Journal of Computational Physics **171,** 79–94 (2001)
doi:10.1006/jcph.2001.6771, available online at http://www.idealibrary.com on IDEAL®

A High-Order Fast Direct Solver for Singular Poisson Equations

Yu Zhuang and Xian-He Sun

Department of Computer Science, Illinois Institute of Technology, Chicago, Illinois 60616

Received October 7, 1999; revised August 28, 2001

We present a fourth order numerical solution method for the singular Neumann boundary problem of Poisson equations. Such problems arise in the solution process of incompressible Navier–Stokes equations and in the time-harmonic wave propagation in the frequence space with the zero wavenumber. The equation is first discretized with a fourth order modified Collatz difference scheme, producing a singular discrete equation. Then an efficient singular value decomposition (SVD) method modified from a fast Poisson solver is employed to project the discrete singular equation into the orthogonal complement of the null space of the singular matrix. In the complement of the null space, the projected equation is uniquely solvable and its solution is proven to be a solution of the original singular discrete equation when the original equation has a solution. Analytical and experimental results show that this newly proposed singular equation solver is efficient while retaining the accuracy of the high order discretization.

Key Words: Poisson equation; Neumann boundary condition; SVD; fast Fourier transform (FFT); high order discretization.

1. INTRODUCTION

In this paper, we consider high-order solutions for the Poisson equation

$$p_{xx} + p_{yy} = r(x, y) \tag{1.1}$$

on a rectangular domain Ω with Neumann boundary condition

$$\frac{dp}{dn} = b(x, y) \quad \text{on } \partial\Omega, \tag{1.2}$$

where $\partial\Omega$ is the boundary of the rectangular domain Ω, n is the normal vector of the boundary of the computational domain, and $\frac{dp}{dn}$ indicates the derivative normal to the boundary. This type of problems arise in the solution of the incompressible Navier–Stokes equation [9, 25],

0021-9991/01 $35.00

and in acoustics, elastics, and electromagnetics in the time-harmonic wave propagation in the frequency space [1, 12] with the zero wavenumber.

It is well known [3, 11, 19] that the Neumann problem of the Poisson equation is singular. A singular linear system is solvable if and only if its right hand side is orthogonal to the null space of the singular matrix [3, 11]. The condition number[1] of a singular matrix is infinity. Thus in general, singular problems are difficult to solve efficiently and accurately. For the singular Neumann problem of the Poisson equation, however, there exist several efficient solvers, including Schumann and Sweet's cyclic reduction [19], Golub, Huang, Simon and Tang's generalized eigen-decomposition [11], Bialecki and Remington's Fast Poisson solver-based eigen-decomposition [3], and the Kaczmarz projection method that Tanabe introduced for singular systems [24].

The cyclic reduction is fast and has a complexity of $O(n^2 \log n)$ on a square grid of size $n \times n$. But Schumann and Sweet did not discuss how the singularity is treated and how cyclic reduction affects the accuracy of the solution of the discrete system. The generalized eigen-decomposition employed by Golub *et al.* has a complexity of $O(n^3)$; however, it singles out the null space of the singular matrix, successfully avoiding error enlargement and thus retaining the discretization accuracy. Their method also allows non-uniform meshes and hence is applicable to more general singular problems. The method of Bialecki and Remington is also an eigen-decomposition method, which is a modification of Hochney's fast Poisson solver with the singularity singled out for special treatment. Their method thus retains accuracy of discretization and also achieves a high efficiency of $O(n^2 \log n)$. The Kaczmarz projection method is an iterative method that keeps the solution component in the null space of the matrix fixed for each iteration. Thus, it is also a decomposition method which iteratively decomposes the equation into singular and non-singular problems. The Kaczmarz method was later combined with multigrid procedures [7] to improve its convergence (efficiency of the multigrid Kaczmarz method was analyzed by Shapira [20]).

The singular solver proposed in this paper is similar to that of Bialecki and Remington in utilizing FFT for efficient matrix decomposition but with a reduced programming complexity for singularity treatment. In Bialecki and Remington's method, the original discrete singular equation is first perturbed by adding a term to the right hand side of the equation to ensure the solvability of the perturbed equation. They proved that the solution of the perturbed equation is a least square solution of the original singular problem. Our method differs from their method in two places. First, we do not perturb the singular discrete equation. We project the singular equation into the orthogonal complement of the null space of the singular matrix. In the orthogonal complement of the null space, the projected equation is non-singular and always uniquely solvable regardless of the solvability of the original singular discrete system. The second difference (see Section 3) is that we ignore the projected equation in the null space and only solve the projected equation in the orthogonal complement of the null space. We have proven that the solution of the projected equation in the complement of the null space is a solution of the original singular discrete equation when the original is solvable, and is a least-squares solution of the original equation when the original equation is not solvable. Thus, comparing with Bialecki and Remington's method, our method does not compute the perturbation to ensure the solvability, and

[1] The condition number $\mathcal{K}(\mathbf{A})$ of an invertible matrix $\mathbf{A}$ satisfies $\mathcal{K}(\mathbf{A}) = |\lambda_{\max}/\lambda_{\min}|$, where $\lambda_{\max}$ and $\lambda_{\min}$ denote the largest eigenvalue in magnitude and smallest eigenvalue in magnitude, respectively.

also has avoided a process to determine a solution of the projected equation in the null space.

High-order discretization methods for the Laplace operator have been investigated for a long time. Collatz studied several finite difference methods for the 2-D Laplace operator in 1960 [10]. One of the fourth-order methods Collatz studied is a square stencil nine-point scheme which is a popular choice for the Dirichlet problem of Possion equations. In 1975, Lynch and Rice introduced a systematic procedure called HODIE [16] for calculating the coefficients of finite difference discretization formulas for general elliptic equations for almost "any" numerical order. The HODIE procedure was employed by Boisvert in 1981 to discretize the Helmholtz equations [5] and was applied in 1985 to the Neumann problem of the Helmholtz equation (see [18, pp. 199–200]). Another finite difference method for obtaining high-order discretization of the Laplace operator is the Padé-type high-order approximation of Singer and Turkel [21]. A different approach to high-order discretization is the finite element type schemes, among which are Kaufman and Warner's [14] Rayleigh–Ritz–Galerkin method with tensor product B-splines and Bialecki and Fairweather's high-order orthogonal spline collocation method [2].

In this paper we modify the Collatz's popular nine-point scheme for Neumann problems and obtain a fourth order formula which is more general than both the Collatz formula and Boisvert's formula given in ([18, pp. 199–200]).

While our discussion of the singular Poisson solver is restricted to Poisson equations, it is readily extendible to singular Helmholtz equations with either Dirichlet or Neumann boundary conditions. For a Helmholtz equation arising in the time harmonic wave propagation, when the square of the wavenumber happens to be equal to an eigenvalue of the Laplace operator in magnitude, the equation becomes singular. With the proposed singular treatment modified for the non-zero wavenumber cases, singular Helmholtz equations can be solved efficiently and accurately on rectangular domains with uniform meshes chosen, or combined with domain decomposition methods [8, 17] to produce fast and accurate subdomain solvers.

This paper is organized as follows. Fourth order discretizations of the equation and the Neumann boundary condition are presented in Section 2. A decomposition, projection, and solution method for the singular discrete equation are discussed in Section 3. Section 4 contains an error analysis and an efficiency comparison with second order methods. Finally, testing results are presented in Section 5. Section 6 gives the conclusion.

2. DISCRETIZATION

One of the popular high-order discretization schemes for the Laplace operator is the following Collatz [10] formula written in the stencil form,

$$h^{-2}\begin{pmatrix} -\frac{1}{4} & -1 & -\frac{1}{4} \\ -1 & 5 & -1 \\ -\frac{1}{4} & -1 & -\frac{1}{4} \end{pmatrix} p^{i,j} = -\begin{pmatrix} & \frac{1}{8} & \\ \frac{1}{8} & 1 & \frac{1}{8} \\ & \frac{1}{8} & \end{pmatrix} \triangle p^{i,j} + O(h^4), \tag{2.1}$$

where $\triangle p = (\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2})p$. This high-order method together with others studied by Collatz were later generalized by Lynch and Rice to a method called HODIE [16] for general elliptic equations. For Neumann problems, when scheme (2.1) is applied at boundary grid point

(0, j), the right hand side needs the value of $\triangle p^{-1,j}$, which is outside the domain. So this method is not applicable to the Neumann problem of Poisson equations. A modification is hence necessary. In 1987 Boisvert successfully calculated fourth order discretization coefficients using the HODIE method for the Neumann problem of Helmholtz equations [6].

In this paper, we use a fourth order discretization formula which is more general than Boisvert's formula given in ([18, pp. 199–200]). Our discretization formula was first derived in [22]. For self-containedness, we re-present it here.

The right hand side of (2.1) is equal to

$$\frac{3}{2}\triangle p^{i,j} - \frac{1}{8}\begin{pmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{pmatrix}\triangle p^{i,j}.$$

The second term above is obviously an approximation of $(h^2/8)\triangle^2 p^{i,j}$ with an error of $h^2 O(h^2) = O(h^4)$. So we approximate the second term by $(h^2/8)\triangle^2 p^{i,j}$ and arrive at the following modified Collatz method:

$$h^{-2}\begin{pmatrix} -\frac{1}{4} & -1 & -\frac{1}{4} \\ -1 & 5 & -1 \\ -\frac{1}{4} & -1 & -\frac{1}{4} \end{pmatrix} p^{i,j} = \frac{3}{2}\triangle p^{i,j} - \frac{h^2}{8}\triangle^2 p^{i,j} + O(h^4). \tag{2.2}$$

Applying the above scheme to Eq. (1.1) at grid point (i, j) on a uniformly spaced grid, we obtain

$$h^{-2}\begin{pmatrix} -\frac{1}{4} & -1 & -\frac{1}{4} \\ -1 & 5 & -1 \\ -\frac{1}{4} & -1 & -\frac{1}{4} \end{pmatrix} p^{i,j} = -\frac{3}{2}r^{i,j} - \frac{h^2}{8}\triangle r^{i,j} + O(h^4). \tag{2.3}$$

The approximation of $\triangle r^{i,j}$ in the above equation only needs to be second order accurate for the truncation error to remain fourth order. The main advantage of formula (2.3) is that it allows people to choose different approximation schemes for $\triangle r^{i,j}$. This flexibility is especially useful for Neumann problems since for interior grid points, the five-point finite difference can be used, while for boundary points other approximation formulae can be used to avoid using values of $\triangle r$ outside the domain. Formula (2.3) becomes the Collatz formula (2.1) when the five-point finite difference is used to approximate $\triangle r^{i,j}$.

We use the following fourth order discretization for the boundary condition,

$$f_x^i + \frac{1}{6}h^2 f_{xxx}^i = \frac{f^{i+1} - f^{i-1}}{2h} + O(h^4), \tag{2.4}$$

which is a direct application of Taylor expansion up to the fourth derivative terms. Applying (2.4) to the boundary condition at grid point $(0, j)$, we have

$$\frac{p^{1,j} - p^{-1,j}}{2h} = \frac{h^2}{6}b_{xx}^{0,j} + b^{0,j} + O(h^4). \tag{2.5}$$

A difficulty exists in approximating $b_{xx}^{0,j}$ in the above formula. To approximate $b_{xx}^{0,j}$, it requires values of boundary condition b in an x-direction neighborhood of the grid point $(0, j)$, e.g., $b^{-1,j}$ and $b^{1,j}$. A boundary condition is usually given only on the boundary not over a small neighborhood of boundary. So, on an x-direction boundary, b_{xx} cannot be directly approximated but b_{yy} can be approximated directly since on the boundary point

$(0, j)$ the Poisson equation $p_{xx}^{0,j} + p_{yy}^{0,j} = r^{0,j}$ holds, which is equivalent to $p_{xx}^{0,j} = r^{0,j} - p_{yy}^{0,j}$. Taking x-directional partial derivatives of both sides, we obtain $p_{xxx}^{0,j} = r_x^{0,j} - p_{xyy}^{0,j}$, which is the same as $b_{xx}^{0,j} = r_x^{0,j} - b_{yy}^{0,j}$ by noticing that $p_x^{0,j} = b^{0,j}$. Replacing b_{xx} in (2.5) with the above formula, we obtain

$$\frac{p^{1,j} - p^{-1,j}}{2h} = \frac{h^2}{6}\left(r_x - b_{yy}^{0,j}\right) + b^{0,j} + O(h^4). \tag{2.6}$$

The approximation of $b_{yy}^{0,j}$ in the above equation needs to be only second order for (2.6) to remain fourth order. Through the above discretization and derivation, the Poisson equation (1.1) and the boundary conditions (1.2) can be incorporated into the linear system

$$\mathbf{A}\ P = -\frac{3}{2}h^2 r - \frac{h^4}{8}\triangle r + 2hB\ + O(h^5), \tag{2.7}$$

where P denotes the solution vector in natural ordering (see [4, p. 62]), r the vector corresponding to the right hand side of (1.1), and B the vector resulting from the boundary condition (2.6), which vanishes at interior points and is given by

$$B^{0,j} = -\frac{3}{2}p_x^{0,j} - \frac{h^2}{24}\left(r_x^{0,j-1} + 4r_x^{0,j} + r_x^{0,j+1}\right),$$

$$B^{m,j} = \frac{3}{2}p_x^{m,j} + \frac{h^2}{24}\left(r_x^{m,j-1} + 4r_x^{m,j} + r_x^{m,j+1}\right),$$

$$B^{i,0} = -\frac{3}{2}p_y^{i,0} - \frac{h^2}{24}\left(r_x^{i-1,0} + 4r_x^{i,0} + r_x^{i+1,0}\right),$$

$$B^{i,n} = \frac{3}{2}p_y^{i,n} + \frac{h^2}{24}\left(r_x^{i-1,n} + 4r_x^{i,n} + r_x^{i+1,n}\right),$$

for $1 \le i \le m-1, 1 \le j \le n-1$, and

$$B^{0,0} = -\frac{1}{4}\left(5\left(p_x^{0,0} + p_y^{0,0}\right) + p_x^{0,1} + p_y^{1,0}\right) - h^2\frac{5\left(r_x^{0,0} + r_y^{0,0}\right) + r_x^{0,1} + r_y^{1,0}}{24}$$
$$+ \frac{5\left(p_x^{0,0} + p_y^{0,0}\right) - 12\left(p_x^{0,1} + p_y^{1,0}\right) + 9\left(p_x^{0,2} + p_y^{2,0}\right) - 2\left(p_x^{0,3} + p_y^{3,0}\right)}{24},$$

$$B^{m,0} = \frac{1}{4}\left(5\left(p_x^{m,0} - p_y^{m,0}\right) + p_x^{m,1} - p_y^{m-1,0}\right) + h^2\frac{5\left(r_x^{m,0} - r_y^{m,0}\right) + r_x^{m,1} - r_y^{m-1,0}}{24}$$
$$- \frac{5\left(p_x^{m,0} - p_y^{m,0}\right) - 12\left(p_x^{m,1} - p_y^{m-1,0}\right) + 9\left(p_x^{m,2} - p_y^{m-2,0}\right) - 2\left(p_x^{m,3} - p_y^{m-3,0}\right)}{24},$$

$$B^{0,n} = -\frac{1}{4}\left(5\left(p_x^{0,n} - p_y^{0,n}\right) + p_x^{0,n-1} - p_y^{1,n}\right) - h^2\frac{5\left(r_x^{0,n} - r_y^{0,n}\right) + r_x^{0,n-1} - r_y^{1,n}}{24}$$
$$+ \frac{5\left(p_x^{0,n} - p_y^{0,n}\right) - 12\left(p_x^{0,n-1} - p_y^{1,n}\right) + 9\left(p_x^{0,n-2} - p_y^{2,n}\right) - 2\left(p_x^{0,n-3} - p_y^{3,n}\right)}{24},$$

$$B^{m,n} = \frac{1}{4}\left(5\left(p_x^{m,n} + p_y^{m,n}\right) + p_x^{m,n-1} + p_y^{m-1,n}\right) + h^2\frac{5\left(r_x^{m,n} + r_y^{m,n}\right) + r_x^{m,n-1} + r_y^{m-1,n}}{24}$$
$$- \frac{5\left(p_x^{m,n} + p_y^{m,n}\right) - 12\left(p_x^{m,n-1} + p_y^{m-1,n}\right) + 9\left(p_x^{m,n-2} + p_y^{m-2,n}\right) - 2\left(p_x^{m,n-3} + p_y^{m-3,n}\right)}{24}.$$

The matrix $\mathbf{A}$ is an $(m+1)(n+1)$ by $(m+1)(n+1)$ matrix given by

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & -2\mathbf{A}_2 & 0 & \cdot & 0 & 0 & 0 \\ -\mathbf{A}_2 & \mathbf{A}_1 & -\mathbf{A}_2 & \cdot & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & -\mathbf{A}_2 & \mathbf{A}_1 & -\mathbf{A}_2 \\ 0 & 0 & 0 & \cdot & 0 & -2\mathbf{A}_2 & \mathbf{A}_1 \end{pmatrix}, \tag{2.8}$$

where $\mathbf{A}_1$ and $\mathbf{A_2}$ are $(m+1)$ by $(m+1)$ matrices given by

$$\mathbf{A_1} = \begin{pmatrix} 5 & -2 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 5 & -1 & \cdots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & -1 & 5 & -1 \\ 0 & 0 & 0 & \cdots & 0 & -2 & 5 \end{pmatrix}, \quad \mathbf{A_2} = \begin{pmatrix} 1 & \frac{1}{2} & 0 & \cdots & 0 & 0 & 0 \\ \frac{1}{4} & 1 & \frac{1}{4} & \cdots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & \frac{1}{4} & 1 & \frac{1}{4} \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{2} & 1 \end{pmatrix}.$$

Formula (2.7) is not fully discretized. It still contains differential terms such as $\triangle r$, r_x, and r_y. This is the main difference between our discretization formula and existing ones which are usually fully discretized. But since $r(x, y)$ are given functions with known values and approximation of these differential terms $\triangle r$, r_x and r_y needs to be only $O(h^2)$ for maintaining an $O(h^5)$ accuracy for formula (2.7), we can easily find many simple second order approximations (e.g., 1-D formulae in [15]). We can also obtain Boisvert's formula[2] (see [18, p. 200]) by choosing different second-order discretization formulae for $\triangle r$, r_x, and r_y at different grid points.

3. SINGULAR VALUE DECOMPOSITION AND PROJECTION

After further discretizing the terms $\triangle r$, r_x, and r_y on the right-hand side of (2.7), we obtain a linear system of the form

$$\mathbf{A}P = R \tag{3.1}$$

in space $\mathbf{R}^{(m+1)(n+1)}$, where $\mathbf{A}$ is given by (2.8). For notational simplicity, we let $N = (m+1)(n+1)$. It is easy to verify that the matrix $\mathbf{A}$ is singular and its rank is $N-1$. Let $N(\mathbf{A})$ denote the null space of matrix $\mathbf{A}$ and $N(\mathbf{A})^\perp$ the orthogonal complement of $N(\mathbf{A})$, namely,

$$N(\mathbf{A}) = \{P \in \mathbf{R}^N : \mathbf{A}P = 0\}, \quad \text{and}$$
$$N(\mathbf{A})^\perp = \{P \in \mathbf{R}^N : P^t Q = 0 \quad \text{for all} \quad Q \in N(\mathbf{A})\},$$

where P^t denotes the transpose of column vector P. Let $\mathbf{F}$ be an $N \times (N-1)$ matrix whose $(N-1)$ column vectors are chosen to form an orthonormal basis of the space $N(\mathbf{A})^\perp$.

[2] The right hand side of the Boisvert formula as given in Table 1 in ([18, p. 200) seems to have printing errors. The first row should be $I_h g + h^{-1} J_h u_n$ instead of $I_h g + J_h u_n$ to be consistent with the equation on p. 199, and the terms u_x, u_y in the last two rows should be multiplied by h^{-1}. When these corrections are made to Boisvert's formula, our discretization formula (2.7) contains the Boisvert's formula as a special case.

Then $\mathbf{F}^t\mathbf{F} = \mathbf{I}_{N-1}$, where $\mathbf{I}_{N-1}$ denotes the identity matrix in space $\mathbf{R}^{N-1}$. Define a matrix $\mathbf{\Pi} = \mathbf{F}\mathbf{F}^t$. It is easy to check that $\mathbf{\Pi}$ is the projection matrix from space $\mathbf{R}^N$ to $N(\mathbf{A})^\perp$; i.e., $\mathbf{\Pi}P \in N(\mathbf{A})^\perp$ for $P \in \mathbf{R}^N$ and $\mathbf{\Pi}P = P$ for $P \in N(\mathbf{A})^\perp$.

Now we project Eq. (3.1) onto $N(\mathbf{A})^\perp$ and obtain the projected equation $\mathbf{\Pi}\mathbf{A}P = \mathbf{\Pi}R$. Since $\mathbf{A}P \in N(\mathbf{A})^\perp$, we have that $\mathbf{\Pi}\mathbf{A}P = \mathbf{A}P$. Hence the projected equation becomes

$$\mathbf{A}P = \mathbf{\Pi}R. \tag{3.2}$$

The solvability of a singular linear system, as stated below, is known [3, 11].

PROPOSITION 3.1. *A singular linear system of form* (3.1) *is solvable if and only if its right hand side is orthogonal to* $N(\mathbf{A})$, *i.e.* $R \in N(\mathbf{A})^\perp$.

However, the projected equation (3.2), as we shall show below, is always solvable even when the original singular equation (3.1) has no solution.

THEOREM 3.1.

(i) *The projected equation* (3.2) *has a unique solution in* $N(\mathbf{A})^\perp$.

(ii) *If* $P \in N(\mathbf{A})^\perp$ *is the solution of the projected equation* (3.2), *then* P *is a solution of* (3.1) *when Eq.* (3.1) *is solvable*; *and* P *is a least-squares solution of* (3.1) *when Eq.* (3.1) *is not solvable, where a least-squares solution* P *of* (3.1) *is defined as*

$$(\mathbf{A}P - R)^t(\mathbf{A}P - R) = \min_{Q \in \mathbf{R}^N} (\mathbf{A}Q - R)^t(\mathbf{A}Q - R).$$

Proof.

(i) Since $\mathbf{\Pi}R \in N(\mathbf{A})^\perp$, by Proposition 3.1 the projected equation (3.2) has a solution $P \in \mathbf{R}^N$. But $\mathbf{A}P = \mathbf{A}\mathbf{\Pi}P$, so we have that $\mathbf{A}\mathbf{\Pi}P = \mathbf{\Pi}R$, which means that $\mathbf{\Pi}P \in N(\mathbf{A})^\perp$ is a solution of (3.2).

To prove the uniqueness of the solution, we let $P, Q \in N(\mathbf{A})^\perp$ satisfying Eq. (3.2). Thus, $\mathbf{A}(P - Q) = 0$, which means that $(P - Q) \in N(\mathbf{A})$. But $(P - Q) \in N(\mathbf{A})^\perp$ since both $P, Q \in N(\mathbf{A})^\perp$. So $(P - Q) = 0$, which proves the uniqueness.

(ii) Let $P \in N(\mathbf{A})^\perp$ be the solution of (3.2). We first show that if the original singular equation (3.1) is solvable, P is also a solution of (3.1). By Proposition 3.1, the solvability of Eq. (3.1) implies that $R \in N(\mathbf{A})^\perp$. Hence $\mathbf{\Pi}R = R$. Then $\mathbf{A}P = R$, which shows that P is a solution of (3.1).

Now we shall show that when Eq. (3.1) has no solution, P is a least-squares solution of (3.1). Since for all $Q \in \mathbf{R}^N$, $\mathbf{A}Q \in N(\mathbf{A})^\perp$, we have that

$$\min_{Q \in \mathbf{R}^N} (\mathbf{A}Q - R)^t(\mathbf{A}Q - R) \le \min_{U \in N(\mathbf{A})^\perp} (U - R)^t(U - R) = (\mathbf{\Pi}R - R)^t(\mathbf{\Pi}R - R). \tag{3.3}$$

Since P is the solution of (3.2), we have that $(\mathbf{\Pi}R - R)^t(\mathbf{\Pi}R - R) = (\mathbf{A}P - R)^t(\mathbf{A}P - R)$, which, together with (3.3), leads to $\min_{Q \in \mathbf{R}^N}(\mathbf{A}Q - R)^t(\mathbf{A}Q - R) = (\mathbf{A}P - R)^t(\mathbf{A}P - R)$. **Q.E.D.**

Due to the relation between the solution of the projected equation and the original singular equation established in the theorem above, our solution algorithm is designed to solve the projected equation (3.2) regardless of the solvability of the original singular equation. That

is one of the differences between our algorithm and Bialecki and Remington's solver, where the right hand side of the original singular equation is perturbed to guarantee the solvability of the perturbed equation.

Since $\mathbf{\Pi} = \mathbf{F}\mathbf{F}^t$, the projected equation (3.2) is the same as

$$\mathbf{A}\mathbf{F}\mathbf{F}^t P = \mathbf{F}\mathbf{F}^t R. \tag{3.4}$$

The column vectors of $\mathbf{F}$ form an orthonormal basis of $N(\mathbf{A})^{\perp}$, so for any vector $Q \in \mathbf{R}^N$, the matrix–vector product $\mathbf{F}^t Q$ gives the coefficients of Q with respect to the orthonormal basis $\mathbf{F}$. Our algorithm starts by multiplying $\mathbf{F}^t$ to both sides of (3.4), yielding

$$(\mathbf{F}^t\mathbf{A}\mathbf{F})(\mathbf{F}^t P) = (\mathbf{F}^t\mathbf{F})\mathbf{F}^t R.$$

Since $\mathbf{F}^t\mathbf{F} = \mathbf{I}_{N-1}$, the above equation is equivalent to

$$(\mathbf{F}^t\mathbf{A}\mathbf{F})(\mathbf{F}^t P) = \mathbf{F}^t R. \tag{3.5}$$

Then we solve the above equation for the coefficient vector $\mathbf{F}^t P$. Finally, we multiply $\mathbf{F}$ by $\mathbf{F}^t P$ to recover the solution $\mathbf{F}\mathbf{F}^t P = \mathbf{\Pi} P$ of the projected equation (3.2).

The decomposition and projection method discussed above works for all singular linear systems, not just for singular discrete Poisson equations. For general problems, however, the main difficulty is how to find a matrix $\mathbf{F}$ that can efficiently accomplish the projection and recovery operations. For the singular Poisson problem, utilizing the eigenvectors of $\mathbf{A}$ we construct matrix $\mathbf{F}$ by

$$\mathbf{F} = \begin{pmatrix} \frac{1}{\sqrt{2}}\mathbf{I}_m & c_{0,1}\mathbf{I}_m & c_{0,2}\mathbf{I}_m & \cdots & c_{0,n}\mathbf{I}_m \\ \frac{1}{\sqrt{2}}\mathbf{I}_m & c_{1,1}\mathbf{I}_m & c_{1,2}\mathbf{I}_m & \cdots & c_{1,n}\mathbf{I}_m \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ \frac{1}{\sqrt{2}}\mathbf{I}_m & c_{n-1,1}\mathbf{I}_m & c_{n-1,2}\mathbf{I}_m & \cdots & c_{n-1,n}\mathbf{I}_m \\ \frac{1}{\sqrt{2}}\mathbf{I}_m & c_{n,1}\mathbf{I}_m & c_{n,2}\mathbf{I}_m & \cdots & c_{n,n}\mathbf{I}_m \end{pmatrix} \begin{pmatrix} \mathbf{C}_{m-1} & 0 & \cdots & 0 & 0 \\ 0 & \mathbf{I}_m & \cdots & 0 & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & \cdots & \mathbf{I}_m & 0 \\ 0 & 0 & \cdots & 0 & \mathbf{I}_m \end{pmatrix}, \tag{3.6}$$

where $c_{ij} = \sqrt{\frac{2}{n+1}}\cos(\frac{ij\pi}{n})$ for $i = 0, 1, 2, \ldots, n,\ j = 1, 2, \ldots, n$, $\mathbf{C_{m-1}} = (c_{i,j})$ is an $m \times (m-1)$ matrix with $i = 0, 1, 2, \ldots, m$ and $j = 1, 2, \ldots, m$, and $\mathbf{I}_m$ is the identity matrix in the space $\mathbf{R}^m$. With the choice of this matrix $\mathbf{F}$, we can accomplish the two matrix–vector multiplication operators $\mathbf{F}^t R$ and $\mathbf{F}(\mathbf{F}^t P)$ efficiently using FFT, and at the same time tridiagonalize the matrix $\mathbf{A}$ into $\mathbf{F}^t\mathbf{A}\mathbf{F}$ in Eq. (3.5).

Denote the first matrix on the right hand side of (3.6) by $\mathbf{F}_1$ and the second matrix by $\mathbf{F}_2$, namely, $\mathbf{F} = \mathbf{F}_1\mathbf{F}_2$. The $N \times N$ matrix $\mathbf{F}_1$ is the cosine transform matrix Boisvert used in the FFT solver for the nonsingular Neumann problem of Helmholtz equations [6]. For the singular Poisson problem, $\mathbf{F}_1$ tridiagonalizes (and also block-diagonalizes) $\mathbf{A}$ into $\mathbf{F}_1^t\mathbf{A}\mathbf{F}_1$, resulting in $n+1$ independent subequations with one subequation being singular. As mentioned in the Introduction, a difference exists between our algorithm and Bialecki and Remington's solver in the treatment of the discrete singular subequation . In our algorithm, further applying the $N \times (N-1)$ matrix $\mathbf{F}_2$ to the already tridiagonalized matrix $\mathbf{F}_1^t\mathbf{A}\mathbf{F}_1$ completes the projection operation so that the projected equation is uniquely solvable

in $N(\mathbf{A})^{\perp}$, while in Bialecki and Remington's solver, after obtaining $n+1$ independent discrete subequations (one of them singular) by using a cosine transform, to handle the singular discrete subequation they return to the continuous (nondiscrete) equation which the singular discrete subequation approximates at collocation points and impose a Dirichlet boundary condition at one boundary point for the continuous subequation, thus turning it into a non-singular subequation.

Returning to our algorithm, we summarize below the solution process of our solver.

1. Compute the right hand side of (2.7).
2. Multiply $\mathbf{F}^t$ to the right hand side of Eq. (3.1) to obtain (3.5).
3. Compute the entries in matrix $\mathbf{F}^t\mathbf{A}\mathbf{F}$ and solve the tridiagonal system (3.5) for the coefficient vector $\mathbf{F}^t P$.
4. Recover $\mathbf{\Pi} P$ by multiplying $\mathbf{F}$ to the coefficients $\mathbf{F}^t P$.

The operation count of each step of the algorithm on a square domain of $n \times n$ is

1. $5n^2 + O(n)$ floating point operations;
2. $2.5n^2 \log_2 n + 1.5n^2 + O(n \log_2 n)$ operations;
3. $8n^2 + O(n)$ operations;
4. $2.5n^2 \log_2 n + 2.5n^2 + O(n \log_2 n)$ operations.

The total operation count of the algorithm, the sum of the work of the four steps, is

$$5n^2 \log_2 n + 17n^2 + O(n \log_2 n). \tag{3.7}$$

Since the second order finite difference approximation of the Laplace operator produces a matrix with the same eigenvectors as that produced by the fourth order discretization (2.2), the decomposition and projection-based SVD method discussed above is also applicable to the discrete system obtained via the second order discretization. A second order solver with this SVD treatment goes through the same four steps as the fourth order method, and the operation counts only differ in step 1—the second order solver needs only n^2 operations in approximating the right hand side of the discrete matrix equation, resulting a total count of

$$5n^2 \log_2 n + 13n^2 + O(n \log_2 n), \tag{3.8}$$

for the second order method.

4. ERROR AND EFFICIENCY

Assuming that the solution $p(x, y)$ is sufficiently smooth, the truncation error of (2.7) when all differential operators replaced by their respective discrete versions is

$$t(i,j) = \left[\frac{h^6}{144}\left(\frac{\partial^6}{\partial x^4 \partial y^2} + \frac{\partial^6}{\partial x^2 \partial y^4}\right) - \frac{h^6}{240}\left(\frac{\partial^6}{\partial x^6} + \frac{\partial^6}{\partial y^6}\right) + \frac{h^6}{96}\left(\frac{\partial^4}{\partial y^4} + \frac{\partial^4}{\partial x^4}\right)\triangle\right] p^{i,j},$$

$$t(0,j) = \left[\frac{h^6}{144}\left(\frac{\partial^6}{\partial x^4 \partial y^2} + \frac{\partial^6}{\partial x^2 \partial y^4}\right) - \frac{h^6}{240}\left(\frac{\partial^6}{\partial x^6} + \frac{\partial^6}{\partial y^6}\right) + \frac{h^6}{96}\left(\frac{\partial^4}{\partial y^4} + 11\frac{\partial^4}{\partial x^4}\right)\triangle\right] p^{0,j} - h^5\left[\frac{1}{40}\frac{\partial^5}{\partial x^5} + \frac{1}{6}\frac{\partial^3}{\partial x^3}\triangle + \frac{1}{24}\frac{\partial^5}{\partial x \partial y^4}\right] p^{0,j},$$

$$
\begin{aligned}
t(0,0) = {} & h^6\left[\frac{1}{144}\left(\frac{\partial^6}{\partial x^4\partial y^2}+\frac{\partial^6}{\partial x^2\partial y^4}\right)-\frac{1}{240}\left(\frac{\partial^6}{\partial x^6}+\frac{\partial^6}{\partial y^6}\right)+\frac{11}{96}\left(\frac{\partial^4}{\partial y^4}+\frac{\partial^4}{\partial x^4}\right)\triangle\right]p^{0,0}\\
& -h^5\left[\frac{1}{240}\left(\frac{\partial}{\partial x}+\frac{\partial}{\partial y}\right)^5+\frac{1}{60}\left(\frac{\partial^5}{\partial x^5}+\frac{\partial^5}{\partial y^5}\right)+\frac{5}{36}\left(\frac{\partial^3}{\partial x^3}+\frac{\partial^3}{\partial y^3}\right)\triangle\right.\\
& \left.+\frac{11}{24}\left(\frac{\partial^5}{\partial x\partial y^4}+\frac{\partial^5}{\partial x^4\partial y}\right)\right]p^{0,0}\\
& -h^5\left(\frac{1}{240}\frac{\partial^5}{\partial x^5}+\frac{1}{36}\frac{\partial^3}{\partial x^3}\triangle+\frac{1}{144}\frac{\partial^5}{\partial x\partial y^4}\right)p^{0,1}\\
& -h^5\left(\frac{1}{240}\frac{\partial^5}{\partial y^5}+\frac{1}{36}\frac{\partial^3}{\partial y^3}\triangle+\frac{1}{144}\frac{\partial^5}{\partial x^4\partial y}\right)p^{1,0},
\end{aligned}
$$

where $t(i, j)$ denotes the truncation error at grid point (i, j).

The solution error e, defined as the difference between the true solution and the computed solution, satisfies $\mathbf{A}\,e = t$. Since the solution of a singular system has total freedom in its null space, so no error exists with respect to the zero eigenvalue. Thus, the relation between solution error e and truncation error t satisfies $\mathbf{A}_\perp e = t$, where $\mathbf{A}_\perp$ denotes the projection of matrix $\mathbf{A}$ in space $N(\mathbf{A})^\perp$. Since $\mathbf{A}_\perp$ is invertable, we obtain

$$e = \mathbf{A}_\perp^{-1} t. \tag{4.1}$$

Matrix $\mathbf{A}_\perp$ has eigenvalues

$$\lambda_{k,l} = 5 - 2\cos\left(\frac{i\pi}{m}\right) - 2\cos\left(\frac{j\pi}{n}\right) - \cos\left(\frac{i\pi}{m}\right)\cos\left(\frac{j\pi}{n}\right) \tag{4.2}$$

for $i = 0, 1, \ldots, m$, $j = 0, 1, \ldots, n$ with $(i, j) \neq (0, 0)$.

Matrix $\mathbf{A}_\perp^{-1}$ has the same eigenvectors as $\mathbf{A}_\perp$ with corresponding eigenvalues $1/\lambda_{ij}$. Since there are $(mn + m + n)$ distinct eigenvalues in the $(mn + m + n)$-dimensional discrete space $N(\mathbf{A})^\perp$, the $(mn + m + n)$ eigenvectors are mutually orthogonal and thus span the $(mn + m + n)$ dimensional space in which we are solving the equation. Therefore the truncation error t can be expanded in terms of the orthonormal eigenvectors V_{ij} for $i = 0, 1, \ldots, m$ and $j = 1, \ldots, n$ as

$$t = \sum_{i,j} c_{ij} V_{ij}\ . \tag{4.3}$$

Since the truncation error t is of $O(h^5)$, (4.3) means that $\sum_{i,j} c_{ij} V_{ij}$ is of $O(h^5)$. And by (4.1) and (4.3)

$$\|e\|_2 = \|\mathbf{A}_\perp^{-1} t\|_2 = \sqrt{\sum_{i,j} \frac{c_{ij}^2}{\lambda_{ij}^2}}. \tag{4.4}$$

Therefore,

$$\sqrt{\sum_{i,j} \frac{c_{ij}^2}{\lambda_{max}^2}} \le \|e\|_2 \le \sqrt{\sum_{i,j} \frac{c_{ij}^2}{\lambda_{min}^2}}. \tag{4.5}$$

The eigenvalues (4.2) of $\mathbf{A}_\perp$ are positive and satisfy

$$\lambda_{i-1,j} < \lambda_{i,j}, \quad \text{and} \quad \lambda_{i,j-1} < \lambda_{i,j}.$$

Since $\lambda_{0,1}$ and $\lambda_{1,0}$ are of $O(h^2)$, $\lambda_{0,\sqrt{n}}$ and $\lambda_{\sqrt{m},0}$ are of $O(h)$, and $\lambda_{i,j}$ is of $O(1)$ for all (i, j) pairs such that $i \geq \frac{m}{10}$ or $j \geq \frac{n}{10}$, it is clear from (4.5) that the order of e ranges from $O(h^3)$ to $O(h^5)$. The distribution of the truncation error $\sum_{i,j} c_{ij} V_{ij}$ depends mainly on the problem (i.e, solution p) and discretization methods. For the solution to be of $O(h^3)$, the truncation error t must concentrate on the near-zero low frequence in the sense that coefficients $c_{i,j}$ in (4.3) are close to 0 for i, j not near zero. Assuming uniform distribution for the coefficients $c_{i,j}$'s, this solution method is in average case fourth order.

With the error estimation given above, we can proceed to give a comparison of efficiency for fourth and second order methods in terms of execution time.

Execution time in general is approximately proportional to the number of operations. For a given error tolerance, a high-order method allows much larger mesh sizes than a lower order method, resulting in significant reduction in the number of grid points and consequently execution time if the high-order method has the same computation complexity as that of the second order method for the same grid size. Such time reduction can be seen clearly from the discussion below for a fourth order method against a second order method of the same complexity.

For the sake of brevity, we restrict our discussion on the unit square domain $[0, 1] \times [0, 1]$. With slight modifications, the same analysis can be conducted for general rectangular domains. We introduce the following notations: $E(Mthd)$ denotes the difference between the true solution and the numerical solution computed by method *Mthd*; $\varepsilon > 0$ is the error tolerance, i.e., the difference between the computed numerical solution and the true solution must be less than or equal to ε. With these notations, the error of our fourth order method can be denoted by $E(order4)$, and the error of the second order Poisson solver will be $E(order2)$. The solution error of the fourth order direct method in general satisfies

$$E(order4) = a \cdot h^4 \text{ for some problem-dependent coefficient } a.$$

The error of a second order solution method in general satisfies

$$E(order2) = b \cdot h^2 \text{ for some problem-dependent coefficient } b.$$

To meet the error tolerance, the fourth and second order methods need to take different mesh sizes and partition sizes, say partition size N and mesh size h for the fourth order solver, and partition size N' and mesh size h' for the second order method. Then

$$a \cdot h^4 \leq \varepsilon \quad \text{and} \quad b \cdot h'^2 \leq \varepsilon.$$

Roughly we can equate them to yield

$$a \cdot h^4 = b \cdot h'^2. \tag{4.6}$$

Since $h = 1/N$ and $h' = 1/N'$, (4.6) is equivalent to

$$N' = \sqrt{\frac{b}{a}}\, N^2 \;=\; C\, N^2, \tag{4.7}$$

where

$$C = \sqrt{\frac{b}{a}}\,. \tag{4.8}$$

Thus, if our fourth order solver can satisfy the error tolerance by taking a partition of size N, then it requires the second order solver to take a partition size of CN^2 to achieve the same accuracy. Let T_4 and T_2 denote the time needed by the order 4 and order 2 methods respectively to solve a problem within a given error tolerance. Then (3.8), (3.7), and (4.7) imply that

$$T_2 : T_4 \;=\; \frac{C^2N^2(5\log_2 CN^2 + 13)}{(5\log_2 N + 17)}. \tag{4.9}$$

The parameter C in general could vary largely from problem to problem. For Poisson equations which have only twice differentiable solutions, the fourth order method has only second order accuracy, and probably has no gain in reducing execution time for a given error tolerance. But for problems with at least three times differentiable solutions, the fourth order can take advantage of the smoothness of the solution and reduce the computation cost for a given error tolerance.

5. EXPERIMENT RESULTS

To test the accuracy and efficiency of the high-order fast singular Poisson solver (HFSPS), we choose four testing problems with solutions of different orders of differentiability; they are:

1. $p(x, y) = (xy)^{3.5}[1 - \cos(xy)]$, which is five times differentiable;
2. $p(x, y) = x^{4.5} + y^{4.5}$, which is four times differentiable;
3. $p(x, y) = (x + y)^{2.5}\sin(x)$, which is three times differentiable;
4. $p(x, y) = (x + y)^{2.5}$, which is twice differentiable;

The testing problem domain is chosen to be the unit square $[0, 1] \times [0, 1]$, and uniform mesh size $h = 1/N$ is chosen on each dimension, where N is the number of grid points on each x- and y-dimension. We tested the HFSPS on an IBM RS/6000 machine running operating system AIX 3.2.5, and the test results are listed in Tables I to V. For the comparison of accuracy, on the same machine we also tested a second-order method (FSPS) with the same decomposition and projection-based SVD method described in Section 3 and the traditional five-point second order discretization. The test data of FSPS are also listed in the tables.

Tables I to IV present the time-accurate comparison between the HFSPS solver and the second order FSPS solver for the four test problems. Measured experimental results show that the HFSPS method is much more accurate and achieves high accuracy without increasing execution time as compared with the second order solver. In the tables, we use a metric Order [23, 26] to indicate the numerical order of a solver, which is calculated as follows:

$$\text{Order}(n, 2n) = \log_2 \frac{\text{Error}(n)}{\text{Error}(2n)}.$$

TABLE I

Problem 1: $p_{xx} + p_{yy} = (xy)^{3.5}[1 - \cos(xy)]$, Five Times Differentiable

Method	n	16	32	64	128	256	512	1024
HFSPS	Error	1.54–04	1.00–05	6.36–07	3.99–08	2.50–09	1.56–10	9.65–12
	Order		3.9	4.0	4.0	4.0	4.0	4.0
	Time	0.01 s	0.02 s	0.11 s	0.45 s	1.74 s	7.23 s	30.2 s
FSPS	Error	6.07–03	1.51–03	3.76–04	9.41–05	2.35–05	5.88–06	1.47–06
	Order		2.0	2.0	2.0	2.0	2.0	2.0
	Time	0.01 s	0.02 s	0.11 s	0.40 s	1.76 s	7.23 s	30.5 s

TABLE II

Problem 2: $p(x, y) = x^{4.5} + y^{4.5}$, Four Times Differentiable

Method	n	16	32	64	128	256	512	1024
HFSPS	Error	2.39–04	2.13–05	1.89–06	1.68–07	1.49–08	1.32–09	1.23-10
	Order		3.5	3.5	3.5	3.5	3.5	3.4
	Time	0.01 s	0.03 s	0.10 s	0.44 s	1.84 s	7.43 s	31.1 s
FSPS	Error	1.15–02	2.89–03	7.21–04	1.80–04	4.51–05	1.l3–05	2.82–06
	Order		2.0	2.0	2.0	2.0	2.0	2.0
	Time	0.01 s	0.02 s	0.11 s	0.43 s	1.75 s	7.15 s	31.5 s

TABLE III

Problem 3: $p(x, y) = (x + y)^{2.5} \sin(x)$, Three Times Differentiable

Method	n	16	32	64	128	256	512	1024
HFSPS	Error	8.12–05	7.30–06	6.96–07	6.81–08	6.78–09	6.68–10	7.30–11
	Order		3.5	3.4	3.4	3.3	3.3	3.2
	Time	0.01 s	0.02 s	0.11 s	0.43 s	1.76 s	7.25 s	31.0 s
FSPS	Error	3.94–03	9.78–04	2.45–04	6.12–05	1.53–05	3.83–06	9.57–07
	Order		2.0	2.0	2.0	2.0	2.0	2.0
	Time	0.01 s	0.03 s	0.11 s	0.50 s	1.80 s	7.19 s	31.0 s

TABLE IV

Problem 4: $p(x, y) = (x + y)^{2.5}$, Twice Differentiable

Method	n	16	32	64	128	256	512	1024
HFSPS	Error	1.00–03	2.03–04	4.04–05	7.94–06	1.55–06	2.98–07	5.72–08
	order		2.3	2.3	2.3	2.4	2.4	2.4
	Time	0.01 s	0.02 s	0.11 s	0.43 s	1.80 s	7.28 s	30.9 s
FSPS	Error	6.23–03	1.54–03	3.79–04	9.36–05	2.31–05	5.72–06	1.42–06
	order		2.0	2.0	2.0	2.0	2.0	2.0
	Time	0.01 s	0.03 s	0.11 s	0.43 s	1.78 s	7.30 s	30.6 s

TABLE V
Computation Time of the Two Methods for the Same Accuracy

Problem	Method	n	Error	Time (sec.)	Time ratio
1	HFSPS	16	1.54–04	0.01	
	FSPS	64	3.76–04	0.11	11
1	HFSPS	32	1.00–05	0.02	
	FSPS	256	2.35–05	1.76	88
2	HFSPS	32	2.13–05	0.03	
	FSPS	256	4.51–05	1.75	58
2	HFSPS	64	1.89–06	0.10	
	FSPS	1024	2.82–06	31.5	315
3	HFSPS	16	8.12–05	0.01	
	FSPS	128	2.05–04	0.45	45
3	HFSPS	64	6.96–07	0.11	
	FSPS	1024	9.57–07	30.1	273
4	HFSPS	32	2.03–04	0.02	
	FSPS	64	3.79–04	0.11	5
4	HFSPS	256	1.55–06	1.80	
	FSPS	1024	1.42–06	30.6	17

The definition of this metric is based on the observation that for a numerical method of order s, the error will decrease at a rate of $\left(\frac{1}{2}\right)^s$ when a uniformly spaced grid doubles its grid points. The log plot of error against grid size (or mesh size) is usually used to measure the order of a numerical method. The metric *Order* used here gives the value of the slope of the log plot of the error vs grid size between each two neighboring testing grid sizes. Since the slope of a curve is difficult to exactly visually determine, the metric *Order* is a clearer quantitative indication of the order of a numerical method. The error analysis given in Section 4 shows that the HFSPS method is fourth order in the average case if the true solution is five times differentiable, and the order of our numerical method decreases as the differentiability of the solution falls below the order of 5. This is matched by the experimental results shown.

Table V compares the measured execution times of the two tested solvers. The four test problems are solved by the high-order HFSPS method. Then the same problems are solved with the second order method to match the achieved accuracy with an increased number of grid points and execution time. The execution times of the HFSPS and the FSPS algorithms are listed side-by-side in Table V for each of the testing problems. Table V shows that the high-order method is 5 to 300 times faster depending upon problem and grid size, as indicated by the column of time ratios for the two solvers. Notice that the performance gain increases when the problem size increases.

6. CONCLUSION

We present a fourth order fast solver for the singular Neumann boundary problem of Poisson equations on a rectangular domain. A modified Collatz finite difference scheme is used to discretize the Laplace operator. This discretization produces a singular discrete

equation which is projected into the orthogonal complement of the null space of the singular matrix and solved in the complement of the null space. It is proven that the solution of the projected equation is a solution of the original singular discrete equation when the original equation is solvable. The projection of the singular equation into the complement of the null space utilizes the fast Fourier transform whose application to Poisson equations was pioneered by Hockney. As both analytical and testing results show, our proposed SVD keeps the accuracy obtained from the high-order discretization while maintaining high efficiency.

ACKNOWLEDGMENTS

This research was supported in part by the National Science Foundation (NSF) under NSF Grant CCR-9972251 and by the Office of Naval Research (ONR) under the PET program. We also thank the anonymous referees for their comments and suggestions.

REFERENCES

1. H. M. Atassi, Unsteady aerodynamics of vortical flows: Early and recent developments, in *Aerodynamics and Aeroacoustics*, edited by K. Y. Fung (World Scientific, Singapore 1994), pp. 119–169.
2. B. Bialecki and G. Fairweather, Matrix decomposition algorithms in orthogonal spline collocation for separable elliptic boundary value problems, *SIAM J. Sci. Comput.* **16**, 330 (1995).
3. B. Bialecki and K. A. Remington, Fourier matrix decomposition methods for the least squares solution of singular Neumann and periodic Hermite bicubic collocation problems, *SIAM J. Sci. Comput.* **16**, 431 (1995).
4. G. Birkhoff and R. Lynch, *Numerical Solution of Elliptic Problems* (SIAM, Philadelphia, 1984).
5. R. F. Boisvert, Families of high order accurate discretizations of some elliptic problems, *SIAM J. Sci. Stat. Comput.* **2**, 268 (1981).
6. R. F. Boisvert, A fourth-order-accurate Fourier method for the Helmholtz equation in three dimensions, *ACM Trans. Math. Software* **13**, 221 (1987).
7. A. Brandt and S. Ta'asan, Multigrid methods for nearly singular and slightly indefinite problems, in *Lecture Notes in Mathematics 1228: Multigrid Methods II*, edited by W. Hackbusch and U. Trottenberg (Springer-Verlag, Berlin/New York, 1985) pp. 100–122.
8. X. Cai, M. A. Cassarin, F. W. Elliott, and J. O. B. Widlund, Overlapping Schwarz algorithms for solving Helmholtz equations, *Contemp. Math.* **218**, 437 (1998).
9. A. J. Chorin, Numerical solution of the Navier–Stokes equations, *Math. Comput*. **22**, 745 (1968).
10. L. Collatz, *The Numerical Treatment of Differential Equation*, (Springer-Verlag, New York, 1960).
11. G. H. Golub, L. C. Huang, H. Simon, and W. Tang, *A Fast Solver for Incompressible Navier–Stokes Equations with Finite Difference Methods.* Stanford University SCCM Technical Report, Stanford University Scientific Computing and Computational Mathematics Program (1994).
12. I. Harari and E. Turkel, Accurate finite difference methods for time-harmonic wave propogation , *J. Comput. Phys.* **119**, 252 (1995).
13. R. Hockney, A fast direct solution of Poisson's equation using Fourier analysis, *J. ACM* **12**, 95 (1965).
14. L. Kaufman and D. Warner, *A program for solving separable elliptic equations, ACM Trans Math. Software* **16**, 325 (1990).
15. S. Lele, Compact finite difference schemes with spectral-like resolution, *J. Comput. Phys.* **103**, 16 (1992).
16. R. E. Lynch and J. R. Rice, High accuracy finite difference approximation to solutions of elliptic partial differential equations, *Proc. Nat. Acad. Sci.* **75**, 2541 (1978).
17. L. C. McInnes, R. Susan-Resiga, D. E. Keyes, and H. M. Atassi, Additive schwarz methods with nonreflecting boundary conditions for the parallel computation of Helmholtz problems, *Contemp. Math.* **218**, 349 (1998).
18. J. R. Rice and R. F. Boisvert, *Solving Elliptic Problems Using ELLPACK* (Springer-Verlag, New York, 1985).

19. U. Schumann and R. Sweet, A direct method for the solution of Poisson equation with Neumann boundary conditions on a staggered grid of arbitrary sizes, *J. Comput. Phys.* **20**, 171 (1976).
20. Y. Shapira, Multigrid methods for 3-D definite and indefinite problems, *Appl. Numer. Math.* **26**, 165 (1998).
21. I. Singer and E. Turkel, High order finite differnce methods for the Helmholtz equation, *Comput. Meth. Appl. Mech. Eng.* **163**, 533 (1998).
22. X.-H. Sun and Y. Zhuang, *A high-order direct solver for Helmholtz equations with Neumann boundary conditions.* NASA ICASE Technical Report No. 97-11, NASA Langley Research Center Hampton, VA 23681-0001 (1997).
23. X.-H. Sun and Y. Zhuang, A highly accurate fast solver for helmholtz equations, in *Proc. ACM International Conference on Supercomputing* (July 1997).
24. K. Tanabe, Projection methods for solving a singular system of linear, *Numer. Math.* **17**, 203 (1971).
25. R. Temem, *Navier–Stokes Equations, Theory and Numerical Analysis* (Elsevier, New York, 1984).
26. Y. Zhuang and X.-H. Sun, A high-order ADI solver for separable generalized Helmholtz equations, *Advances in Engineering Software* **31**, 585 (2000).