The Direct Solution of the Biharmonic Equation on Rectangular Regions and the Poisson Equation on Irregular Regions

B. L. Buzbee, Fred W. Dorr

SIAM Journal on Numerical Analysis, Volume 11, Issue 4 (Sep., 1974), 753-763.

Stable URL:
http://links.jstor.org/sici?sici=0036-1429%28197409%2911%3A4%3C753%3ATDSOTB%3E2.0.CO%3B2-J

Your use of the JSTOR archive indicates your acceptance of JSTOR's Terms and Conditions of Use, available at http://www.jstor.org/about/terms.html. JSTOR's Terms and Conditions of Use provides, in part, that unless you have obtained prior permission, you may not download an entire issue of a journal or multiple copies of articles, and you may use content in the JSTOR archive only for your personal, non-commercial use.

Each copy of any part of a JSTOR transmission must contain the same copyright notice that appears on the screen or printed page of such transmission.

SIAM Journal on Numerical Analysis is published by Society for Industrial and Applied Mathematics. Please contact the publisher for further permissions regarding the use of this work. Publisher contact information may be obtained at http://www.jstor.org/journals/siam.html.

SIAM Journal on Numerical Analysis
©1974 Society for Industrial and Applied Mathematics

JSTOR and the JSTOR logo are trademarks of JSTOR, and are Registered in the U.S. Patent and Trademark Office. For more information on JSTOR contact jstor-info@umich.edu.

©2002 JSTOR

http://www.jstor.org/
Sat Aug 24 14:56:04 2002

SIAM J. NUMER. ANAL.
Vol. 11, No. 4, September 1974

THE DIRECT SOLUTION OF THE BIHARMONIC EQUATION ON RECTANGULAR REGIONS AND THE POISSON EQUATION ON IRREGULAR REGIONS*

B. L. BUZBEE AND FRED W. DORR†

Abstract. The discrete biharmonic equation on a rectangular region and the discrete Poisson equation on an irregular region can be treated as modifications to matrix problems with very special structure. We show how to use the direct method of matrix decomposition to formulate an effective numerical algorithm for these problems. For typical applications the operation count is $O(N^3)$ for an $N \times N$ grid. Numerical comparisons with other techniques are included.

1. Introduction. *Modification methods* obtain the solution of a matrix equation $A\mathbf{u} = \mathbf{v}$ by solving another equation $B\mathbf{u} = \mathbf{w}$. Usually B differs only slightly from A, and B is chosen so that the equation $B\mathbf{u} = \mathbf{w}$ can be solved efficiently. The modification $\mathbf{w}$ of the right-hand side $\mathbf{v}$ must be computed, and in order to determine $\mathbf{w}$ we need to know certain elements of B^{-1}. Since these elements are independent of $\mathbf{v}$, this computation can be done as a *preprocessing* phase. Using the results of the preprocessing, the solution of a particular equation $A\mathbf{u} = \mathbf{v}$ can then be done very efficiently.

The capacitance matrix approach has been used by several authors to solve problems that are modifications of elliptic difference equations [5], [16], [26], [27], [33]. In [7] we described this technique in detail and indicated how it could be applied to the solution of the discrete Poisson equation on irregular regions. A similar approach was used by Golub [17] in deriving an algorithm for the solution of the discrete biharmonic equation on rectangular regions. In both cases, the algorithms were presented in a general form and without detailed consideration of any specific method for the solution of the modified matrix problem. In this paper, we show that the direct method of matrix decomposition can be used effectively in these algorithms. Matrix decomposition has been used in Poisson solvers by Christiansen and Hockney [9], Hockney [25]–[27] and Hughes [29], and the method has been described for more general problems in [8].

Efficient methods for the solution of the discrete biharmonic equation on rectangular regions have recently received a great deal of attention. The direct method described by Golub [17] has been examined experimentally by Walker [36] and Ehrlich [14]. Application of block Gaussian elimination to this problem was described in Bauer and Reiss [3] and Angel and Bellman [1]. Iterative methods that treat the biharmonic equation as a coupled pair of Poisson equations have been examined by Axelsson [2], Ehrlich [11]–[14], Greenspan and Schultz [18], Gupta [19]–[21], Gupta and Manohar [22], McLaurin [30], and Smith [31]–[32]. Iterative methods applied directly to the biharmonic difference equations have been investigated by Fairweather, Gourlay and Mitchell [15] and Hadjidimos

* Received by the editors May 11, 1973, and in revised form August 27, 1973.

† Los Alamos Scientific Laboratory of the University of California, Los Alamos, New Mexico 87544. This research was supported by the U.S. Atomic Energy Commission under Contract W-7405-ENG-36.

[23]–[24]. The direct method which we shall describe requires $O(N^3)$ operations and $3N^2$ storage locations to solve the discrete biharmonic equation on an $N \times N$ grid.

For the discrete Poisson equation on an irregular region, we let p denote the number of modified equations (cf. [7]). We also let $\theta(N)$ denote the number of operations required to solve a discrete Poisson equation on a rectangular region. Neglecting lower order terms, $\theta(N) = c_1N^3$ for an arbitrary value of N and $\theta(N) \doteq c_2N^2 \log_2 N$ if N is restricted to be $2^k - 1$ for an integer k [10]. The direct application of the capacitance matrix approach described in [7] requires $p\theta(N) + O(p^3)$ operations for preprocessing and $2\theta(N) + O(p^2)$ operations for each particular solution. The method which we shall describe requires $O(pN^2 + p^3)$ operations for preprocessing and $2\theta(N) + O(p^2)$ operations for each particular solution. The storage requirements are the same for both direct methods.

2. Algorithm for the biharmonic equation. Consider the biharmonic boundary value problem

$$\begin{aligned} \Delta^2 u(x,y) &= f(x,y) \quad \text{in } R,\\ u(x,y) &= g(x,y) \quad \text{on } \partial R,\\ \frac{\partial u(x,y)}{\partial n} &= h(x,y) \quad \text{on } \partial R, \end{aligned}$$

where $R = (a,b) \times (c,d)$ is a rectangle and $\partial u(x,y)/\partial n$ is the outward normal derivative of $u(x,y)$ on ∂R. Imposing a uniform mesh in both directions, we define

$$\Delta x = \frac{b-a}{M+1}, \qquad x_i = a + i\Delta x,$$
$$\Delta y = \frac{d-c}{N+1}, \qquad y_j = c + j\Delta y,$$

and we seek an approximation $u_{ij} \cong u(x_i, y_j)$. Ordering the unknowns by vertical lines, we let

$$\mathbf{u}_i = \begin{bmatrix} u_{i1} \\ u_{i2} \\ \cdot \\ \cdot \\ \cdot \\ u_{iN} \end{bmatrix}, \qquad U = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{u}_M \end{bmatrix}.$$

Using the standard thirteen-point difference approximation to the biharmonic operator [4], [38], [39], we must solve a linear system of equations $AU = V$ (we shall use capital letters such as U, V to denote vectors defined on the grid points (x_i, y_j)).

The matrix A can be written as $A = B + D$, where

$$B = \begin{bmatrix} (P^2 + 3I) & 2P & I & & & 0 \\ 2P & (P^2 + 2I) & 2P & I & & \\ I & 2P & (P^2 + 2I) & 2P & I & \\ & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & & I & 2P & (P^2 + 2I) & 2P \\ & & & I & 2P & (P^2 + 3I) \end{bmatrix},$$

$$P = \begin{bmatrix} -2(\rho + 1) & \rho & & & \\ \rho & -2(\rho + 1) & \rho & & 0 \\ & \ddots & \ddots & \ddots & \\ 0 & & \rho & -2(\rho + 1) & \rho \\ & & & \rho & -2(\rho + 1) \end{bmatrix}, \quad \rho = \left(\frac{\Delta x}{\Delta y}\right)^2,$$

$$D = 2\rho^2 \begin{bmatrix} E & & & \\ & E & & 0 \\ & & \ddots & \\ 0 & & & E \\ & & & & E \end{bmatrix}$$

and

$$E = \begin{bmatrix} 1 & & & \\ & 0 & & 0 \\ & & \ddots & \\ 0 & & & 0 \\ & & & & 1 \end{bmatrix}.$$

Since

$$B = \begin{bmatrix} P & I & & & \\ I & P & I & & 0 \\ & \ddots & \ddots & \ddots & \\ 0 & & I & P & I \\ & & & I & P \end{bmatrix}^2 + \begin{bmatrix} 2I & & & \\ & 0 & & 0 \\ & & \ddots & \\ 0 & & & 0 \\ & & & & 2I \end{bmatrix},$$

both A and B are positive definite. The right-hand side V is $v_{ij} = (\Delta x)^4 f(x_i, y_j)$, with additional terms to include the boundary conditions.

We consider the matrix B as a modification of the matrix A and use the capacitance matrix approach to solve the modified matrix problem. Systems involving the matrix B are solved using matrix decomposition. In [17] Golub combined the Woodbury formula and matrix decomposition to derive a similar method that required $O(N^4)$ operations and $9N^2$ storage locations (cf. the implementation of a modification of the Golub algorithm described in [14]). We first review these basic concepts and then show how to efficiently apply them to this particular problem.

Following the capacitance matrix development in [7], we assume that we are given an $n \times n$ matrix A and an integer p with $1 \leqq p \leqq n$. We modify p rows of A to obtain the matrix B, and B is chosen so that we can efficiently solve $B\mathbf{u} = \mathbf{w}$. For convenience of exposition we assume that the first p rows of A are changed. This is seldom the case in practice, but we can accomplish the same result by using an *implicit indexing* scheme. Partition A in the form

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix},$$

where A_1 is a $p \times n$ matrix and A_2 is an $(n - p) \times n$ matrix. By assumption,

$$B = \begin{pmatrix} B_1 \\ A_2 \end{pmatrix},$$

where B_1 is a $p \times n$ matrix. The *preprocessing* stage of the algorithm requires the computation of the $p \times p$ *capacitance matrix*

$$C = A_1 B^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix},$$

where the identity matrix is $p \times p$. We shall also assume that the capacitance matrix is factored into an LU decomposition [37, pp. 93–110] as part of the preprocessing phase. To find the solution to a particular equation $A\mathbf{u} = \mathbf{v}$, we partition $\mathbf{v}$ in the same way as A and proceed in the following steps:

1. Compute $\bar{\mathbf{u}}_1 = A_1 B^{-1}\mathbf{v}$,
2. Solve $C\boldsymbol{\beta} = \mathbf{v}_1 - \bar{\mathbf{u}}_1$,
3. Solve $B\mathbf{u} = \mathbf{v} + \begin{pmatrix} \boldsymbol{\beta} \\ \mathbf{0} \end{pmatrix}$.

To solve a system $BU = W$ we use the method of matrix decomposition as given in [8]. In describing this algorithm it is useful to think of the block-vectors $U = (u_{ij})$ and $W = (w_{ij})$ as two-dimensional arrays. Thus $\mathbf{u}_i$ can also be thought of as the ith row of U. The orthogonal matrix $R = (r_{ij})$ of eigenvectors of P is given by

$$r_{ij} = \sqrt{\frac{2}{N+1}} \sin \frac{ij\pi}{N+1}, \qquad 1 \leqq i, j \leqq N,$$

and the associated eigenvalues are

$$\lambda_j = 2\left[\rho\left(\cos\frac{j\pi}{N+1} - 1\right) - 1\right].$$

The solution of $BU = W$ is obtained as follows:

1. Compute the vectors $\mathbf{w}_i' = R\mathbf{w}_i$. In practice, this means that we multiply each row of W by R.

2. Reorder the array $W' = (\mathbf{w}_i')$ by horizontal lines instead of vertical lines to generate an array $W'' = (\mathbf{w}_j'')$, and then solve the pentadiagonal systems

$$\begin{bmatrix} (\lambda_j^2 + 3) & 2\lambda_j & 1 & & & \\ 2\lambda_j & (\lambda_j^2 + 2) & 2\lambda_j & 1 & & 0 \\ 1 & 2\lambda_j & (\lambda_j^2 + 2) & 2\lambda_j & 1 & \\ & \ddots & \ddots & \ddots & \ddots & \ddots \\ & 0 & 1 & 2\lambda_j & (\lambda_j^2 + 2) & 2\lambda_j \\ & & & 1 & 2\lambda_j & (\lambda_j^2 + 3) \end{bmatrix} \mathbf{u}_j'' = \mathbf{w}_j''.$$

In practice, this means that we solve the systems using the columns of W for each right-hand side and then store each vector $\mathbf{u}_j''$ in the columns of U. Of course, U can overwrite W if this is desired.

3. Reorder the array $U'' = (\mathbf{u}_j'')$ by vertical lines instead of horizontal lines to generate an array $U' = (\mathbf{u}_i')$ and then set $\mathbf{u}_i = R\mathbf{u}_i'$. In practice, this means that we multiply each row of U by R.

Given these two basic methods, we now incorporate them into an efficient algorithm for the solution of the biharmonic equation.

Step 1. Preprocessing. The matrix B is obtained by modifying the equations of A corresponding to the mesh points $\{(x_i, y_j) | 1 \leqq i \leqq M, j = 1, N\}$; thus $p = 2M$. Since $A = B + D$, the capacitance matrix C is given by

$$C = I + 2\rho^2 \bar{B},$$

where $\bar{B}$ is the $p \times p$ minor of B^{-1} corresponding to the modified mesh points. Since B is positive definite, so is $\bar{B}$ and hence C. This means that we only have to store a triangle of C and we can use Cholesky decomposition [37, pp. 9–30] to factor C and solve the capacitance matrix equation.

To compute the kth column of $\bar{B} = (\bar{b}_{ik})$ we proceed as follows: (i) set $T = 0$ at all mesh points, (ii) determine the kth modified mesh point and set $T = 1$ at that point, and (iii) find the value of the solution of $BS = T$ at the ith modified mesh point and store the result in $\bar{b}_{ik}$. *Note that we only need p components of the vector S.*

There are symmetries in the biharmonic operator and these particular right-hand sides that can be used to advantage in the calculation of the capacitance matrix. Because the solution can be reflected about the lines $x = (a + b)/2$ and $y = (c + d)/2$, the solution for the modified mesh point (x_i, y_1) $(1 \leqq i \leqq [(M + 1)/2])$[1] also determines the solution corresponding to the modified mesh points (x_{M+1-i}, y_1), (x_i, y_N) and (x_{M+1-i}, y_N). Thus we only have to calculate $[(M + 1)/2]$ columns of C instead of all $2M$ columns.

The important part of the algorithm is the fact that we can determine each required column of C using $O(N^2)$ operations and $O(N)$ storage locations.[2] To

[1] $[(M + 1)/2]$ is the greatest integer less than or equal to $(M + 1)/2$.

[2] When discussing operations and storage, we shall assume that $M = N$.

show this, we examine the application of matrix decomposition to the calculation of a column of C. Because there is only one nonzero row of T and that row is a unit vector, the multiplication of each row of T by R is done implicitly. The pentadiagonal systems are positive definite, so to solve all of them requires $O(N^2)$ operations. Finally, instead of multiplying each row of S by R, we *accumulate only the* $2N$ *sums of those components that correspond to the modified mesh points*, which requires $O(N^2)$ operations.

After C is computed, we factor it using Cholesky decomposition. Thus the total cost of the preprocessing is $O(N^3)$ operations and $2N^2 + O(N)$ storage locations ($2N^2 + N$ for the capacitance matrix and $O(N)$ for temporary storage). We now proceed with the solution of a particular equation $AU = V$ using the capacitance matrix.

Step 2. Compute $A_1B^{-1}V$. This part of the algorithm proceeds as in the calculation of a column of $\bar{B}$ in Step 1. Since V may be full, the multiplication of each row of V by R requires $O(N^3)$ operations. The solution of the pentadiagonal systems requires $O(N^2)$ operations. As in Step 1, we only need p components of the solution so we accumulate only the $2N$ sums of those components that correspond to the modified mesh points. This requires $O(N^2)$ operations, so that the total cost of this step is $O(N^3)$ operations and $N^2 + O(N)$ additional storage locations (N^2 for the right-hand side V and $O(N)$ for temporary storage).

Step 3. Solve the capacitance matrix equation. Using the factored capacitance matrix, this step requires $O(N^2)$ operations and $O(N)$ additional storage locations.

Step 4. Solution of the modified matrix equation. We add the vector $\boldsymbol{\beta}$ determined in Step 3 to V at the modified mesh points. The final system is then solved using the full matrix decomposition algorithm. One effective method of implementation is to overwrite the solution U on C and leave V unchanged, which preserves the storage requirement at $3N^2$. If the capacitance matrix is to be used again, the solution U can be overwritten on V. In this case, it is most efficient to store half of the symmetric matrix R, thus increasing the storage to $3.5N^2$. In either case, the number of operations for this step is $O(N^3)$.

In solving nonlinear or time-dependent problems, we frequently have to solve a sequence of equations

$$(A + \alpha^2 I)U_n = V_n, \qquad n = 0, 1, 2, \cdots.$$

The algorithm given above can be applied directly to the matrix $A + \alpha^2 I$, since this only adds α^2 to the diagonal of each of the pentadiagonal systems. After the preprocessing has been done, the solution time for each new right-hand side is essentially the $O(N^3)$ operations required for Steps 2 and 4. If N is restricted to be $2^k - 1$ for an integer k, the fast Fourier transform could be used to reduce this time to $O(N^2 \log_2 N)$ operations.

3. Numerical results for the biharmonic equation. We have implemented this direct method for the biharmonic equation and tested it on a variety of problems. Tables 1 and 2 summarize some results for a case in which $\Delta x = \Delta y = h$ and the solution U is composed of random numbers uniformly distributed on the interval $[1, 2]$. The computer used was a CDC 7600 and all subroutines were written in FORTRAN. The right-hand side was determined by multiplying U by the matrix

TABLE 1
Solution of the biharmonic equation using FORTRAN *subroutines*

h	Method	Number of iterations	Maximum error	Computation time (sec.)	Scaled computation time
$\frac{1}{25}$	Direct 1	—	5.409(−11)	0.185	1.00
	Direct 2	—	5.409(−11)	0.104	0.56
	Ehrlich 1	21	2.073(−4)	2·701	14.60
$\frac{1}{50}$	Direct 1	—	6.697(−10)	1.370	1.00
	Direct 2	—	6.697(−10)	0.747	0.55
	Ehrlich 1	32	1.006(−4)	30.762	22.45
$\frac{1}{100}$	Direct 1	—	3.228(−8)	10.453	1.00
	Direct 2	—	3.228(−8)	5.655	0.54
	Ehrlich 1	51	2.811(−5)	377.903	36.15

TABLE 2
Solution of the biharmonic equation for $M = 2^k - 1$ using FORTRAN *subroutines*

h	Method	Number of iterations	Maximum error	Computation time (sec.)	Scaled computation time
$\frac{1}{32}$	Direct 1		1.461(−10)	0.376	1.00
	Direct 2		1.461(−10)	0.209	0.56
	Ehrlich 2	24	1.705(−4)	1.582	4.21
$\frac{1}{64}$	Direct 1	—	2.171(−9)	2.812	1.00
	Direct 2	—	2.171(−9)	1.530	0.54
	Ehrlich 2	39	4.970(−5)	11.073	3.94

A. This is an interesting example because it gives an indication of the round-off error that can be expected in this problem. For the case $M = 2^k - 1$, we could improve the speed of the direct methods by using the fast Fourier transform, but we have not done so.

The tables also include results for the semidirect method examined by Ehrlich [11], [14]. The abbreviations used for the methods are:

Direct 1—direct method,
Direct 2—direct method using precomputed and factored capacitance matrix,
Ehrlich 1—Ehrlich iteration using matrix decomposition [8] for the Poisson solver (valid for arbitrary M),
Ehrlich 2—Ehrlich iteration using odd–even reduction [6] for the Poisson solver (valid for $M = 2^k - 1$).

For the Ehrlich method, we use the iteration parameters described in [14] and terminate the iteration when

$$\max_{i,j} |u_{ij}^{(n)} - u_{ij}^{(n-1)}| \leqq 0.1(\Delta x)^2.$$

The initial guess is $u_{ij}^{(0)} \equiv 1.5$. Because the convergence criterion is a function of Δx, the number of operations required for the solution is $O(N^{7/2} \log_2 N)$ using matrix decomposition and $O(N^{5/2}(\log_2 N)^2)$ using odd–even reduction (cf. [14]). The storage is $4.5N^2$ locations because we have to retain the right-hand side V.

4. Numerical results for the Poisson equation. The techniques developed in § 2 can also be applied to modified problems such as those described in [7] for Poisson equations on irregular regions. The only significant difference is that we may not be able to take as much advantage of symmetries in the calculation of the capacitance matrix. If there are p modified equations, the number of operations is $O(pN^2 + p^3)$ for Step 1, $O(N^3)$ for Step 2, $O(p^2)$ for Step 3, and $O(N^3)$ for Step 4. As we remarked earlier, if $M = 2^k - 1$ the operation count for Steps 2 and 4 can be reduced to $O(N^2 \log_2 N)$.

To illustrate the application of this method, we have considered the example treated in [7]. The region is a square with sides of length 1 that has a symmetrically located square removed from its center. We consider two cases: for Region 1 the inner square has sides of length $\frac{1}{8}$, and for Region 2 the inner square has sides of length $\frac{1}{4}$. We solve the Poisson equation with Dirichlet boundary conditions for the function $u(x, y) = x^2 + y^2$. All of the computations were performed on a CDC 7600 computer.

TABLE 3
Solution of the Poisson equation using FORTRAN *subroutines*

Region	h	p	Method	Maximum error	Computation time (sec.)	Scaled computation time
1	$\frac{1}{32}$	16	SOR	1.566(−4)	0.267	1.28
			SLOR	1.436(−4)	0.322	1.54
			ADI	2.672(−5)	0.215	1.03
			Direct 1	4.370(−13)	0.557	2.67
			Direct 2	4.370(−13)	0.209	1.00
			Direct 3	4.370(−13)	0.066	0.32
	$\frac{1}{64}$	32	SOR	2.497(−5)	2.571	1.85
			SLOR	2.998(−5)	2.752	1.98
			ADI	1.950(−5)	1.151	0.83
			Direct 1	1.890(−12)	4.511	3.25
			Direct 2	1.890(−12)	1.389	1.00
			Direct 3	1.890(−12)	0.280	0.20
2	$\frac{1}{32}$	32	SOR	1.270(−4)	0.266	0.73
			SLOR	1.107(−4)	0.307	0.84
			ADI	2.627(−5)	0.171	0.47
			Direct 1	3.695(−13)	1.051	2.88
			Direct 2	3.695(−13)	0.365	1.00
			Direct 3	3.695(−13)	0.067	0.18
	$\frac{1}{64}$	64	SOR	1.664(−5)	2.536	0.96
			SLOR	3.467(−5)	2.585	0.98
			ADI	1.842(−5)	1.090	0.41
			Direct 1	1.535(−12)	8.760	3.31
			Direct 2	1.535(−12)	2.649	1.00
			Direct 3	1.535(−12)	0.284	0.11

The results listed in Table 3 are for a program written entirely in FORTRAN. The methods used are:

SOR —successive point overrelaxation [34, p. 58],
SLOR —successive line overrelaxation [34, p. 80],
ADI —Peaceman–Rachford alternating direction implicit iteration [35, Chap. 6],
Direct 1—direct method including preprocessing time (calculation of the capacitance matrix implemented as described in [7]),
Direct 2—direct method including preprocessing time (calculation of the capacitance matrix implemented as described in this paper),
Direct 3—direct method using precomputed and factored capacitance matrix.

The initial guess for the iterative procedures is identically zero, and the iterations are terminated when the maximum difference between iterates is less than $0.1(\Delta x)^2$. The iteration parameters used are those for the imbedding rectangle, and for ADI the parameters for cycles of length four are determined by the algorithm described in [35, Chap. 6]. The direct methods use odd–even reduction [6] for the Poisson solver, so we are taking advantage of the fact that $M = 2^k - 1$. The

TABLE 4
Solution of the Poisson equation using some assembly language subroutines

Region	h	p	Method	Maximum error	Computation time (sec.)	Scaled computation time
1	$\frac{1}{32}$	16	SOR	1.566(−4)	0.265	2.43
			SLOR	1.436(−4)	0.192	1.76
			ADI	2.672(−5)	0.117	1.07
			Direct 1	4.370(−13)	0.312	2.86
			Direct 2	4.370(−13)	0.109	1.00
			Direct 3	4.370(−13)	0.039	0.36
	$\frac{1}{64}$	32	SOR	2.497(−5)	2.554	3.83
			SLOR	2.998(−5)	1.594	2.39
			ADI	1.950(−5)	0.614	0.92
			Direct 1	1.890(−12)	2.271	3.40
			Direct 2	1.890(−12)	0.667	1.00
			Direct 3	1.890(−12)	0.151	0.23
2	$\frac{1}{32}$	32	SOR	1.270(−4)	0.265	1.42
			SLOR	1.107(−4)	0.185	0.99
			ADI	2.627(−5)	0.094	0.51
			Direct 1	3.695(−13)	0.588	3.16
			Direct 2	3.695(−13)	0.186	1.00
			Direct 3	3.695(−13)	0.040	0.22
	$\frac{1}{64}$	64	SOR	1.664(−5)	2.523	2.03
			SLOR	3.467(−5)	1.510	1.21
			ADI	1.842(−5)	0.575	0.46
			Direct 1	1.535(−12)	4.428	3.56
			Direct 2	1.535(−12)	1.244	1.00
			Direct 3	1.535(−12)	0.151	0.12

differences between Table 3 and Table 1 in [7] are due to: (i) different computers (CDC 7600 for Table 3 and CDC 6600 for [7]) and (ii) different programming languages (FORTRAN for Table 3 and some assembly language in [7]). In addition, in [7] we recomputed the right-hand side in each iteration of the iterative procedures but in this program we have saved it. This increases the storage requirement but reduces the execution time.

Because of the "instruction stack" on the CDC 7600 computer, it is sometimes possible to significantly accelerate numerical algorithms by programming short loops in assembly language. In Table 4 we give the results for running these examples with the vector addition and inner product routines and tridiagonal system solver written in assembly language. It is interesting to note that the qualitative as well as the quantitative aspects of the results have changed for this problem. Hockney [28] has reported some similar results comparing the same FORTRAN programs on different computers.

Acknowledgment. The algorithm for the biharmonic equation described in this paper is closely related to earlier work of Professor Gene Golub. We have benefited greatly from several discussions with Professor Golub, and he provided much of the inspiration and incentive for this work.

REFERENCES

[1] E. ANGEL AND R. BELLMAN, *Dynamic Programming and Partial Differential Equations*, Academic Press, New York, 1972.

[2] O. AXELSSON, *Notes on the numerical solution of the biharmonic equation*, to appear.

[3] L. BAUER AND E. L. REISS, *Block five diagonal matrices and the fast numerical solution of the biharmonic equation*, Math. Comp., 26 (1972), pp. 311–326.

[4] J. H. BRAMBLE, *A second order finite difference analog of the first biharmonic boundary value problem*, Numer. Math., 9 (1966), pp. 236–249.

[5] O. BUNEMAN, *Computer simulation of the satellite photosheath*, ESRIN Internal Note 92, European Space Research Institute, Rome, 1970.

[6] B. L. BUZBEE, *Truncated Buneman Poisson solver*, Program Library Subroutine D301, Los Alamos Scientific Laboratory, Los Alamos, N.M.

[7] B. L. BUZBEE, F. W. DORR, J. A. GEORGE AND G. H. GOLUB, *The direct solution of the discrete Poisson equation on irregular regions*, this Journal, 8 (1971), pp. 722–736.

[8] B. L. BUZBEE, G. H. GOLUB AND C. W. NIELSON, *On direct methods for solving Poisson's equations*, this Journal, 7 (1970), pp. 627–656.

[9] J. P. CHRISTIANSEN AND R. W. HOCKNEY, *DELSQPHI, a two-dimensional Poisson-solver program*, Computer Physics Communications, 2 (1971), pp. 139–155.

[10] F. W. DORR, *The direct solution of the discrete Poisson equation on a rectangle*, SIAM Rev., 12 (1970), pp. 248–263.

[11] L. W. EHRLICH, *Solving the biharmonic equation as coupled finite difference equations*, this Journal, 8 (1971), pp. 278–287.

[12] ———, *Point and block SOR applied to a coupled set of difference equations*, Rep. TG 1166, Applied Physics Laboratory, The Johns Hopkins University, Silver Spring, Md., 1971.

[13] ———, *Coupled harmonic equations, SOR, and Chebyshev acceleration*, Math. Comp., 26 (1972), pp. 335–343.

[14] ———, *Solving the biharmonic equation in a square: A direct versus a semidirect method*, Comm. ACM, 16 (1973), pp. 711–714.

[15] G. FAIRWEATHER, A. R. GOURLAY AND A. R. MITCHELL, *Some high accuracy difference schemes with a splitting operator for equations of parabolic and elliptic type*, Numer. Math., 10 (1967), pp. 56–66.

[16] J. A. GEORGE, *The use of direct methods for the solution of the discrete Poisson equation on nonrectangular regions*, Rep. STAN-CS-70-159, Computer Science Dept., Stanford Univ., Stanford, Calif., 1970.

[17] G. H. GOLUB, *An algorithm for the discrete biharmonic equation*, unpublished.

[18] D. GREENSPAN AND D. SCHULTZ, *Fast finite-difference solution of biharmonic problems*, Comm. ACM, 15 (1972), pp. 347–350.

[19] M. M. GUPTA, *Convergence and stability of finite difference schemes for some elliptic equations*, Doctoral thesis, Univ. of Saskatchewan, Saskatoon, 1971.

[20] ———, *Efficient difference schemes for biharmonic equation*, to appear.

[21] ———, *Numerical solution of a second biharmonic boundary value problem*, to appear.

[22] M. M. GUPTA AND R. MANOHAR, *On numerical solution of biharmonic equation by splitting*, Proc. Manitoba Conference on Numerical Mathematics, Univ. of Manitoba, Winnipeg, 1971, pp. 345–359.

[23] A. HADJIDIMOS, *On some high accuracy difference schemes for solving elliptic equations*, Numer. Math., 13 (1969), pp. 396–403.

[24] ———, *The numerical solution of a model problem biharmonic equation by using extrapolated alternating direction implicit methods*, Ibid., 17 (1971), pp. 301–317.

[25] R. W. HOCKNEY, *A fast direct solution of Poisson's equation using Fourier analysis*, J. Assoc. Comput. Mach., 12 (1965), pp. 95–113.

[26] ———, *The potential calculation and some applications*, Methods in Computational Physics, 9 (1970), pp. 135–211.

[27] ———, *POT4—A fast direct Poisson-solver for the rectangle allowing some mixed boundary conditions and internal electrodes*, to appear.

[28] ———, *Computers, compilers and Poisson-solvers*, to appear.

[29] M. H. HUGHES, *Solution of Poisson's equation in cylindrical coordinates*, Computer Physics Communications, 2 (1971), pp. 157–167.

[30] J. W. MCLAURIN, *A general coupled equation approach for solving the biharmonic boundary value problem*, to appear.

[31] J. SMITH, *The coupled equation approach to the numerical solution of the biharmonic equation by finite differences, I, II*, this Journal, 5 (1968), pp. 323–339; 7 (1970), pp. 104–111.

[32] ———, *On the approximate solution of the first boundary value problem for $\nabla^4 u = f$*, this Journal, 10 (1973), pp. 967–982.

[33] P. N. SWARZTRAUBER AND R. A. SWEET, *The direct solution of the discrete Poisson equation on a disk*, this Journal, 10 (1973), pp. 900–907.

[34] R. S. VARGA, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1962.

[35] E. L. WACHSPRESS, *Iterative Solution of Elliptic Systems*, Prentice-Hall, Englewood Cliffs, N.J., 1966.

[36] L. WALKER, *Approach to approximating biharmonic equation with finite differences*, M.S. thesis, Computer Science Dept., Stanford University, Stanford, Calif., 1970.

[37] J. H. WILKINSON AND C. REINSCH, *Linear Algebra*, Springer-Verlag, New York, 1971.

[38] M. ZLAMAL, *Asymptotic error estimates in solving elliptic equations of the fourth order by the method of finite differences*, this Journal, 2 (1965), pp. 337–344.

[39] ———, *Discretization and error estimates for elliptic boundary value problems of the fourth order*, this Journal, 4 (1967), pp. 626–639.