + J_y \otimes B) + J_z \otimes (I_m \otimes B + J_y \otimes C) \tag{9}$$

2.4 The Two-Dimensional Case

One may obtain a related discretization for the Helmholtz equation in two dimensions by considering the application of the three-dimensional discretization to a problem whose solution u is independent of z. When this is done one obtains the fourth-order accurate discretization displayed in [4]. The resulting matrix problem $Mu = g$ has a matrix of the somewhat simplified form

$$M = I_m \otimes A + J_y \otimes B \tag{10}$$

where A and B have the same form as the matrices A and B described in Section 2.3 above.

3 The Fourier Algorithm

The solution of linear algebraic systems involving matrices of the form (9) or (10) are the prototypical discrete problems for which fast-direct methods have been developed. Our software is based on the fact that high-order-accurate discretizations may be obtained by a proper choice of a, b, c, and d, and a perturbation of the right hand side. We use the Fourier method since software for the discrete Fourier transform is readily available, and the algorithm is adaptable for use on parallel or vector processors [26, 24].

We describe the Fourier algorithm only briefly; details may be found in [6] or [25]. We first consider the two-dimensional problem $Mu = g$ with M defined as in (10); this yields the matrix problem

$$\begin{bmatrix} A & \mu_x B & & & & & \rho_x B \\ B & A & B & & & & \\ & B & A & B & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & \cdot \\ & & & & B & A & B \\ \rho_x B & & & & & \nu_x B & A \end{bmatrix} \begin{bmatrix} u_{1\bullet} \\ u_{2\bullet} \\ u_{3\bullet} \\ \cdot \\ \cdot \\ \cdot \\ u_{m-1,\bullet} \\ u_{m\bullet} \end{bmatrix} = \begin{bmatrix} g_{1\bullet} \\ g_{2\bullet} \\ g_{3\bullet} \\ \cdot \\ \cdot \\ \cdot \\ g_{m-1,\bullet} \\ g_{m\bullet} \end{bmatrix}, \qquad (11)$$

Note that we use double indices to denote the elements of the unknown vector u and the right side vector g; u_{ij} denotes the value of u at the $(i,j)^{\text{th}}$ grid point, $u_{\bullet j}$ denotes the j^{th} column of u, and $u_{i\bullet}$ denotes the i^{th} row of u.

The Fourier method relies on the fact that the matrices A and B have a common set of linearly independent eigenvectors of a very special form; see Table 3. Let Q be a matrix of these eigenvectors, and let $\omega_i(A)$ denote the eigenvalue corresponding to the i^{th} eigenvector of A. If we premultiply each block equation in (11) by Q^{-1} and scale each block of unknowns by Q^{-1}, the system decouples to the solution of n tridiagonal systems of size m by m. The two-dimensional algorithm may be summarized as follows.

Algorithm: 2D Decomposition

1. (Transform x variable) Compute $g_{\bullet j} \leftarrow Q^{-1} g_{\bullet j}$ for $j = 1, \ldots, m$
2. (Solve n 1D problems in y) Solve $T_i u_{i\bullet} = g_{i\bullet}$ for $i = 1, \ldots, n$
3. (Back transform x variable) Compute $u_{\bullet j} \leftarrow Q u_{\bullet j}$ for $j = 1, \ldots, m$

T_i is the tridiagonal matrix $T(\omega_i(A), \omega_i(B); \mu_y, \nu_y, \rho_y; m)$. Multiplication by Q^{-1} is equivalent to performing a discrete Fourier transform and hence may be computed using FFT techniques. As a result, for NX-1 a power of two, the algorithm requires only $\mathcal{O}(2mn \log_2 n)$ floating point operations on a scalar computer. Note that no matrix need be stored, and the arrays u and g may coincide in memory, so that the algorithm is quite storage efficient.

In the three-dimensional case, we consider a problem $Mu = g$ with M defined by (9) and triply subscripted u and g. In this case, after premultiplying each block equation by Q^{-1} and scaling each block of unknowns by Q^{-1}, one obtains a set of decoupled block tridiagonal systems. The three-dimensional algorithm may be summarized as follows.

Algorithm: 3D Decomposition

1. (Transform x variable) Compute $g_{\bullet jk} \leftarrow Q^{-1} g_{\bullet jk}$ for $j = 1, \ldots, m$, $k = 1, \ldots, l$
2. (Solve n 2D problems in yz) Solve $M_i u_{i\bullet\bullet} = g_{i\bullet\bullet}$ for $i = 1, \ldots, n$
3. (Back transform x variable) Compute $u_{\bullet jk} \leftarrow Q u_{\bullet jk}$ for $j = 1, \ldots, m$, $k = 1, \ldots, l$

The matrices M_i are given by

$$M_i = I_l \otimes A_i + J_z \otimes B_i$$

where $A_i = T(\omega_i(A), \omega_i(B); \mu_y, \nu_y, \rho_y; m)$ and $B_i = T(\omega_i(B), \omega_i(C); \mu_y, \nu_y, \rho_y; m)$. The n subproblems in Step 2 may be solved by the two-dimensional algorithm. Thus, in the three-dimensional case, when NX-1 and NY-1 are powers of two, the Fourier method requires only $\mathcal{O}(2nml(\log_2 n + \log_2 m))$ floating point operations on a scalar computer.

4 Computational Examples

The algorithms described in sections 2 and 3 are the basis of a suite of Fortran subprograms for the solution of the Helmholtz equation on rectangular two or three dimensional domains with any combination of Dirichlet, Neumann, or periodic boundary conditions. This subprogram package, called HFFT, is described in [5]. It is also available as part of the ELLPACK system [23]. It's two primary entry points are HFFT2 and HFFT3, for two-dimensional and three-dimensional problems, respectively. In each case the user may select either a fourth-order accurate or a second-order accurate discretization; the discretizations used are those discussed in section 2.

We illustrate the performance of the HODIE-based algorithms by comparing the software in HFFT with other widely-available software for this problem. HFFT2 is contrasted with the subprogram SEPX4 [1] which uses standard second order differences, solving the discrete problem using recursive cyclic reduction, an algorithm of the same complexity as the Fourier method. Optionally, SEPX4 will produce fourth order accuracy using deferred corrections; this requires a second fast solve using a modified right hand side. HFFT3 is compared with HW3CRT [1] which implements the Fourier method for the standard five-point second-order accurate discretization. In summary, the programs we consider are

Two-Dimensional Software

HFFT2(2) HFFT2 with second-order accurate compact nine-point discretization. Fourier method.

HFFT2(4) HFFT2 with fourth-order accurate compact nine-point discretization. Fourier method.

SEPX4(2) SEPX4 with second-order accurate five-point discretization. Recursive cyclic reduction.

SEPX4(4) SEPX4 with second-order accurate five-point discretization. Fourth-order accuracy obtained by deferred corrections. Recursive cyclic reduction.

Three-Dimensional Software

HFFT3(2) HFFT3 with second-order accurate compact 19-point discretization. Fourier method.

HFFT3(4) HFFT3 with fourth-order accurate compact 19-point discretization. Fourier method.

HW3CRT Second-order accurate seven-point discretization. Fourier method.

All our tests are run in single precision on a Cyber 180/855 computer (NOS 2.3, FTN compiler, version 5.1, OPT=2). Timing data includes time to evaluate the forcing function g and the boundary conditions. Since HFFT2 requires only one fast solve we expect its basic execution time to be roughly the same as SEPX4(2) which, in turn, should be half that of SEPX4(4). Similarly, we expect execution times for HW3CRT, HFFT3(2), and HFFT3(4) to be roughly the same. These are verified in Table 4 where we present timings for the homogeneous Dirichlet problem for Laplace's equation on a sequence of square grids (a problem for which the cost of function evaluations is minimum).

Figure 1 displays the results obtained by running the two-dimensional software on six additional test problems. These plots display maximum error (scaled by maximum value of u in the domain) versus computing time, plotted on a log-log scale. The data points correspond to grids of 9×9, 17×17, 33×33, 65×65, and 129×129. These are

Two-Dimensional Test Problems

A2 $Lu = \Delta u$, u prescribed (=0) on $x = 0$, $x = 1$, $y = 0$, $y = 1$, $u = 3xy(1-x)(1-y)\exp x + y$. Solution is entire slowly varying.

B2 $Lu = \Delta u - 5u$, u_n prescribed on $x = 0$, $x = 1$, $y = 0$, and u prescribed (=0) on $y = 1$; solution same as in problem A2.

C2 $Lu = \Delta u - 20u$, u periodic in x and y, $u = \cos 4\pi y + \sin 4\pi(x - y)$. Solution is entire and slowly varying.

D2 Same as problem C2 with u prescribed on $y = 0$, u_n prescribed on $y = 1$, and u periodic in x.

E2 $Lu = \Delta u$, u prescribed on $x = 1, y = 1$, u_n prescribed on $x = 0, y = 0$, $u = (xy)^{5/2}$. Solution has singular third derivatives along $x = 0$ and $y = 0$.

F2 $Lu = \Delta u$, u prescribed on $x = 0, 1$, $y = 0, 1$, $u = f(x)f(y)$, $f(x) = (x^{3/4} - x)$. Solution has singular first derivatives along $x = 0$ and $y = 0$.

Since the solutions to problems A2, B2, C2, and D2 are entire the observed convergence rate of each method is as expected. The two second-order methods, HFFT2(2) and SEPX4(2) have nearly the same behavior for all these problems. HFFT2(4) shows no clear advantage over SEPX4(4) for problem A2 (a Dirichlet problem for Poisson's equation), although, for a given grid, it runs twice as fast. However, when normal derivative or periodic boundary conditions are present (problems B2, C2, D2), the error produced by HFFT2(4) is much smaller (10 times smaller in problem B2, 45 times smaller in problem D2). When combined with its faster execution times, these problems clearly show the advantage of HFFT2(4). For problems E2 and F2, which have solutions with singular low-order derivatives, the observed convergence rates were 1.5 and 0.7 for all methods. In contrast to SPEX4(4) which shows no advantage over SEPX4(2) for these problems, HFFT2(4) again produces a smaller error, although this effect is less prominent in problem F2.

Figure 2 displays the results obtained by running the three-dimensional software on six similar problems. These plots display maximum error (scaled by maximum value of u in the domain) versus computing time, plotted on a log-log scale. The data points correspond to grids of $9 \times 9 \times 9$, $13 \times 13 \times 13$, $17 \times 17 \times 17$, $21 \times 21 \times 21$, $25 \times 25 \times 25$, and $29 \times 29 \times 29$. These are

Three-Dimensional Test Problems

A3 $Lu = \Delta u$, u prescribed (=0) on $x = 0$, $x = 1$, $y = 0$, $y = 1$, $z = 0$, $z = 1$, $u = xyz(1-x)(1-y)(1-z)exp(x+y+z)$. Solution is entire slowly varying.

B3 $Lu = \Delta u - 5u$, u_n prescribed on $x = 0$, $x = 1$, $y = 0$, $z = 0$, and u prescribed (=0) on $y = 1$, $z = 1$; solution same as in problem A3.

C3 $Lu = \Delta u - 20u$, u periodic in x, y, and z, $u = \cos 4\pi y + \cos 4\pi z + \sin 4\pi(x - y)$. Solution is entire and slowly varying.

D3 Same as problem C3 with u prescribed on $y = 0$, $z = 0$, u_n prescribed on $y = 1$, $z = 1$, and u periodic in x.

E3 $Lu = \Delta u$, u prescribed on $x = 1$, $y = 1$, $z = 0$, $z = 1$, and u_n prescribed on $x = 0$, $y = 0$, $u = (xyz)^{5/2}$. Solution has singular third derivatives along $x = 0$ and $y = 0$.

F3 $Lu = \Delta u$, u prescribed on $x = 0, 1$, $y = 0, 1$, $z = 0, 1$, $u = f(x)f(y)f(z)$, $f(x) = (x^{3/4} - x)$. Solution has singular first derivatives along $x = 0$ and $y = 0$.

The relative performance of these programs is seen to be similar to the two-dimensional case.

References

[1] J. Adams, P. N. Swarztrauber, and R. A. Sweet. FISHPAK, a package of Fortran subprograms for the solution of separable elliptic partial differential equations. Version 3.1. NCAR Program Library, National Center for Atmospheric Research, P. O. Box 3000, Boulder, CO 80307, 1981.

[2] R. F. Boisvert. Families of high order accurate discretizations of some elliptic problems. *SIAM J. Sci. Stat. Comp.*, 2:268–284, 1981.

[3] R. F. Boisvert. High order compact difference formulas for elliptic problems with mixed boundary conditions. In R. Vichnevetsky and R. S. Stepleman, editors, *Advances in Computer Methods for Partial Differential Equations IV*, pages 193–199. IMACS, Rutgers University, New Brunswick, NJ, 1981.

[4] R. F. Boisvert. A fourth-order accurate fast direct method for the Helmholtz equation. In G. Birkhoff and A. Schoenstadt, editors, *Elliptic Problem Solvers II*, pages 35–44. Academic Press, Orlando, 1984.

[5] R. F. Boisvert. Algorithm 651: HFFT — high-order fast-direct solution of the Helmholtz equation. *ACM Trans. Math. Softw.*, 13:235–249, 1987.

[6] B. L. Buzbee, G. H. Golub, and C. W. Nielson. On direct methods for solving Poisson's equations. *SIAM J. Num. Anal.*, 7:627–655, 1970.

[7] M. Ciment, S. H. Leventhal, and B. C. Weinberg. The operator compact implicit method for parabolic equations. *J. Comp. Phys.*, 28:135–166, 1978.

[8] L. Collatz. *The Numerical Treatment of Differential Equations.* Springer–Verlag, Berlin, 1960.

[9] W. R. Dyksen. The tensor product generalized ADI method for elliptic problems. CSD-TR 493, Purdue University, Computer Sciences Dept., W. Lafayette, IN, 1984.

[10] E. N. Houstis and T. S. Papatheodorou. High-order fast elliptic equation solver. *ACM Trans. Math. Softw.*, 5:431–441, 1979.

[11] L. Kaufman and D. D. Warner. High order, fast-direct methods for separable elliptic equations. *SIAM J. Num. Anal.*, 21:672–694, 1984.

[12] R. E. Lynch. $o(h^6)$ accurate finite difference approximation to solutions of the Poisson equation in three variables. CSD-TR 221, Purdue University, Computer Sciences Dept., W. Lafayette, IN, 1977.

[13] R. E. Lynch. $O(h^4)$ and $O(h^6)$ finite difference approximations to the Helmholtz equation in n-dimensions. In R. Vichnevetsky and R. S. Stepleman, editors, *Advances in Computer Methods for Partial Differential Equations V*, pages 199–202. IMACS, Rutgers University, New Brunswick, NJ, 1984.

[14] R. E. Lynch and J. R. Rice. High accuracy finite difference approximations to solutions of elliptic partial differential equations. *Proc. Nat. Acad. Sci.*, 75:2541–2544, 1978.

[15] R. E. Lynch, J. R. Rice, and D. H. Thomas. Tensor product analysis of partial differential equations. *Bull. Am. Math. Soc.*, 70:378–384, 1964.

[16] R. Manohar and J. W. Stephenson. High order difference schemes for linear partial differential equations. *SIAM J. Sci. Stat. Comp.*, 5:69–77, 1984.

[17] A. Mayo. Fast high order accurate solution of Laplace's equation on irregular regions. *SIAM J. Sci. Stat. Comp.*, 6:144–157, 1985.

[18] P. Mercier and M. Deville. A multidimensional compact higher-order scheme for 3-d Poisson's equation. *J. Comp. Phys.*, 39:443–455, 1981.

[19] V. Pereyra. On improving the approximate solution of a functional equation by deferred corrections. *Numer. Math.*, 8:376–391, 1966.

[20] V. Pereyra, W. Proskurowski, and O. Widlund. High order fast Laplace solvers for the Dirichlet problem on general regions. *Math. Comp.*, 31:1–16, 1977.

[21] W. Proskurowski. Numerical solution of Helmholtz's equation by implicit capacitance matrix methods. *ACM Trans. Math. Softw.*, 5:36–49, 1979.

[22] W. Proskurowski. Algorithm 593. A package for the Helmholtz equation in nonrectangular planar regions. *ACM Trans. Math. Softw.*, 9:117–124, 1983.

[23] J. R. Rice and R. F. Boisvert. *Solving Elliptic Problems Using ELLPACK*, volume 2 of *Springer Series in Computational Mathematics*. Springer–Verlag, New York, 1985.

[24] M. H. Schultz. Solving elliptic problems on an array processor system. In G. Birkhoff and A. Schoenstadt, editors, *Elliptic Problem Solvers II*, pages 77–92. Academic Press, Orlando, FL, 1984.

[25] P. N. Swarztrauber. The methods of cyclic reduction, Fourier analysis, and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle. *SIAM Rev.*, 19:490–501, 1977.

[26] P. N. Swarztrauber. Vectorizing the FFTs. In G. Rodrigue, editor, *Parallel Computation*, pages 51–84. Academic Press, New York, 1982.

Table 1: Coefficients of finite difference equations

Coefficient	Method A	Method B
a	$-24+5\Lambda-\Lambda^2/4$	$-24+6\Lambda$
b	$2-\Lambda/24$	2
c	$1+5\Lambda/48$	1
d	0	0
β_0	$2-\Lambda/4$	6
β_1	$1/2$	0
β_2	$\Lambda/48$	0
γ_0	$-12+11\Lambda/12$	–12
γ_1	$\Lambda/12$	0
δ_0	$-(1104-125\Lambda)/144$	–10
δ_1	$-6+\Lambda/4$	–2
δ_2	$2-\Lambda/48$	0
δ_3	$-(24+\Lambda)/72$	0
δ_4	$-(24-\Lambda)/48$	0
δ_5	$(8+\Lambda)/16$	0
ϵ_0	0	–8
ϵ_1	$-(2328-245\Lambda)/144$	–2
ϵ_2	$(141-14\Lambda)/18$	0
ϵ_3	$-(24+\Lambda)/72$	0
ϵ_4	$(840-103\Lambda)/36$	0
ϵ_5	$-(1752-221\Lambda)/144$	0
ϵ_6	$(57-7\Lambda)/9$	0

Table 2: Dependence of μ, ν, ρ, n on boundary conditions

Boundary Conditions					
Left	Right	μ	ν	ρ	$n^\dagger$
Dirichlet	Dirichlet	1	1	0	N–2
Dirichlet	Neumann	1	2	0	N
Neumann	Dirichlet	2	1	0	N–1
Neumann	Neumann	2	2	0	N
Periodic	Periodic	1	1	1	N–1

† N is the number of grid lines

Table 3: Eigenvalues/eigenvectors of $T(a, b; \mu, \nu, \rho; n)^{\dagger}$

Case : Nonperiodic ($\rho = 0$)		
	$\nu = 1$	$\nu = 2$
$\mu = 1$	$\sigma_j = \cos j\pi/(n+1)$ $Q_{ij} = \sin ij\pi/(n+1)$	$\sigma_j = \cos(2j-1)\pi/(2n)$ $Q_{ij} = \sin i(2j-1)\pi/(2n)$
$\mu = 2$	$\sigma_j = \cos(2j-1)\pi/(2n)$ $Q_{ij} = \cos(i-1)(2j-1)\pi/(2n)$	$\sigma_j = \cos(j-1)\pi/(n-1)$ $Q_{ij} = \cos(i-1)(j-1)\pi/(n-1)$
Case : Periodic ($\mu = \nu = \rho = 1$)		
$\sigma_j = \cos 2k\pi/n$ where $k = \lfloor j/2 \rfloor$ $Q_{ij} = \cos j(i-1)\pi/n$, j even; $-\sin(j-1)(i-1)\pi/n$, j odd		

† Eigenvalues are $a + 2b\sigma_j,\ j = 1, \ldots, n$

Table 4: Raw Timing Data †

2D	Second Order		Fourth Order	
n	SEPX4	HFFT2	SEPX4	HFFT2
8	0.01	0.01	0.01	0.01
16	0.01	0.02	0.03	0.02
32	0.05	0.05	0.12	0.06
64	0.20	0.18	0.51	0.22
128	0.84	0.69	2.09	0.82
3D	Second Order		Fourth Order	
n	HW3CRT	HFFT3		HFFT3
8	0.10	0.11		0.12
12	0.25	0.27		0.30
16	0.51	0.52		0.59
20	0.91	0.94		1.08
24	1.56	1.50		1.76
28	2.90	2.75		3.06

† in seconds on Cyber 180/855

Figure Captions

1. Maximum error at grid points versus computing time for six two-dimensional test problems. Each line corresponds to a different software module: $\Box$ denotes SEPX4(2), $\Box$ denotes SEPX4(4), $\circ$ denotes HFFT2(2), and $\bullet$ denotes HFFT2(4). The data points on each line correspond to grids of size 9×9, 17×17, 33×33, 65×65, and 129×129.

2. Maximum error at grid points versus computing time for six three-dimensional test problems. Each line corresponds to a different software module: $\Box$ denotes HW3CRT, $\circ$ denotes HFFT3(2), and $\bullet$ denotes HFFT3(4). The data points correspond to grids of size $9 \times 9 \times 9$, $17 \times 17 \times 17$, $21 \times 21 \times 21$, $25 \times 25 \times 25$, and $29 \times 29 \times 29$.