Comput. Math. Applic. Vol. 13, No. 8, pp. 681–686, 1987
Printed in Great Britain.

0097-4943/87 $3.00 + 0.00

A PARALLEL PERTURBED BIHARMONIC SOLVER

G. LOTTI
Dipartimento di Informatica, Università di Pisa, Corso Italia 40, 56100 Pisa, Italia

(*Received* 16 *June* 1986)

Communicated by E. Y. Rodin

Abstract—An algorithm for the solution of the finite difference approximation of the perturbed biharmonic problem is proposed which consists of both direct and iterative stages. The optimality of this algorithm is proved under the opportune hypothesis on the perturbation.

1. INTRODUCTION

In solving non linear or time-dependent problems we frequently have to solve equations of the type

$$(T + a^2 I)\mathbf{u} = \mathbf{v}, \quad a \in \mathbb{R}, \tag{1}$$

where $T = M$ or $T = N$, M and N being the finite difference matrices respectively of the Poisson and of the biharmonic equation on an $n \times n$ grid with square mesh.

The parallel solution of such systems can be obtained by using direct algorithms similar to those proposed in Ref. [1] requiring $O(\log n)$ and $O(n \log n)$ time steps respectively when $T = M$ and $T = N$, and a number of processors $O(n^2)$.

In Ref. [2] an iterative method has been shown (parallel Gauss algorithm) for the LDU factorization and the solution of tridiagonal and block-tridiagonal systems.

It is possible to show that the combined use of these direct and iterative algorithms produces two algorithms to solve equation (1), for sufficiently large values of a, requiring $O(n^2)$ processors and $O(\log n)$, $O(n)$ time steps respectively when $T = M$ and $T = N$. Since the fastest available sequential algorithm for equation (1) when $T = N$ requires $O(n^3)$ operations [3] [this algorithm uses a preprocessing phase requiring $O(n^3)$ operations] then the resulting speed up and efficiency of our "perturbed" biharmonic solver are $S(p) = O(n^2)$ and $E(p) = O(1)$.

2. THE PARALLEL GAUSS ALGORITHM

The parallel Gauss (PG) algorithm for the $n \times n$ Toeplitz tridiagonal linear system $A\mathbf{x} = \mathbf{b}$, with

$$A = \begin{bmatrix} a & -1 & & & \\ -1 & a & -1 & & \\ & \cdots & \cdots & \cdots & \\ & & -1 & a & -1 \\ & & & -1 & a \end{bmatrix}, \quad a \in \mathbb{C}, \tag{2}$$

is essentially based on the classical Gaussian elimination algorithm:

1. Factor $A = (I + L)D(I + L^{\mathrm{T}})$,
   where $D = \mathrm{diag}(d_1, d_2, \ldots, d_n)$, $L = \mathrm{subdiag}(l_2, l_3, \ldots, l_n)$.
2. Solve $(I + L)\mathbf{f} = \mathbf{b}$;
3. Solve $(I + L^{\mathrm{T}})\mathbf{x} = \mathbf{g}$, where $\mathbf{g} = D^{-1}\mathbf{f}$.
   Let $d_1 = a$, then $d_i = a - d_{i-1}^{-1}$, $l_i = -1/d_{i-1}$, $i = 1, 2, \ldots, n$.

The PG algorithm is carried out by the following three iterative methods:

1. $d_1^k = a$, $d_i^0 = a$, $d_i^k = a - (d_{i-1}^{k-1})^{-1}$, $k = 1, \ldots, ID$.

Define $\tilde{l}_j = -1/d_{j-1}^{(ID)}$, $j = 2, 3, \ldots, n-1$, then

2. $\mathbf{f}^0 = \mathbf{b}$, $\mathbf{f}^k = \mathbf{b} - \tilde{L}\mathbf{f}^{k-1}$, $k = 1, \ldots, IF$.

Define $\tilde{g}_j = f_j^{(IF)}/d_j^{(ID)}$, $j = 1, 2, \ldots, n$, then

3. $\mathbf{x}^0 = \tilde{\mathbf{g}}$, $\mathbf{x}^k = \tilde{\mathbf{g}} - \tilde{L}^T\mathbf{x}^{k-1}$, $k = 1, \ldots, IX$;

where the iteration limit *ID*, *IF*, *IX* may be fixed in advance or determined automatically. Note that, because of the special structure of matrix A, at the kth step of the iterative method 1, only the new value of the $(k+1)$th entry has to be computed, whereas the iterative methods 2 and 3 require $O(n)$ processors.

Let $l = 4|a^{-1}|^2$, then it is possible to prove the following.

Lemma 2.1

Let $l \leqslant 1$, then A is nonsingular, $|ad_i^{-1}| < 2/(1+\sqrt{1-l})$ and $|1 - a^{-1}d_i| < (1-\sqrt{1-l})/2 \quad \forall\, i$. Furthermore, if $|1 - a^{-1}d_i^0| < (1-\sqrt{1-l})/2$ and $d_1^k = a \quad \forall\, i$ then $|a(d_i^k)^{-1}| < 2/(1+\sqrt{1-l})$ and $|1 - a^{-1}d_i^k| < (1-\sqrt{1-l})/2 \quad \forall\, i$ and k.

Proof

The bounds can be proved by using the same technique as used in Ref. [2] for the block PG algorithm. ■

Under the same hypothesis of Lemma 2.1 it is possible to prove that $\|D^k - D\| < l/(1+\sqrt{1-l})^2 \|D^{k-1} - D\|$. Moreover, we have

$$|d_i - d_{i-1}| = |d_{i-2}^{-1} d_{i-1}^{-1}|\,|d_{i-1} - d_{i-2}| < l/(1+\sqrt{1+l})^2 |d_{i-1} - d_{i-2}|,$$

and a similar bound holds for the iterates d^k, i.e.

$$|d_i^k - d_{i-1}^k| < l/(1+\sqrt{1-l})^2 |d_{i-1}^{k-1} - d_{i-2}^{k-1}|.$$

By using the same technique shown in Ref. [2] it is possible to prove the following.

Theorem 2.1

Assume $l \leqslant 1$, then for PG

$$\|D^k - D\| < l/(1+\sqrt{1-l})^2 \|D^{k-1} - D\|, \qquad k = 2, \ldots, ID,$$

$$\|\mathbf{f}^k - \mathbf{f}\| < \sqrt{l}/(1+\sqrt{1-l}) \|\mathbf{f}^{k-1} - \mathbf{f}\| + d_{\mathbf{f}}, \quad k = 2, \ldots, IF,$$

and

$$\|\mathbf{x}^k - \mathbf{x}\| < \sqrt{l}/(1+\sqrt{1-l}) \|\mathbf{x}^{k-1} - \mathbf{x}\| + d_{\mathbf{x}}, \quad k = 2, \ldots, IX,$$

where $d_{\mathbf{f}}$ and $d_{\mathbf{x}}$ depend upon the previous steps and can be made arbitrarily small.

In Ref. [2] an estimation is given for the total number of iterations required to reduce the initial error $\mathbf{e}^0$ by a constant factor 2^{-b}. Moreover, numerical experiments giving results which are consistent with the theoretical estimates are shown.

The theoretical analysis of the PG algorithm is carried out by assuming that the D and $\mathbf{f}$ iterations are performed long enough so that the quantities $d_{\mathbf{f}}$ and $d_{\mathbf{x}}$ which occur in Theorem 2.1 are negligible, i.e. the following relations are assumed to hold:

$$\|D^k - D\| < [l/(1+\sqrt{1-l})^2]^k \|D^0 - D\|,$$

$$\|\mathbf{f}^k - \mathbf{f}\| < [\sqrt{l}/(1+\sqrt{1-l})]^k \|\mathbf{f}^0 - \mathbf{f}\|,$$

$$\|\mathbf{x}^k - \mathbf{x}\| < [\sqrt{l}/(1+\sqrt{1-l})]^k \|\mathbf{x}^0 - \mathbf{x}\|.$$

3. A PERTURBED BIHARMONIC SOLVER

Let us consider the perturbed biharmonic equation.

$$\nabla 4u + d^2 u = f \quad \text{in} \quad R, \quad d \in \mathbb{R}, \tag{3}$$

where R is the unit square with the boundary conditions $u(x, y) = 0$ and $u_n(x, y) = 0$ for $(x, y) \in \delta R$, and $u_n(x, y)$ is the outward normal derivative at the boundary. Superimposing a square mesh on the unit square with mesh size $h = 1/(n+1)$, then a finite difference approximation leads to the linear system

$$(N + a^2 I)\mathbf{x} = \mathbf{b}, \quad a \in \mathbb{R}, \tag{4}$$

where

$$N = \begin{bmatrix} C+I & 2B & I & & & \\ 2B & C & 2B & I & & \\ I & 2B & C & 2B & I & \\ & \cdots & \cdots & \cdots & \cdots & \cdots \\ & & \cdots & \cdots & \cdots & \cdots \\ & & & I & 2B & C+I \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \cdots \\ \cdots \\ \cdots \\ \mathbf{x}_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \cdots \\ \cdots \\ \cdots \\ \mathbf{b}_n \end{bmatrix},$$

with

$$C = B^2 + (2 + a^2)I + 2EE^{\mathrm{T}}, \quad E^{\mathrm{T}} = \begin{pmatrix} \mathbf{e}_1^{\mathrm{T}} \\ \mathbf{e}_n^{\mathrm{T}} \end{pmatrix}, \quad B = -A \quad \text{and}$$

$$A = \begin{bmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & \cdots & \cdots & \cdots & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{bmatrix} \text{ is an } n \times n \text{ matrix with eigenvalues}$$

$l_j = 4 - 2\cos[j\pi/(n+1)]$, $j = 1, 2, \ldots, n$, and the corresponding eigenvector matrix

$$Q = (q_{ij}) = \left(\frac{2}{n+1}\right)^{1/2} \sin\frac{ij\pi}{n+1}.$$

The system (4) can be expressed as

$$(G + 2FF^{\mathrm{T}})\mathbf{x} = \mathbf{b},$$

where $F = \mathrm{diag}(E, E, \ldots, E)$, $G = M^2 + 2J + a^2 I$ with $J = \mathrm{diag}(I, 0, \ldots, 0, I_n)$ and M is the matrix associated to the five-point finite difference approximation of the Poisson equation with Dirichlet boundary conditions, i.e.

$$M = \begin{bmatrix} A & -I & & & \\ -I & A & -I & & \\ & \cdots & \cdots & \cdots & \\ & & -I & A & -I \\ & & & -I & A \end{bmatrix}.$$

Thus, by the Woodbury formula,

$$\mathbf{x} = \mathbf{G}^{-1}\mathbf{b} - 2G^{-1}F(I + 2F^{\mathrm{T}}G^{-1}F)^{-1}F^{\mathrm{T}}G^{-1}\mathbf{b},$$

and the algorithm proceeds as in the following four stages:

1. Solve the $2n$ linear systems $GY = F$.
2. Solve the linear system $G\mathbf{v} = \mathbf{b}$.
3. Solve the linear system $(I_{2n} + 2F^{\mathrm{T}}Y)\mathbf{z} = \mathbf{F}^{\mathrm{T}}\mathbf{v}$.
4. Compute $\mathbf{x} = \mathbf{v} - 2Y\mathbf{z}$.

It is possible to show that for the solution of the linear systems $G\mathbf{v} = \mathbf{b}$ and $GY = F$, by using direct algorithms similar to that proposed in Ref. [1], $O(\log n)$ steps and a number of processors of order $O(n^2)$ and $O(n^3)$ respectively suffice. On the other hand the use of the PG algorithm allows solving the linear systems $G\mathbf{v} = \mathbf{b}$ and $GY = F$ with $O(n^2)$ processors and a number of steps of order $O(\log n)$ and $O(\log^2 n)$ respectively, under the opportune hypothesis on d^2.

Stage 1

We have to solve the $2n$ systems

$$G\mathbf{y}^k = \mathbf{f}^k, \quad k = 2(i-1)+j, \quad i = 1, 2, \ldots, n, j = 1, 2, \tag{5}$$

where $\mathbf{y}^k$ and $\mathbf{f}^k$ are the kth columns of Y and F respectively, $\mathbf{f}^k = \mathbf{e}_i \otimes \mathbf{e}_1$, if $j = 1$, and $\mathbf{f}^k = \mathbf{e}_i \otimes \mathbf{e}_n$, if $j = 2$, where e_l is the lth column of the $n \times n$ identity matrix.

The problem of solving the linear systems (5) can be reduced [1] to that of solving the $2n^2$ linear systems

$$T_j \hat{\mathbf{y}}_j^k = \hat{\mathbf{f}}_j^k, \quad j = 1, 2, \ldots, n, \quad k = 1, 2, \ldots, 2n, \tag{6}$$

where

$$T_j = \begin{bmatrix} q_j+1 & p_j & 1 & & & \\ p_j & q_j & p_j & 1 & & \\ 1 & p_j & q_j & p_j & 1 & \\ & \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & & p_j & q_j+1 \end{bmatrix}, \quad \text{with } q_j = l_j^2 + 2 + a^2,$$

$$p_j = -2l_j, \quad \hat{\mathbf{f}}_j^k = (\bar{f}_{j1}^k, \bar{f}_{j2}^k, \ldots, \bar{f}_{jn}^k)^T \quad \text{and} \quad \hat{\mathbf{y}}_j^k = (\bar{y}_{j1}^k, \bar{y}_{j2}^k, \ldots, \bar{y}_{jn}^k)^T,$$

in which $\bar{f}_{jl}^k$ and $\bar{y}_{jl}^k$ are the jth components of $\bar{\mathbf{f}}_l^k = Q^T \mathbf{f}_l^k$ and $\bar{\mathbf{y}}_l^k = Q^T \mathbf{y}_l^k$, respectively. Therefore the algorithm consists of the following three stages:

(a) Compute $\bar{\mathbf{f}}_l^k = Q^T \mathbf{f}_l^k$, $l = 1, 2, \ldots, n$, $k = 1, 2, \ldots, 2n$.
(b) Solve the $2n^2$ systems (6).
(c) Compute $\mathbf{y}_l^k = Q\bar{\mathbf{y}}_l^k$, $l = 1, 2, \ldots, n$, $k = 1, 2, \ldots, 2n$.

Stage (a) does not affect the complexity of the algorithm because of the simple structure of F. In Stage (b) it is easy to see that the 5-diagonal matrix T_j cannot be decomposed as the matrix Ω_j in Ref. [1], but can be written as

$$T_j = B_j S_j + 2EE^T,$$

where

$$R_j = \begin{bmatrix} r_j & -1 & & \\ -1 & r_j & -1 & \\ & \cdots & \cdots & \cdots \\ & -1 & r_j & -1 \\ & & -1 & r_j \end{bmatrix}, \quad S_j = \begin{bmatrix} s_j & -1 & & \\ -1 & s_j & -1 & \\ & \cdots & \cdots & \\ & -1 & s_j & -1 \\ & & -1 & s_j \end{bmatrix},$$

are Toeplitz tridiagonal matrices with entries $r_j = l_j + a\mathbf{i}$, $s_j = l_j - a\mathbf{i}$, $\mathbf{i}$ being the imaginary unit, and $|r_j| = |s_j| > 2$.

By the Woodbury formula we have

$$\hat{\mathbf{y}}_j^k = T_j^{-1} \hat{\mathbf{f}}_j^k = S_j^{-1} R_j^{-1} \hat{\mathbf{f}}_j^k - 2S_j^{-1} R_j^{-1} E(I + 2E^T S_j^{-1} R_j^{-1} E)^{-1} E^T S_j^{-1} R_j^{-1} \hat{\mathbf{f}}_j^k,$$

therefore the algorithm proceeds in the following way:

(i) Solve the linear system $R_j S_j \mathbf{z}_j^k = \hat{\mathbf{f}}_j^k$.
(ii) Solve the two linear systems $R_j S_j X_j = E$.
(iii) Solve the linear system $(I_2 + 2E^T X_j)\mathbf{x}_j^k = E^T \mathbf{z}_j^k$.
(iv) Compute $\hat{\mathbf{y}}_j^k = \mathbf{z}_j^k - 2X_j \mathbf{x}_j^k$.

Phase (i) can be performed by successively applying the PG algorithms to the two systems

$$R_j \mathbf{w}_j^k = \hat{\mathbf{f}}_j^k \tag{7a}$$

and

$$S_j \mathbf{z}_j^k = \mathbf{w}_j^k. \tag{7b}$$

It is possible to show that, under the opportune hypothesis on d^2, the bounds $T = O(\log^2 n)$ and $p = O(1)$ can be obtained by successively applying the PG algorithm to the two systems (7a) and (7b), if we want to reduce the initial error by a factor $1/n^4$. In fact, assuming that the errors, for the three iterations which comprise the parallel Gauss algorithm, satisfy relations of the type

$$\|\mathbf{e}^{k+1}\| < s\|\mathbf{e}^k\|,$$

then we estimate the number of iterations to reduce the error norm by $2^{-b} = 1/n^4$ as $K = -4 \log n / \log s$. Therefore a number of iterations *ID*, *IF*, *IX* of order $O(\log n)$ suffice, provided that $s = s(n)$ is upper bounded by a constant < 1. We will comment on this point later.

The system (7a) is of the type $R\mathbf{w} = \mathbf{g}$ where $\mathbf{g} = \mathbf{e}_i b$, $\mathbf{e}_i$ being the ith column of the $n \times n$ identity matrix, and R is an $n \times n$ Toeplitz matrix of the type (2). As previously noted, at the kth step of the first iterative process of the PG algorithm only the $(k+1)$th entry has to be computed. Therefore $O(\log n)$ steps and $O(1)$ processors suffice to perform the first phase of the PG algorithm.

Let $R = (I + L)D(I + L^T)$ be the factorization of R, then the solution of $(I + L)\mathbf{p} = \mathbf{g}$ can be obtained in $T = O(\log n)$ steps with $O(1)$ processors because of the special structure of vector $\mathbf{g}$ and of matrix $(I + L)$. In fact by applying the iterative method $\mathbf{p}^0 = \mathbf{g}$. $\mathbf{p}^k = \mathbf{g} - L\mathbf{p}^{k-1}$, where $\mathbf{p}^0 = \mathbf{e}_i b$ and $L = \text{subdiag}(l_2, l_3, \ldots, l_n)$, at the IFth step we obtain

$$\mathbf{p}^{IF} = \sum_{j=i}^{i+IF} (-1)^{j-i} \mathbf{e}_j q_j, \quad \text{with} \quad q_i = b, \quad q_j = l_j q_{j-1}, \quad j = i+1, \ldots, i+IF.$$

Therefore we have to perform one multiplicative step at each iteration and after $\text{IF} = O(\log n)$ iterations we obtain a vector $\mathbf{p}^{\text{IF}}$ having only $O(\log n)$ nonzero entries.

Analogously, the special structure of vector $\mathbf{p}^{\text{IF}}$ and the special structure of matrix $(I + L^T)$, allow solving the system $R\mathbf{w} = \mathbf{g}$ in no more than $O(\log^2 n)$ steps with $O(1)$ processors. Moreover the resulting vector has only $O(\log n)$ nonzero entries. From these considerations it follows that phase (i) can be carried out in $T = O(\log^2 n)$ steps with $p = O(1)$ processors.

Phase (ii) can be carried out in $T = O(\log^2 n)$ steps and $p = O(1)$ processors.

Phase (iii) can be performed in $O(1)$, steps with $O(1)$ processors.

Phase (iv) can be performed in $O(\log n)$ steps with $O(1)$ processors because of the special structure of matrix X_j and vector $\mathbf{z}_j^k$. Therefore stage (b) requires $T = O(\log^2 n)$ steps by using no more than $p = O(n^2)$ processors.

Stage (c), because of the special structure of $\mathbf{f}^k$ and the finiteness of the PG algorithm, we have that $\bar{\mathbf{y}}^k$ contains only $O(\log n)$ blocks $\bar{\mathbf{y}}_i^k$ not equal to the null vector. Therefore stage (c) requires only $O(n \log n)$ FFT which can be performed in $T = O(\log^2 n)$ steps by using no more than $p = O(n^2)$ processors. From the previous analysis it follows that Stage 1 can be carried out in $T = O(\log^2 n)$ steps using $p = O(n^2)$ processors.

Note that at phases (i) and (ii) of the previous algorithms we can apply a direct method to solve both systems (7a) and (7b). This method can be derived from a decomposition of the Toeplitz tridiagonal matrices (in the complex field) similar to that shown in Ref. [1] (see Lemma 4). It is easy to see that this direct algorithm to solve the linear systems (7a) and (7b) can be carried out in $O(\log n)$ steps with $O(n)$ processors. Then, in this case, stage (b) can be performed in $T = O(\log n)$ steps with $p = O(n^2)$ processors. Moreover stage (c) consists of $O(n^2)$ FFT which can be performed in $T = O(\log n)$ steps by using $p = O(n^3)$ processors. The resulting bounds of this direct algorithm are $T = O(\log n)$, $p = O(n^3)$ or, as pointed out in Ref. [1], if the requirement

of $O(n^3)$ processors is impractical, we may restrict it to $O(n^2)$ at the cost of increasing time $O(n \log n)$.

Stage 2

Let us consider now the problem of solving the linear system $G\mathbf{v} = \mathbf{b}$. Note that in this case we cannot make use of the structure of vector $\mathbf{b}$, then the trivial application of the previous algorithm produces the bounds $T = O(\log n)$, $p = O(n^2)$ either by using a direct method similar to that shown in Ref. [3] or by using the PG iterative method.

Stages 3 and 4

Can be performed in $T = O(n)$ steps with $p = O(n^2)$ processors [1].

Therefore $T = O(n)$ steps and $p = O(n^2)$ processors suffice to solve equation (4). The resulting speed up and efficiency of this algorithm are respectively $S = O(n^2)$ and $E = O(1)$ and they follow from the fact that the fastest available serial algorithm for the finite difference approximation of the perturbed biharmonic equation requires $O(n^3)$ steps [3].

4. CONCLUSION

We have shown that the use of the PG algorithm in the solution of the finite approximation (4) of the perturbed biharmonic equation on the unit square produces the bounds $p = O(n^2)$, $T = O(n)$, with a resulting speed up $S = O(n^2)$ and efficiency $E = O(1)$, which are better than the corresponding ones obtained by direct methods.

We have shown also that the above statement is valid when $s = 1 - e$, with $e > e' > 0$ for each n. This means that the constant d^2 in equation (3) must be numerically of the same order of n^4.

The required approximation error implies a precise choice of n. Once n is fixed the proposed method is convenient when the constant d^2 is sufficiently large to make e' such that $\log n / \log s \simeq \log n$.

On the other hand, it is possible to see that the use of the PG algorithm in the solution of the finite approximation of the perturbed Poisson equation with Dirichlet boundary conditions on a square domain produces the same bounds obtained by direct methods, that is $p = O(n^2)$, $T = O(\log n)$ [1, 4].

Acknowledgements—It is a pleasure to thank M. Capovani and D. Bini for helpful discussions and suggestions. The author wants to thank P. Flajolet for his kind hospitality at the INRIA (France) where most of the work was carried out.

REFERENCES

1. A. H. Sameh, S. C. Chen and D. J. Kuck, Parallel Poisson and biharmonic solvers. *Computing* **17**, 219–230 (1976).
2. D. E. Heller, D. K. Stevenson and J. F. Traub, Accelerated iterative methods for the solution of tridiagonal systems on parallel computers. *J. Ass. comput. Mach.* **23**, 636–654 (1976).
3. B. L. Buzbee and F. W. Dorr, The direct solution of the biharmonic equation on rectangular regions and the Poisson equation on irregular regions. *SIAM Jl numer. Analysis* **11**, 753–762 (1974).
4. B. L. Buzbee, G. H. Golub and C. W. Nielson, On direct methods for solving Poisson's equations. *SIAM Jl numer. Analysis* **7**, 627–656 (1970).