SIAM J. Sci. Stat. Comput. ©1997 Society for Industrial and Applied Mathematics
Vol. XX, No. Y, pp. xy-xz, ZZ 1997

EFFICIENT ALGORITHMS FOR SOLVING A FOURTH ORDER EQUATION WITH THE SPECTRAL-GALERKIN METHOD *

PETTER E. BJØRSTAD † AND BJØRN PETER TJØSTHEIM ‡

Abstract. We show that one can derive an $O(N^3)$ spectral-Galerkin method for fourth order (biharmonic type) elliptic equations based on the use of Chebyshev polynomials. The use of Chebyshev polynomials provides a fast transform between physical and spectral space which is advantageous when a sequence of problems must be solved e.g., as part of a nonlinear iteration. This improves the result of Shen [9] which reported an $O(N^4)$ algorithm inferior to the $O(N^3)$ method developed earlier [8] based on Legendre polynomials, but less practical in the case of multiple problems. We further compare our method with an improved implementation of the Legendre-Galerkin method based on the same approach.

Key words. spectral-Galerkin method, Chebyshev polynomial, Legendre polynomial, Helmholtz equation, biharmonic equation, direct solver, iterative solver.

AMS(MOS) subject classification. 65N35, 65N22, 65F05, 35J05.

1. Introduction. In his recent work [8] and [9] Shen develops a class of spectral-Galerkin methods for the direct solution of elliptic equations of Helmholtz and biharmonic type based on Legendre and Chebyshev polynomials. The first paper develops methods based on Legendre polynomials for a class of elliptic equations in one, two and three dimensions. The choice of Legendre polynomials can be motivated by the symmetry and sparsity properties of the resulting linear systems of equations. However, as Shen points out, there is yet no practical fast transform between physical and spectral space for the Legendre transform despite recent work on this problem [3], [7].

In his second paper, Shen develops the same algorithms based on Chebyshev polynomials. An important motivation for this is the availability of a fast transform (the FFT) between physical and spectral space. Thus, the methods of this class may be more attractive for situations where one repeatedly needs this change of basis, an example being a more complex nonlinear or time dependent problem where many right hand sides are needed.

We proceed to discretize our model partial differential equation using the spectral-Galerkin method based on Chebyshev polynomials, developed by Shen.

$$\begin{aligned} \Delta^2 u - \alpha \Delta u + \beta u &= f, \quad in \ \ \Omega = I^d, \\ u = \frac{\partial u}{\partial n} &= 0, \quad on \ \ \partial\Omega, \end{aligned} \tag{1}$$

where $I = (-1, 1)$ and $d = 1, 2, 3$. We restrict our discussion to the two dimensional

* Received by the editors February 1, 1996; accepted for publication Aug. 22, 1996.

† Institutt for Informatikk, University of Bergen, Høyteknologisenteret, N-5020 Bergen, Norway. http://www.ii.uib.no/~petter, Email: petter@ii.uib.no.

‡ Institutt for Informatikk, University of Bergen, Høyteknologisenteret, N-5020 Bergen, Norway. Email:bjornpt@ii.uib.no.

case where this gives rise to a linear system of equations with the following structure.

$$Mu^h \equiv (A\otimes B + 2C\otimes C + B\otimes A + \alpha(C\otimes B + B\otimes C) + \beta B\otimes B)u^h = f^h. \tag{2}$$

Shen [9] derived analytic expressions for all the nonzero elements of the matrices. We note that the matrix elements are indexed starting with subscript $k = 1$ in our expressions below.

$$\begin{aligned}
b_{kk} &= \frac{\pi}{2}(d_k + \frac{k^2 + 4(k+1)^2}{(k+2)^2}), \quad d_1 = 2, \;\; d_k = 1, \;\; k \geq 2 \\
b_{kk+2} &= -\pi(\frac{k+1}{k+2} + \frac{k(k+3)}{(k+2)(k+4)}) \\
b_{k+2k} &= b_{kk+2} \\
b_{kk+4} &= \frac{\pi}{2}\frac{k}{k+2} \\
b_{k+4k} &= b_{kk+4} \\
c_{kk} &= 4\pi\frac{k(k+1)^2}{k+2} \\
c_{kk-2} &= -2\pi(k-2)(k+1) \\
c_{kk+2} &= -2\pi k(k+1) \\
a_{kk} &= 8\pi k^2(k+1)(k+3) \\
a_{kj} &= \frac{8\pi k(k+1)}{j+2}((k-1)(k+3) + 3(j+1)^2), \quad j = k+2, k+4, \ldots
\end{aligned}$$

Shen observes that since $A^{-1}B$ and $A^{-1}C$ can not be made to commute by altering a finite number of rows, the capacitance matrix method that he used for the Legendre case may not be applied. He concludes that a special band elimination procedure with complexity $O(N^4)$ can be used to solve the system, but points out how unfortunate this is, since it leaves the Legendre method as the most efficient method for this problem, without any practical fast transform between the physical space and the spectral space.

In this paper an $O(N^3)$ method for solving (2) is presented, and it will be shown that in the case of many right hand sides the solution of (2) with a Chebyshev basis is preferable to the solution with a Legendre basis.

2. An $O(N^3)$ direct algorithm. Our proposed algorithm to solve (2) can be viewed as a direct application of the Sherman-Morrison formula rather than as a special capacitance matrix method. The capacitance matrix method used by Shen can also be viewed as a reordering of the rows of the matrices, followed by the use of the Sherman-Morrison formula.

The matrices A, B and C have alternating zero elements. A reordering by even and odd components splits A into two upper triangular matrices A_1 and A_2 of dimension $N/2$. Similarly, B and C split into two pentadiagonal and two tridiagonal matrices respectively. Note that this defines N slightly differently from Shen, our spectral discretization will have $N + 4$ points in each coordinate direction.

The matrix M from (2) can therefore be split into four parts:

$$M_1 = A_1\otimes B_1 + 2C_1\otimes C_1 + B_1\otimes A_1 + \alpha(C_1\otimes B_1 + B_1\otimes C_1) + \beta B_1\otimes B_1 \tag{3}$$

$$M_2 = A_1\otimes B_2 + 2C_1\otimes C_2 + B_1\otimes A_2 + \alpha(C_1\otimes B_2 + B_1\otimes C_2) + \beta B_1\otimes B_2 \tag{4}$$

$$M_3 = A_2\otimes B_1 + 2C_2\otimes C_1 + B_2\otimes A_1 + \alpha(C_2\otimes B_1 + B_2\otimes C_1) + \beta B_2\otimes B_1 \tag{5}$$

$$M_4 = A_2\otimes B_2 + 2C_2\otimes C_2 + B_2\otimes A_2 + \alpha(C_2\otimes B_2 + B_2\otimes C_2) + \beta B_2\otimes B_2 \tag{6}$$

We will proceed to study the system

$$M_2 u_2 = f_2, \tag{7}$$

the procedure being similar for the three other systems. In order to simplify the notation we do not subscript all variables. Note that most square matrices will have dimension $N/2$, we subscript the identity matrix with its dimension in order to guide the reader. Our first step is to multiply (7) with $(A_1^{-1}\otimes I_{\frac{N}{2}})$ producing the equivalent system $\hat{M}u = \hat{f}$, with $\hat{f} = (A_1^{-1}\otimes I_{\frac{N}{2}})f_2$ where

$$\begin{aligned}\hat{M} = (A_1^{-1}\otimes I_{\frac{N}{2}})M_2 &= I_{\frac{N}{2}}\otimes B_2 + 2A_1^{-1}C_1\otimes C_2 + A_1^{-1}B_1\otimes A_2 \\ &+ \alpha(A_1^{-1}C_1\otimes B_2 + A_1^{-1}B_1\otimes C_2) + \beta A_1^{-1}B_1\otimes B_2.\end{aligned} \tag{8}$$

The computation of $\hat{f}$ requires $O(N^2)$ flops since we can apply A_1^{-1} in $O(N)$ flops (see also the discussion in Section 2.1).

The coefficient matrix in (7) is not separable we therefore introduce a related separable matrix Z, defined by

$$\begin{aligned}Z &= I_{\frac{N}{2}}\otimes B_2 + 2A_1^{-1}C_1\otimes C_2 + (A_1^{-1}C_1)^2\otimes A_2 \\ &+ \alpha(A_1^{-1}C_1\otimes B_2 + (A_1^{-1}C_1)^2\otimes C_2) + \beta(A_1^{-1}C_1)^2\otimes B_2.\end{aligned} \tag{9}$$

The difference between the two matrices (8) and (9) is given by

$$\hat{M} - Z = Q_1\otimes G_2, \tag{10}$$

where $Q_1 = A_1^{-1}B_1 - (A_1^{-1}C_1)^2$ and $G_2 = A_2 + \alpha C_2 + \beta B_2$. It turns out that $A_1^{-1}B_1$ and $(A_1^{-1}C_1)^2$ differ only in the first row and the last column, a difference of rank 2. Q_1 can then be expressed as a matrix product $Q_1 = U_1V_1^T$ where both U_1 and V_1 are $\frac{N}{2}\times 2$ matrices.

The key to the success of our method is to exploit this and use the Sherman–Morrison formula to solve (8). The Sherman-Morrison formula expresses the inverse by

$$\hat{M}^{-1} = (Z + XY^T)^{-1} = Z^{-1}(I_{\frac{N}{2}} - X(I_N + Y^TZ^{-1}X)^{-1}Y^TZ^{-1}), \tag{11}$$

where $X = U_1\otimes G_2$ and $Y = V_1\otimes I_{\frac{N}{2}}$.

From (11) one sees that our problem is reduced to the solution of two linear systems of the form $Zx = b$ and $Px = b$, where $P = I_N + (V_1\otimes I_{\frac{N}{2}})^TZ^{-1}(U_1\otimes G_2)$, and two matrix vector multiplications of the form $y = (U_1\otimes G_2)x$ and $y = (V_1\otimes I_{\frac{N}{2}})^Tx$. We next show how to handle these problems.

2.1. Solution of auxiliary eigenvalue problems. We will use a matrix diagonalization technique that requires the solution of two generalized eigenvalue problems. It is known that these (unsymmetric) problems have positive, real eigenvalues [6]. Consider first the problem

$$C_1 E_1 = A_1 E_1 \Lambda_1. \tag{12}$$

The matrix $A_1^{-1}C_1$ is an upper Hessenberg matrix, thus Λ_1 can be found directly by QR iteration, this requires about $5N^3/4$ flops (page 380 in [5]). However, using more specialized techniques as we outline below, one can compute the eigenvalues Λ_1 in only $O(N^2)$ work. For each known eigenvalue $\lambda_1^{(k)}$, $k = 1, \ldots, N/2$ we can find the corresponding eigenvector e_k by inverse iteration,

$$(C_1 - \lambda_1^{(k)} A_1) e_k^{(j)} = A_1 e_k^{(j-1)}. \tag{13}$$

Due to the special structure of A_1 ($(A_1 + A_1^T)/2$ is a rank two modification of the diagonal of A_1), one can LU factor any nonsingular $A + \mu B + \sigma C$ in only $O(N)$ work for arbitrary constants μ and σ (see also [9]). This LU factorization requires $27N$ flops while a complete back substitution costs $16N$ operations.

Applying this technique to (13) we see that one iteration only costs $O(N)$ flops. For each eigenvector only two iterations are needed, making the cost of finding E_1 an $O(N^2)$ operation.

Also note that if W^T are the left eigenvectors of (12) then $E_1^{-1} = W^T A_1$. The special structure of A_1 enables us to perform this multiplication in $O(N^2)$ operations instead of using the $O(N^3)$ method based on LU factorization of E_1. Clearly, W^T can be computed in $O(N^2)$ work by inverse iteration.

It remains to show how one can compute the eigenvalues in $O(N^2)$ work. We proceed using a method inspired by the work in [4]. First, we compute an initial approximation to the eigenvalues based on the tridiagonal matrix

$$T_1 = \mathrm{diag}(A_1)^{-1} C_1.$$

This matrix is similar (via a diagonal scaling) to a symmetric matrix and all its eigenvectors and eigenvalues can be computed in $O(N^2)$. The approximations to the eigenvectors are further refined by two steps of inverse iteration on the system (13). We then perform a Newton iteration defined by

$$\begin{pmatrix} C_1 - \lambda^{(k)} A_1 & A_1 x \\ e & 0 \end{pmatrix} \begin{pmatrix} \delta x \\ \delta\lambda \end{pmatrix} = \begin{pmatrix} A_1 r \\ 0 \end{pmatrix}, \tag{14}$$

computing the updates δx and $\delta\lambda$ to an eigenvector and its eigenvalue. Here $r = C_1 x - \lambda^{(k)} A_1 x$ is the residual vector and e is a unit vector. The linear system (14) in the Newton step is a rank two modification of the system (13) extended by one row and one column. It can therefore be factored and solved in $O(N)$ work by using the Sherman-Morrison formula. A fixed number of Newton steps are needed, and we therefore obtain all eigenvectors and eigenvalues in $O(N^2)$ operations.

To solve $M_3 u_3 = f_3$ and $M_4 u_4 = f_4$, the eigenvalue problem $C_2 E_2 = A_2 E_2 \Lambda_2$ must be handled in a similar way. Altogether, we must solve two eigenvalue problems each of size $N/2$ and find N additional (left) eigenvectors by inverse iteration. We conclude that all auxiliary eigenvalue problems can be computed in $O(N^2)$ work.

2.2. Solution of $Zx = b$. We first proceed to transform $Zx = b$ to the block diagonal form

$$\begin{aligned} \hat{Z} &= (E_1^{-1} \otimes I_{\frac{N}{2}}) Z (E_1 \otimes I_{\frac{N}{2}}) \\ &= I_{\frac{N}{2}} \otimes B_2 + 2\Lambda_1 \otimes C_2 + \Lambda_1^2 \otimes A_2 + \alpha(\Lambda_1 \otimes B_2 + \Lambda_1^2 \otimes C_2) + \beta \Lambda_1^2 \otimes B_2. \end{aligned} \tag{15}$$

When solving $\hat{Z}\hat{x} = \hat{b}$, the cost of computing $\hat{b} = (E_1^{-1} \otimes I_{\frac{N}{2}})b$ is $N^3/4$. We also need $x = (E_1 \otimes I_{\frac{N}{2}})\hat{x}$ when transforming back, the total cost of these transforms is therefore $N^3/2$.

Each block $\hat{Z}^{(k)},\ k = 1, \ldots \frac{N}{2}$ on the diagonal of $\hat{Z}$ is of the form

$$\hat{Z}^{(k)} = {\lambda_1^{(k)}}^2 A_2 + (1 + \alpha \lambda_1^{(k)} + \beta {\lambda_1^{(k)}}^2) B_2 + (2\lambda_1^{(k)} + \alpha {\lambda_1^{(k)}}^2) C_2.$$

As explained above we can both factor and solve with such a block in $O(N)$ flops, thus the inversion of $\hat{Z}$ requires only $O(N^2)$ flops.

From (11) and the preceding paragraph we can write

$$\hat{M}^{-1}\hat{f} = (E_1 \otimes I_{\frac{N}{2}})\hat{Z}^{-1}(\bar{f} - (J_1 \otimes G_2)P^{-1}(F_1 \otimes I_{\frac{N}{2}})^T \hat{Z}^{-1}\bar{f}),$$

where $\bar{f} = (E_1^{-1} \otimes I_{\frac{N}{2}})\hat{f}$, $J_i = E_i^{-1}U_i$, and $F_i = E_i^T V_i$, $i = 1, 2$. The computation of F_1 and J_1 is $O(N^2)$ work and thereafter we can multiply $(F_1 \otimes I_{\frac{N}{2}})^T$ and $(J_1 \otimes G_2)$ by a vector in $O(N^2)$ flops. Therefore, if we know the LU factorization of P, we can solve (2) in only $2N^3 + O(N^2)$ flops. It remains to estimate the work to build and factor P.

2.3. Assembly and the direct solution of $Px = b$. The matrix P can be assembled in a similar way to that of Bjørstad [1], [2], who assembles the corresponding matrix in the finite difference case. The spectral matrices in this work are denser. Similar to Bjørstad we also cover the general problem with nonzero parameters β and α.

We will first show how to form the matrix

$$P = I_N + (F_1 \otimes I_{\frac{N}{2}})^T \hat{Z}^{-1}(J_1 \otimes I_{\frac{N}{2}})(I_2 \otimes G_2).$$

We can write P more explicitly as

$$P = I_N + \begin{pmatrix} \sum_{k=1}^{\frac{N}{2}} (F_1)_{1k}(J_1)_{k1}\hat{Z}^{(k)^{-1}} & \sum_{k=1}^{\frac{N}{2}} (F_1)_{1k}(J_1)_{k2}\hat{Z}^{(k)^{-1}} \\ \sum_{k=1}^{\frac{N}{2}} (F_1)_{2k}(J_1)_{k1}\hat{Z}^{(k)^{-1}} & \sum_{k=1}^{\frac{N}{2}} (F_1)_{2k}(J_1)_{k2}\hat{Z}^{(k)^{-1}} \end{pmatrix} (I_2 \otimes G_2).$$

Since we already have computed the LU factorization of $\hat{Z}^{(k)}$, we explicitly compute one $\hat{Z}^{(k)^{-1}}$ in $4N^2$ flops and all the inverses in $2N^3$ flops. The multiplication with G_2

can be done in $O(N^2)$ work, the four summations require $7N^3/8$ work and the LU factorization of P additionally $2N^3/3$ flops.

There is, however, an even faster algorithm for computing the initial solution exploiting more structure in $\hat{Z}^{-1}$. First, write $R = (I_{\frac{N}{2}} \otimes A_2^{-1})Z$ and similarly, define $\hat{R} = (I_{\frac{N}{2}} \otimes A_2^{-1})\hat{Z}$. In order to compute $\hat{R}^{-1}$ we introduce

$$\begin{aligned} T &= I_{\frac{N}{2}} \otimes (A_2^{-1}C_2)^2 + 2A_1^{-1}C_1 \otimes A_2^{-1}C_2 + (A_1^{-1}C_1)^2 \otimes I_{\frac{N}{2}} \\ &+ \alpha(A_1^{-1}C_1 \otimes (A_2^{-1}C_2)^2 + (A_1^{-1}C_1)^2 \otimes A_2^{-1}C_2) \\ &+ \beta(A_1^{-1}C_1)^2 \otimes (A_2^{-1}C_2)^2 \end{aligned}$$

and

$$\hat{T} = (E_1^{-1} \otimes I_{\frac{N}{2}})T(E_1 \otimes I_{\frac{N}{2}}).$$

The difference between R and T is

$$R - T = A_1^{-1}\hat{G}_1 \otimes Q_2 = (A_1^{-1} \otimes I)(\hat{G}_1 \otimes U_2 V_2^T), \tag{16}$$

where $\hat{G}_1 = A_1 + \alpha C_1 + \beta C_1 {A_1}^{-1} C_1$.

The difference between two blocks $\hat{R}^{(k)}$ and $\hat{T}^{(k)}$ is

$$\hat{R}^{(k)} - \hat{T}^{(k)} = (1 + \alpha\lambda_1^{(k)} + \beta\lambda_1^{(k)^2})U_2 V_2^T \equiv \delta_k U_2 V_2^T,$$

where we have defined $\delta_k = 1 + \alpha\lambda_1^{(k)} + \beta\lambda_1^{(k)^2}$.

By using Sherman-Morrison again, we explicitly invert $\hat{R}^{(k)}$

$$\hat{R}^{(k)^{-1}} = \hat{T}^{(k)^{-1}}(I_{\frac{N}{2}} - \delta_k U_2(I_2 + \delta_k V_2^T \hat{T}^{(k)^{-1}} U_2)^{-1} V_2^T \hat{T}^{(k)^{-1}}).$$

We transform $\hat{T}^{(k)}$ to diagonal form by solving the generalized eigenvalue problem

$$C_2 E_2 = A_2 E_2 \Lambda_2,$$

considered in Section 2.1. It follows that

$$E_2^{-1}\hat{T}^{(k)}E_2 = \delta_k \Lambda_2^2 + \lambda_1^{(k)^2}(I_{\frac{N}{2}} + \alpha\Lambda_2) + 2\lambda_1^{(k)}\Lambda_2 \equiv D^{(k)},$$

a diagonal matrix. Using this we have

$$\hat{R}^{(k)^{-1}} = E_2 S^{(k)^{-1}} E_2^{-1}$$

where the block $S^{(k)^{-1}}$ is

$$\begin{aligned} S^{(k)^{-1}} &= D^{(k)^{-1}}(I_{\frac{N}{2}} - \delta_k E_2^{-1} U_2 (I_2 + \delta_k V_2^T E_2 D^{(k)^{-1}} E_2^{-1} U_2)^{-1} V_2^T E_2 D^{(k)^{-1}}) \\ &= D^{(k)^{-1}}(I_{\frac{N}{2}} - \delta_k J_2 \hat{P}^{(k)^{-1}} F_2^T D^{(k)^{-1}}), \end{aligned}$$

with $\hat{P}^{(k)} = I_2 + \delta_k F_2^T D^{(k)^{-1}} J_2$. Careful counting reveals that we can compute a block $S^{(k)^{-1}}$ in $3N^2/4 + O(N)$ flops, and therefore all the blocks in $3N^3/8 + O(N^2)$ operations. We now write P as

$$\begin{aligned} P &= (I_2 \otimes E_2)[I_2 \otimes E_2^{-1}G_2^{-1}A_2E_2 \\ &+ \begin{pmatrix} \sum_{k=1}^{\frac{N}{2}} (F_2)_{1k}(J_2)_{k1} S^{(k)^{-1}} & \sum_{k=1}^{\frac{N}{2}} (F_2)_{1k}(J_2)_{k2} S^{(k)^{-1}} \\ \sum_{k=1}^{\frac{N}{2}} (F_2)_{2k}(J_2)_{k1} S^{(k)^{-1}} & \sum_{k=1}^{\frac{N}{2}} (F_2)_{2k}(J_2)_{k2} S^{(k)^{-1}} \end{pmatrix}] (I_2 \otimes E_2^{-1}A_2^{-1}G_2). \end{aligned}$$

We explicitly compute and LU factor the middle (second) factor in P. Fortunately, we can compute the first term in only $O(N^2)$ work,

$$E_2^{-1}G_2^{-1}A_2E_2 = (I_{\frac{N}{2}} + \alpha\Lambda_2 + \beta E_2^{-1}A_2^{-1}B_2E_2)^{-1}.$$

But $A_2^{-1}B_2 = (A_2^{-1}C_2)^2 + U_2V_2^T$ and therfore

$$E_2^{-1}G_2^{-1}A_2E_2 = (D_2 + \beta J_2 F_2^T)^{-1},$$

with $D_2 = I_{\frac{N}{2}} + \alpha\Lambda_2 + \beta\Lambda_2^2$. It is now easy to see that this term can be explicitly computed in $O(N^2)$ work by using the Sherman-Morrison formula. The summations in the second term of the factor require $7N^3/8$ flops as before (since we can scale one of the 4 matrices). When adding the terms for all four systems we get a total cost of $23N^3/3 + O(N^2)$ for assembly and factorization.

We now write

$$P^{-1} = (I_2 \otimes G_2^{-1}A_2E_2)U^{-1}L^{-1}(I_2 \otimes E_2^{-1}),$$

and note that each factor can be multiplied by a vector in $O(N^2)$ work.

With this algorithm the first solution of (2) can be computed in $29N^3/3 + O(N^2)$ flops, while subsequent problems require $2N^3 + O(N^2)$ arithmetic operations.

2.4. Iterative solution of $Px = b$. In order to develop effective preconditioners for P we use ideas similar to the work [2] for the finite difference case. In our case we have non-symmetric matrices. We proceed to build a preconditioner by replacing components in P with matrices from the separable problem that we have used in the construction of our direct method. That is, given our matrix

$$P = I_N + (V_1 \otimes I_{\frac{N}{2}})^T R^{-1}(I_{\frac{N}{2}} \otimes A_2^{-1})(U_1 \otimes G_2),$$

we replace R by T and G_2 by $\hat{G}_2$ obtaining a preconditioner given by

$$\bar{P} = I_N + (V_1 \otimes I_{\frac{N}{2}})^T T^{-1}(I_{\frac{N}{2}} \otimes A_2^{-1})(U_1 \otimes \hat{G}_2).$$

A careful examination of P and $\bar{P}$ shows that we can form the matrix vector product $y = Px$ and solve the linear system $\bar{P}x = r$ with $10N^2$ operations. The eigenvalues of $\bar{P}^{-1}P$ cluster at 1 and only a very small number of eigenvalues have smaller values. To illustrate this we plot the 30 smallest eigenvalues from the eigenvalue distribution

for the system M_1 (the other systems are slightly better) with $\alpha = \beta = 0$ for three different values of N in Figure 1. Based on this data we can expect an appropriate Krylov subspace method to work extremely well. We have used BiCGSTAB [10] in our actual implementation. In this way we avoid the entire setup cost associated with the computation of P. Thus, with a fixed number of iterations our overall iterative scheme for the Chebyshev system (2) will have a cost of $2N^3 + O(N^2)$.

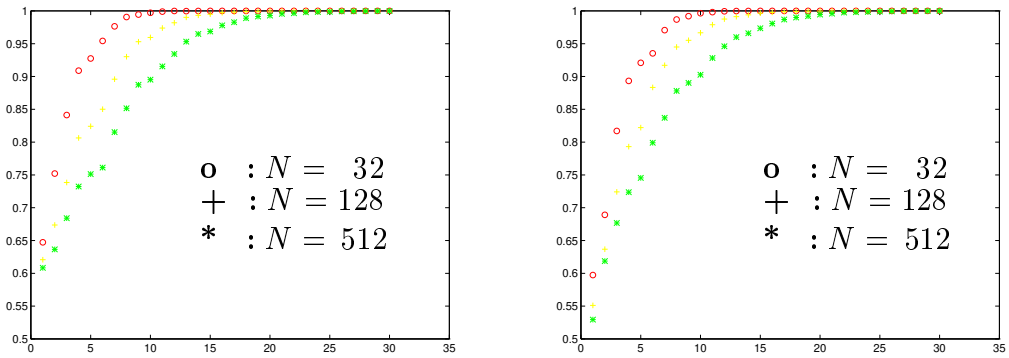

FIG. 1. *The distribution of eigenvalues of $\bar{P}^{-1}P$ for the Legendre-Galerkin (left) and the Chebyshev-Galerkin (right).*

3. The Legendre-Galerkin method. In the Legendre-Galerkin case we solve (2) in the same way as described here for the Chebyshev-Galerkin method, although the matrices involved differ. In this case we have that $C_i = C_i^T$ and $A_i = I,\ i = 1, 2$, thus the eigenvalue problems can be solved straightforwardly in $O(N^2)$ work. Also the matrices Q_i have only two nonzero entries, it is therefore possible to choose $U_i = V_i,\ i = 1, 2$. The symmetries in the Legendre case reduces the number of summations from four to three and the blocks $S^{(k)^{-1}}$ are symmetric. We can factor $G_2 = L_2 L_2^T$ and write (10) as

$$M - Z = (U_1 \otimes L_2)(U_1 \otimes L_2)^T. \tag{17}$$

The matrix P becomes

$$P = I_N + (U_1 \otimes L_2)^T Z^{-1}(U_1 \otimes L_2),$$

a symmetric matrix. Thus, we can apply symmetric factorizations (Cholesky in the positive definite case) and generally only compute the symmetric half of all matrices that occur. With these savings we find a total cost for solving (2) to be $16N^3/3 + O(N^2)$ flops.

We next describe the iterative solution technique in the Legendre-Galerkin case. The factor $\hat{G}_i$ from (16) takes the form $I_{\frac{N}{2}} + \alpha C_i + \beta C_i^2$, let its symmetric factorization be $\hat{L}_i \hat{L}_i^T\ i = 1, 2$. Define the symmetric preconditioner

$$\bar{P} = I_N + (U_1 \otimes \hat{L}_2)^T T^{-1}(U_1 \otimes \hat{L}_2).$$

We expect $\bar{P}^{-1}$ to be a good preconditioning matrix for P. Numerical experiments, see Figure 1 and Table 1, confirm this and indicate that the preconditioned system

TABLE 1
Condition numbers for the Legendre-Galerkin algorithm

N	32	128	512
Cond(P)	22.46	274.9	4072
Cond$(\bar{P}^{-1}P)$	1.49	1.58	1.63

TABLE 2
Algorithm comparison for large N, $N = 726$

	First solution				Second solution			
	L-dir, C-dir		L-it, C-it		L-dir, C-dir		L-it, C-it	
Machine BLAS	113	213	51	34	43	21	43	24
Fortran BLAS	308	234	236	39	226	24	226	29

has a uniformly bounded condition number independent of N. In Table 1 we consider the system M_1 with $\alpha = \beta = 0$. The other systems showed slightly better condition numbers. A conjugate gradient iteration can therefore be used to solve $Px = b$ in $O(N^2)$ flops. Again, the resulting iterative method for (2) costs only $2N^3 + O(N^2)$ flops.

We note that an $O(N^2)$ preconditioned iterative method for the system (15) would enable us to design a Chebyshev-Galerkin algorithm with overall complexity $O(N^2 \log N)$. This idea will not be pursued here, but be reported elsewhere.

4. Multiple right hand sides. In the Chebyshev-Galerkin case one has a fast transform between physical and spectral space, which is not practical in the Legendre case. Therefore the Chebyshev-Galerkin method may be useful for instance for nonlinear equations and eigenvalue problems or other problems where there is a need for several right hand sides. We will now include the cost of transforming (both ways) between physical and spectral space. This added cost is $O(N^2 \log N)$ for the Chebyshev method and $8N^3$ in the Legendre case. For the first right-hand side the total cost of solving (2) is therefore $29N^3/3 + O(N^2 \log N)$ in the Chebyshev case. A direct solver for the Legendre case is $40N^3/3 + O(N^2)$, while the two iterative methods have $2N^3 + O(N^2 \log N)$ and $10N^3 + O(N^2)$ work respectively.

For each subsequent right-hand side the cost for both Chebyshev algorithms is $2N^3 + O(N^2 \log N)$ to be compared with $10N^3 + O(N^2)$ in the Legendre case. So for several right hand-sides and large N the Chebyshev algorithms are asymptotically five times faster than the Legendre-Galerkin algorithms.

5. Numerical Example. In order to illustrate our methods we solve the equation (2) using the same example as in Shen [8].

$$\Delta^2 u = 128\pi^4(\cos(4\pi x)\cos(4\pi y) - \cos(4\pi x)\sin(2\pi y)^2 - \cos(4\pi y)\sin(2\pi x)^2),$$

in $\Omega = I \times I$, with $I = (-1, 1)$, and the homogeneous boundary conditions $u = \frac{\partial u}{\partial n} = 0$. The exact solution of this problem is $u(x, y) = (\sin(2\pi x)\sin(2\pi y))^2$. We note that $\beta = \alpha = 0$ in this case, but that our algorithms are completely general and take no

advantage of this special situation. A good implementation should obviously check for this condition. Perhaps equally important, due to symmetries only the problem M_1 has a nontrivial solution. Again, our algorithms as reported here, do not check for this condition, except for the iterative algorithms that will converge in one iteration for three of the four systems. This situation will present a slightly more favorable picture for the iterative algorithms, but tests where more systems are nontrivial have shown that the relative merits of the algorithms remain the same.

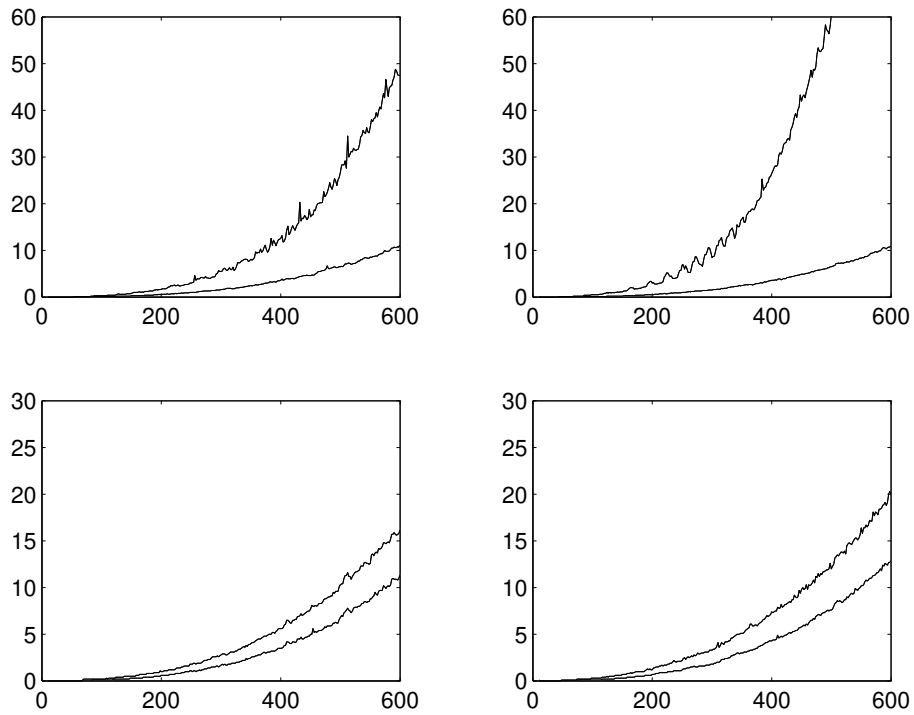

FIG. 2. *Comparison of the four algorithms. The first row shows the direct methods, the second row the iterative methods. The first column is the Legendre algorithm, while the second column shows the Chebyshev algorithm. All plots show the time in seconds as a function of problem size for an initial solution and then for a subsequent solution. Note the different time scales in the two rows. Also note that these times* **do not** *include the transform time between physical and spectral space.*

Our algorithms are all written in Fortran 90, making extensive use of the array notation. Thus, it should be relatively simple to add HPF directives for possible parallel execution, but we have not tested this yet. All computer runs were performed on a DEC Alpha 2000 running at 233MHz. We have used the BLAS supplied by the DEC DXML library in the test that we report on. The times reported are elapsed times on a very lightly loaded system.

First, in Table 2 we check our complexity estimates for a large value of N, $N = 726$. We readily observe that the Legendre based algorithms are very sensitive to a good machine implementation of the BLAS. The difference is about 190 seconds, the tuned DGEMM from the DEC library is about ten times faster than the corresponding Fortran version. The FFT transforms in the Chebyshev algorithm take about 2 seconds in this case. Thus, the Legendre algorithms perform somewhat better than predicted due to

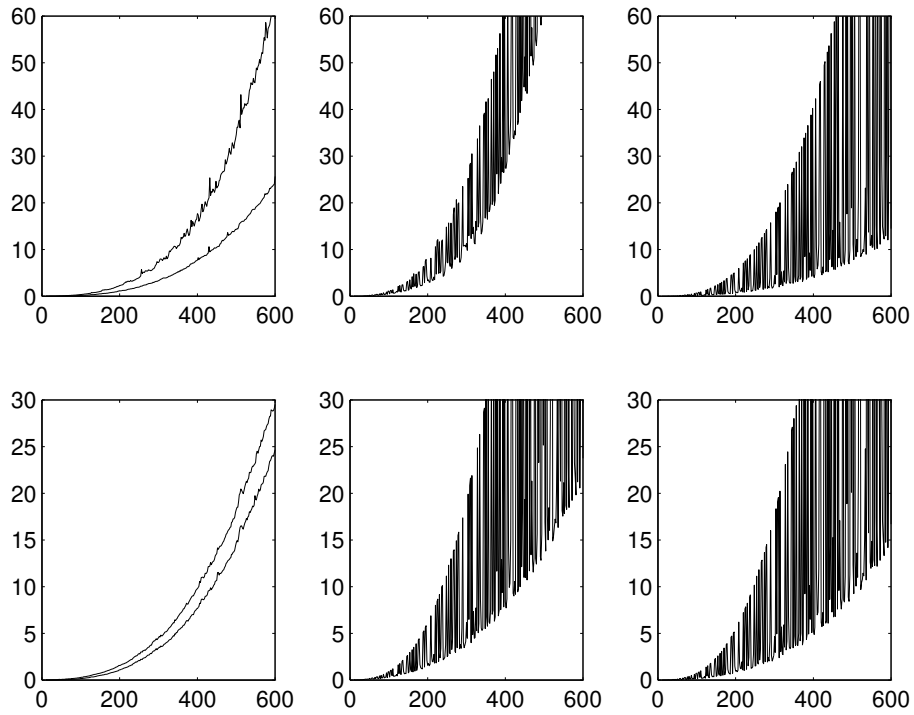

FIG. 3. *Comparison of the four algorithms. The times are complete elapsed times including the necessary transforms between spectral and physical space. The first row shows the direct algorithms, the second row the iterative algorithms. The first column is the Legendre method, the two curves show the time in seconds as a function of the problem size for an initial solution and then for a subsequent solution. The second column shows the range of times for the Chebyshev method for an initial solution, while the last column shows the same data for a second solution. Note the different time scales in the two rows.*

the very efficient implementation of matrix multiplication.

Comparing the two direct algorithms using the Fortran BLAS shows a time ratio of 0.76 between Chebyshev and Legendre for an initial solution, in good agreement with 0.75 as predicted by theory. A similar comparison of the iterative methods puts the Chebyshev algorithm ahead by a factor of 6 to 9. In this case Chebyshev is faster than Legendre also when using the machine BLAS. The situation is similar when solving for a second right hand side. In this case Chebyshev has an even bigger advantage with all code in Fortran, and it is about twice as fast in the tuned version. We also note that there are very small differences between the direct and the iterative algorithms in this case. Thus, for large N the best overall algorithm is clearly the iterative solution of the Chebyshev system.

In Figure 2 we show the performance of our four algorithms. This figure does not include the transformation time between physical and spectral space, but it does include the transformations from the standard spectral basis (Chebyshev or Legendre) to the special basis proposed by Shen. Besides showing the performance of our algorithms, these numbers are relevant in situations where several problems can be solved without the transformations between physical and spectral space. We observe that a first solution is almost twice as expensive with the Chebyshev method, in good agreement with

the predicted factor of 1.8. Also, the iterative algorithms are more than three times faster than the direct methods. The Legendre algorithms are faster than Chebyshev, but the difference is smaller than for the direct methods. For repeated right hand sides when we can use precomputed quantities, the direct and iterative algorithms are virtually identical in performance, with the Legendre methods being slightly faster than Chebyshev.

In Figure 3 we include the necessary transforms (both ways) between spectral and physical space. The performance of our FFT routine depends on the prime factorization of $N + 3$. We show the full range of performance, but most applications would clearly pick a suitable value for N, so it is quite reasonable to just consider the lower envelope of the curves given. In this case, the FFT takes less than 2 seconds and therefore cannot make up for the larger setup cost which is due to the lack of symmetry in the Chebyshev algorithm. For subsequent problems the Chebyshev method is about twice as fast as the Legendre method. We can conclude that the advantage of the Chebyshev transform when solving a sequence of problems is at most a factor two for practical problems.

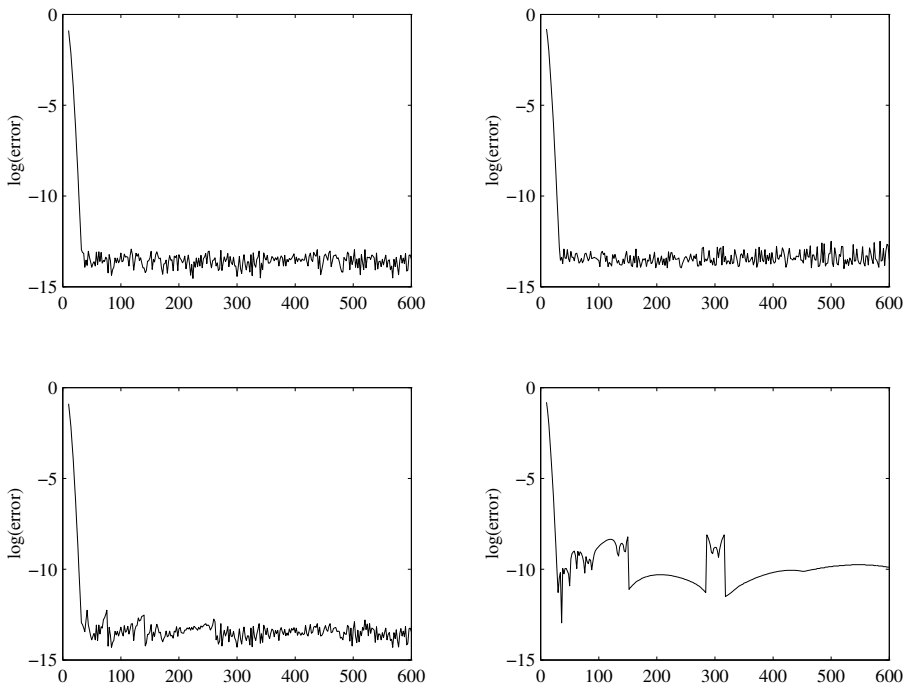

FIG. 4. *Comparison of the four algorithms on a single problem. The plots show the error for the four algorithms. The first row is the direct algorithms, the second row the two iterative algorithms. The first column shows the Legendre method, the second column is Chebyshev. The iterative algorithms both had a residual 2-norm less than* 10^{-10} *as stopping criteria.*

When we compare our iterative methods in the lower half of Figure 3, there is at most a five second difference between a first and a second solution. The Chebyshev algorithm is almost a factor two faster as long as $N+3$ has relatively many small prime factors.

Finally, in Figure 4 we compare the accuracy obtained by the four different algorithms. The exponential convergence is evident in all cases. Thereafter, the accuracy

is very satisfactory for all methods. Both methods show excellent performance with no increase in the error over the large range of N tested. The iterative methods both declared convergence when the 2-norm of the residual dropped below 10^{-10}. This is clearly reflected in the somewhat irregular error oscillating around 10^{-10} for the Chebyshev method. The error plotted in Figure 4 is the maximum pointwise error.

We may assert that our algorithms appear to be stable and that we have solved a very large problem without being severely affected by the growth of the condition number. All four algorithms are available from netlib.

Acknowledgment. We thank Jie Shen for providing the Legendre and Chebyshev transform software used in this project.

REFERENCES

[1] P. E. Bjørstad, *Numerical solution of the biharmonic equation*, PhD thesis, Stanford University, Stanford, CA, 1980.

[2] ——, *Fast numerical solution of the biharmonic dirichlet problem on rectangles*, SIAM J. Numer. Anal., 20 (1983), pp. 59–71.

[3] J. P. Boyd, *Multipole expansions and pseudospectral cardinal functions: A new generation of the fast Fourier transform*, J. Comput. Phys., 103 (1992), pp. 184–186.

[4] J. J. Dongarra, G. A. Geist, and C. H. Romine, *Fortran subroutines for computing the eigenvalues and eigenvectors of a general matrix by reduction to general tridiagonal form*, Transactions on Mathematical Software, 18 (1992), pp. 392–400. Algorithm 710.

[5] G. H. Golub and C. F. V. Loan, *Matrix Computations*, Johns Hopkins Univ. Press, 1989. Second Edition.

[6] D. Gottlieb and L. Lustman, *The spectrum of the chebychev collocation operator for the heat equation*, SIAM J. Numer. Anal., 20 (1983), pp. 909–921.

[7] S. A. Orszag, *Fast eigenfunction transforms*, Science and Computers: Advances in mathematics supplementary studies, 10 (1986), pp. 23–30.

[8] J. Shen, *Efficient spectral-Galerkin method I. direct solvers of second- and fourth-order equations using Legendre polynomials*, SIAM J. Sci. Comput., 15 (1994), pp. 1440–1451.

[9] ——, *Efficient spectral-Galerkin method II. direct solvers of second- and fourth-order equations using Chebyshev polynomials*, SIAM J. Sci. Comput., 16 (1995), pp. 74–87.

[10] H. A. van der Vorst, *Bi-CGSTAB: A fast and smootly converging variant of Bi-CG for solution of non-symmetric limear systems*, SIAM J. Sci. Statist. Comput., 13 (1992).