Journal of Scientific Computing, Vol. 12, No. 4, 1997

A Parallel Solver for Circulant Toeplitz Tridiagonal Systems on Hypercubes

Jung-Gen Wu,[1] Wen-Ming Yan,[2] and Kuo-Liang Chung[3]

Received December 7, 1995

Solving circulant Toeplitz tridiagonal systems arises in many engineering applications. This paper presents a fast parallel algorithm for solving this type of systems. The number of floating-point operations required in our algorithm is less than the previous parallel algorithm [cf. Kim and Lee (1990)] for solving the similar system. Specifically, an overlapping technique is proposed to reduce the communication steps required. In addition, an error analysis is given. The implementation of our algorithm on the nCUBE2/E with 16 processors has been carried out. The experimental results show that the speedup is almost linearly proportional to the number of processors.

KEY WORDS: Diagonally dominant matrices; error analysis; parallel matrix computations; Toeplitz tridiagonal matrices.

1. INTRODUCTION

Throughout this paper, matrices are represented by uppercase letters, vectors by bold lowercase letters, and scalars by lowercase letters. The superscript T corresponds to the transpose operation. Consider to solve an $n \times n$ circulant near-Toeplitz system

$$A\mathbf{x} = \mathbf{b}, \tag{1.1}$$

[1] Department of Information and Computer Education, National Taiwan Normal University, Taipei, Taiwan 10610, R. O. C. E-mail: jgwu@ice.ntnu.edu.tw. This research was supported in part by the National Science Council of R. O. C. under contract NSC85-2213-E003-001.

[2] Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan 10764, R. O. C. E-mail: ganboon@csie.ntu.edu.tw.

[3] To whom correspondence should be addressed at Department of Information Management, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei, Taiwan 10672, R. O. C. E-mail: klchung@cs.ntust.edu.tw. This research was supported in part by the National Science Council of R. O. C. under contracts NSC85-2121-M011-002 and NSC85-2213-E011-009.

0885-7474/97/1200-0409$12.50/0

where

$$A=\begin{pmatrix} \alpha_1 & \gamma & & & & \beta_1 \\ \beta & \alpha & \gamma & & & \\ & \cdot & \cdot & \cdot & & \\ & & & \beta & \alpha & \gamma \\ \gamma_2 & & & & \beta & \alpha_2 \end{pmatrix}$$

and $|\alpha| > |\beta + \gamma|$.

Solving (1.1) arises in many applications [cf. Hockney (1965); Widlund (1972); Fisher *et al.* (1974); Smith (1985); Chung and Yan (1994); Hirsh (1975)]. Previously, [cf. Kim and Lee (1990)] presented an efficient parallel algorithm for solving (1.1) with $\alpha_1 = \alpha_2 = \alpha$, $\beta = \gamma$, and $\beta_1 = \gamma_2 = 0$; their algorithm needs $14n/p$ floating-point (FP) operations and $O(p)$ communication steps, where p is the number of processors.

In this paper, we present a fast parallel algorithm to solve (1.1). The number of FP operations required in our algorithm is about $9n/p$; the communication steps required is $O(\log p)$. This result is superior to the parallel result [cf. Kim and Lee (1990)]. Further, we present a truncated version of our parallel algorithm, and the number of FP operations is ranged from $5n/p$ to $9n/p$. Specifically, an overlapping technique is proposed to reduce the communication steps required. An error analysis is also given. Our parallel algorithm is carried out on the nCUBE 2/E multicomputer with 16 processors. The experimental results show that the speedup is almost linearly proportional to the number of processors.

The remainder of this paper is organized as follows. Section 2 presents our parallel algorithm for solving (1.1). Section 3 presents the truncated version of our parallel algorithm and the related error analysis. Section 4 gives experimental results of executing our algorithm on the nCUBE 2/E multicomputer.

2. THE PARALLEL ALGORITHM

We first consider how to solve $A'\mathbf{z} = \mathbf{b}$, where

$$A'=\begin{pmatrix} a & \gamma & & & & \\ \beta & \alpha & \gamma & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \beta & \alpha & \gamma \\ & & & & \beta & \alpha \end{pmatrix} = LU, \qquad L=\begin{pmatrix} 1 & & & & & \\ -r & 1 & & & & \\ & \cdot & \cdot & & & \\ & & \cdot & \cdot & & \\ & & & -r & 1 & \\ & & & & -r & 1 \end{pmatrix},$$

and

$$U=\begin{pmatrix} a & \gamma & & & & \\ & a & \gamma & & & \\ & & \cdot & \cdot & & \\ & & & \cdot & \cdot & \\ & & & & a & \gamma \\ & & & & & a \end{pmatrix}$$

It follows that $-r\gamma+a=\alpha$ and $r=-\beta/a$. Let $s=-\gamma/a$, we have $a^2-\alpha a+\beta\gamma=0$, $\beta+\alpha r+\gamma r^2=0$, $\beta s^2+\alpha s+\gamma=0$, $\beta s+\alpha=a$, $\alpha+\gamma r=a$, and $\alpha-a=ars$, which will be used later. These six equalities are verified in Appendix A. Solving a from $a^2-\alpha a+\beta\gamma=0$. It give $a=(\alpha\pm\sqrt{\alpha^2-2\beta\gamma})/2$. When $\alpha>|\beta+\gamma|$, we select $a=(\alpha+\sqrt{\alpha^2-4\beta\gamma})/2$; we select $a=(\alpha-\sqrt{\alpha^2-4\beta\gamma})/2$ when $\alpha<-|\beta+\gamma|$. It is clear that our selection always make the matrix L and U to be diagonally dominant.

We solve $A'\mathbf{z}=\mathbf{b}$ by solving $L\mathbf{y}=\mathbf{b}$ first using a forward substitution procedure and then solving $U\mathbf{z}=\mathbf{y}$ using a backward substitution procedure. Suppose we have p processors. The $\mathbf{b}$ is partitioned into p parts and

$$\mathbf{b}=\begin{pmatrix} \mathbf{b}^{(0)} \\ \mathbf{b}^{(1)} \\ \vdots \\ \mathbf{b}^{(p-1)} \end{pmatrix}$$

where the length of vector $\mathbf{b}^{(i)}$ is n_i for $0\leqslant i\leqslant p-1$. Then processor i solves $A'_{n_i}\mathbf{z}^{(i)}=\mathbf{b}^{(i)}$ sequentially using Gaussian elimination method and it takes about $5n_i$ FP operations. Naturally, we partition the system $A'\mathbf{z}=\mathbf{b}$ into parts evenly, then each processor takes about $5n/p$ FP operations for solving $A'_{n_i}\mathbf{z}^{(i)}=\mathbf{b}^{(i)}$. Let

$$\hat{A}_n=\begin{pmatrix} a & \gamma & & & & \\ \beta & \alpha & \gamma & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \beta & \alpha & \gamma \\ & & & & \beta & \alpha \end{pmatrix}_{n\times n}$$

be a perturbed matrix of A, which will be used later too. For convenience, for any vector $\mathbf{x}$, we denote $\bar{\mathbf{x}}$ to be the first entry of $\mathbf{x}$ and $\underline{\mathbf{x}}$ to be the last entry of $\mathbf{x}$. We then have

$$\hat{A}_n \begin{pmatrix} \mathbf{z}^{(0)} \\ \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}^{(p-1)} \end{pmatrix} = \begin{pmatrix} \mathbf{b}^{(0)} \\ \mathbf{b}^{(1)} \\ \vdots \\ \mathbf{b}^{(p-1)} \end{pmatrix} - ars\underline{\mathbf{z}}^{(p-1)}\mathbf{e}_n - \sum_{i=1}^{p-1} [as\bar{\mathbf{z}}^{(i)}\mathbf{e}_{m_i} + (ar\underline{\mathbf{z}}^{(i-1)} - ars\bar{\mathbf{z}}^{(i)})\,\mathbf{e}_{m_i+1}] \tag{2.2}$$

where $m_i = n_0 + n_1 + \cdots + n_{i-1}$, $m_0 = 0$, and $\mathbf{e}_i = (\underbrace{0,\ldots,0}_{i}, 1, \underbrace{0,\ldots,0}_{n-i})$; (2.2) is verified in Appendix B.

Let $\mathbf{p}_k = (\underbrace{0,\ldots,0}_{k}, \underbrace{1, r,\ldots, r^{n-k-2}, r^{n-k-1}}_{n-k})^T$ and $\mathbf{q}_k = (\underbrace{s^{k-1}, s^{k-2},\ldots, s, 1}_{k}, \underbrace{0,\ldots,0}_{n-k})^T$, it yields to

$$\begin{aligned} \hat{A}_n\mathbf{q}_n &= (\beta s + a)\,\mathbf{e}_n = a(1 - rs)\,\mathbf{e}_{m_p} \\ \hat{A}_n\mathbf{p}_k &= \gamma\mathbf{e}_k + (\alpha + \gamma r)\,\mathbf{e}_{k+1} = -as\mathbf{e}_k + a\mathbf{e}_{k+1} \end{aligned} \tag{2.3}$$

and

$$\hat{A}_n\mathbf{q}_k = (\beta s + \alpha)\,\mathbf{e}_k + \beta\mathbf{e}_{k+1} = a\mathbf{e}_k - ar\mathbf{e}_{k+1} \qquad \text{for} \quad 1 < k < n \tag{2.4}$$

Since $\mathbf{e}_1$ and $\mathbf{e}_n$ are absent in these two equations, $\hat{A}^{-1}\mathbf{e}_k$ and $\hat{A}^{-1}\mathbf{e}_{k+1}$ can be represented in terms of $\mathbf{p}_k$ and $\mathbf{q}_k$. As a result, we can recover the solution of $\hat{A}\mathbf{x}' = \mathbf{b}$ from (2.2), then recover the solution of $A\mathbf{x} = \mathbf{b}$ from $\hat{A}\mathbf{x}' = \mathbf{b}$. In what follows, we present a parallel algorithm to realize this approach.

To recover the solution of $\hat{A}\mathbf{x}' = \mathbf{b}$ from (2.2), we first solve

$$\begin{aligned} &\hat{A}_n\mathbf{w}_i = -as\bar{\mathbf{z}}^{(i)}\mathbf{e}_{m_i} - (ar\underline{\mathbf{z}}^{(i-1)} - ars\bar{\mathbf{z}}^{(i)})\,\mathbf{e}_{m_i+1}, \qquad i = 1, 2,\ldots, p-1 \\ &\hat{A}_n\mathbf{w}_p = -ars\underline{\mathbf{z}}^{(p-1)}\mathbf{e}_n. \end{aligned} \tag{2.5}$$

By (2.3) and (2.4), we let

$$\mathbf{w}_i = c_1(i)\,\mathbf{p}_{m_i} + c_2(i)\,\mathbf{q}_{m_i} \tag{2.6}$$

for $i = 1, 2,\ldots, p$, where $c_1(i)$ and $c_2(i)$ are to be determined. Since

$$\begin{aligned} \hat{A}_n\mathbf{w}_i &= \hat{A}_n(c_1(i)\,\mathbf{p}_{m_i} + c_2(i)\,\mathbf{q}_{m_i}) \\ &= -as\bar{\mathbf{z}}^{(i)}\mathbf{e}_{m_i} - (ar\underline{\mathbf{z}}^{(i-1)} - ars\bar{\mathbf{z}}^{(i)})\,\mathbf{e}_{m_i+1} \end{aligned} \tag{2.7}$$

for $i = 1, 2, \ldots, p-1$, by (2.4), we have

$$-sc_1(i) + c_2(i) = -s\bar{\mathbf{z}}^{(i)} \qquad \text{and} \qquad c_1(i) - rc_2(i) = -r\underline{\mathbf{z}}^{(i-1)} + rs\bar{\mathbf{z}}^{(i)}$$

Solving these two equations, it yields to

$$c_1(i) = \frac{r}{rs-1}\underline{\mathbf{z}}^{(i-1)} \qquad \text{and} \qquad c_2(i) = \frac{rs}{rs-1}\underline{\mathbf{z}}^{(i-1)} - s\bar{\mathbf{z}}^{(i)} \tag{2.8}$$

for $i = 1, 2, \ldots, p-1$. By (2.3), we have

$$\hat{A}_n\left(\frac{rs}{rs-1}\underline{\mathbf{z}}^{(p-1)}\mathbf{q}_{m_p}\right) = -ars\underline{\mathbf{z}}^{(p-1)}\mathbf{e}_{m_p} \tag{2.9}$$

Therefore, it is very natural to let

$$c_1(p) = 0 \qquad \text{and} \qquad c_2(p) = \frac{rs}{rs-1}\underline{\mathbf{z}}^{(p-1)} \tag{2.10}$$

After solving (2.5), by (2.2), the solution of $\hat{A}\mathbf{x}' = \mathbf{b}$ is recovered from

$$\mathbf{x}' = \begin{pmatrix} \mathbf{z}^{(0)} \\ \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}(p-1) \end{pmatrix} - \sum_{i=1}^{p} \mathbf{w}_i \tag{2.11}$$

where $\sum_{i=1}^{p} \mathbf{w}_i$ is the updated term.

For parallelizing the computation of $\sum_{i=1}^{p} \mathbf{w}_i$, let

$$\hat{\mathbf{p}}_i = (\underbrace{0, \ldots, 0}_{m_i}, 1, r, \ldots, r^{n_i-1}, 0, \ldots, 0)^T$$

and

$$\hat{\mathbf{q}}_i = (\underbrace{0, \ldots, 0}_{m_i}, s^{n_i-1}, \ldots, s, 1, 0, \ldots, 0)^T$$

where the nonzero terms in $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{q}}_i$ will be hold by processor i. This brings out the parallel computation of $\sum_{i=1}^{p} \mathbf{w}_i$. Now we want to rewrite the updated term, $\sum_{i=1}^{p} \mathbf{w}_i$, as a linear combination of $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{q}}_i$. By the definition of $\mathbf{p}_k$, $\mathbf{q}_k$, and m_i, we have

$$\mathbf{p}_{m_i} = \sum_{j=i}^{p-1} r^{m_j - m_i}\hat{\mathbf{p}}_j \qquad \text{and} \qquad \mathbf{q}_{m_i} = \sum_{j=0}^{i-1} s^{m_i - m_{j+1}}\hat{\mathbf{q}}_j.$$

Therefore, it is given by

$$\begin{aligned}\sum_{i=1}^{p} \mathbf{w}_i &= \sum_{i=1}^{p-1} c_1(i)\, \mathbf{p}_{m_i} + \sum_{i=1}^{p} c_2(i)\, \mathbf{q}_{m_i} \\ &= \sum_{i=1}^{p-1} c_1(i) \sum_{j=i}^{p-1} r^{m_j - m_i} \hat{\mathbf{p}}_j + \sum_{i=1}^{p} c_2(i) \sum_{j=0}^{i-1} s^{m_i - m_{j+1}} \hat{\mathbf{q}}_j \\ &= \sum_{j=1}^{p-1} c_3(j)\, \hat{\mathbf{p}}_j + \sum_{j=0}^{p-1} c_4(j)\, \hat{\mathbf{q}}_j, \end{aligned} \tag{2.12}$$

where

$$\begin{aligned} c_3(j) &= \sum_{i=1}^{j} r^{m_j - m_i} c_1(i) = \sum_{i=1}^{i} r^{m_j - m_i} \frac{r}{rs-1} \underline{\mathbf{z}}^{(i-1)} \\ &= \sum_{i=0}^{j-1} r^{m_j - m_{i+1}} \frac{r}{rs-1} \underline{\mathbf{z}}^{(i)} = r \sum_{i=0}^{j-1} r^{m_j - m_{i+1}} g_i = r c_5(j-1) \\ c_4(j) &= \sum_{i=j+1}^{p} s^{m_i - m_{j+1}} c_2(i) \\ &= \sum_{i=j+1}^{p} s^{m_i - m_{j+1}} \frac{rs}{rs-1} \underline{\mathbf{z}}^{(i-1)} - \sum_{i=j+1}^{p-1} s^{m_i - m_{j+1}} s \bar{\mathbf{z}}^{(i)} \\ &= \sum_{i=j}^{p-1} s^{m_{i+1} - m_{j+1}} \frac{rs}{rs-1} \underline{\mathbf{z}}^{(i)} - \sum_{i=j+1}^{p-1} s^{m_i - m_{j+1}} s \bar{\mathbf{z}}^{(i)} \\ &= \frac{rs}{rs-1} \underline{\mathbf{z}}^{(j)} + \sum_{i=j+1}^{p-1} s^{m_i - m_{j+1}} \left[s^{n_i} \frac{rs}{rs-1} \underline{\mathbf{z}}^{(i)} - s \bar{\mathbf{z}}^{(i)} \right] \\ &= \frac{rs}{rs-1} \underline{\mathbf{z}}^{(j)} + s \sum_{i=j+1}^{p-1} s^{m_i - m_{j+1}} h_i = \frac{rs}{rs-1} \underline{\mathbf{z}}^{(j)} + s c_6(j+1) \end{aligned} \tag{2.13}$$

with

$$g_i = \frac{1}{rs-1} \underline{\mathbf{z}}^{(i)}, \qquad h_i = s^{n_i} \frac{r}{rs-1} \underline{\mathbf{z}}^{(i)} - \bar{\mathbf{z}}^{(i)} \tag{2.14}$$

$$c_5(j) = \sum_{i=0}^{j} r^{m_{j+1} - m_{i+1}} g_i, \qquad \text{and} \qquad c_6(j) = \sum_{i=j}^{p-1} s^{m_i - m_j} h_i \tag{2.15}$$

Processor i is responsible for computing g_i and h_i locally and it takes $O(1)$ time; computing $c_5(i-1)$ and $c_6(i+1)$ needs global communications and it takes $O(\log p)$ communication steps on the hypercube network [cf. Ranka and Sahni (1990)].

Now we want to recover the solution of $A_n\mathbf{x}=\mathbf{b}$ from $\hat{A}_n\mathbf{x}'=\mathbf{b}$. By $\hat{A}_n\mathbf{x}'=\mathbf{b}$, we have

$$A_n\mathbf{x}'=\hat{A}_n\mathbf{x}'+(A_n-\hat{A}_n)\,\mathbf{x}'=\mathbf{b}+f_1\mathbf{e}_1+f_2\mathbf{e}_n$$

where

$$f_1=(\alpha_1-a)\,x_1'+\beta_1 x_n' \qquad \text{and} \qquad f_2=\gamma_2 x_1'+(\alpha_2-a)\,x_n' \tag{2.16}$$

We wish to represent x_1' and x_n' in term of $c_6(0)$ and $c_5(p-1)$, respectively. By (2.11)–(2.14), we have

$$\begin{aligned} x_1' &= \bar{\mathbf{z}}^{(0)}-c_4(0)\,s^{n_0-1}=\bar{\mathbf{z}}^{(0)}-s^{n_0-1}\left[\frac{rs}{rs-1}\,\underline{\mathbf{z}}^{(0)}+s\sum_{i=1}^{p-1}s^{m_i-m_1}h_i\right] \\ &= -\left[h_0+s^{n_0}\sum_{i=1}^{p-1}s^{m_i-m_1}h_i\right] \end{aligned}$$

and

$$\begin{aligned} x_n' &= \underline{\mathbf{z}}^{(p-1)}-c_3(p-1)\,r^{n_{p-1}-1}-c_4(p-1) \\ &= \underline{\mathbf{z}}^{(p-1)}-r^{n_{p-1}}\sum_{i=0}^{p-2}r^{m_{p-1}-m_{i+1}}g_i-\frac{rs}{rs-1}\,\underline{\mathbf{z}}^{(p-1)} \\ &= -\left[g_{p-1}+r^{n_{p-1}}\sum_{i=0}^{p-2}r^{m_{p-1}-m_{i+1}}g_i\right] \end{aligned}$$

then by (2.15), it yields to

$$x_1'=-c_6(0) \qquad \text{and} \qquad x_n'=-c_5(p-1) \tag{2.17}$$

From the definitions of A_n, $\mathbf{p}_0$, and $\mathbf{q}_n$, it follows that

$$A_n\mathbf{p}_0=g_1\mathbf{e}_1+g_2\mathbf{e}_n \qquad \text{and} \qquad A_n\mathbf{q}_n=h_1\mathbf{e}_1+h_2\mathbf{e}_n$$

where

$$\begin{aligned} &g_1=\alpha_1+\gamma r+\beta_1 r^{n-1}, \qquad g_2=\gamma_2+\beta r^{n-2}+\alpha_2 r^{n-1}, \\ &h_1=\alpha_1 s^{n-1}+\gamma s^{n-2}+\beta_1, \qquad \text{and} \qquad h_2=\gamma_2 s^{n-1}+\beta s+\alpha_2 \end{aligned} \tag{2.18}$$

By (2.16), we have

$$\mathbf{x}=\mathbf{x}'-c_1\mathbf{p}_0-c_2\mathbf{q}_n \tag{2.19}$$

where c_1 and c_2 satisfy

$$c_1 g_1 + c_2 h_1 = f_1 \qquad \text{and} \qquad c_1 g_2 + c_2 h_2 = f_2 \tag{2.20}$$

By (2.19), (2.11), and (2.12), we have

$$\begin{aligned} \mathbf{x} &= \mathbf{x}' - c_1 \mathbf{p}_0 - c_2 \mathbf{q}_n = \mathbf{x}' - c_1 \sum_{j=0}^{p-1} r^{m_j} \hat{\mathbf{p}}_j - c_2 \sum_{j=0}^{p-1} s^{n-m_{j+1}} \hat{\mathbf{q}}_j \\ &= \mathbf{z} - \sum_{j=0}^{p-1} c_7(j)\, \hat{p}_j - \sum_{j=0}^{p-1} c_8(j)\, \hat{\mathbf{q}}_j \end{aligned} \tag{2.21}$$

where

$$c_7(j) = c_3(j) + c_1 r^{m_j} = r c_5(j-1) + c_1 r^{m_j}$$

and

$$c_8(j) = c_4(j) + c_2 s^{n-m_{j+1}} = \frac{rs}{rs-1}\, \underline{\mathbf{z}}^{(j)} + s c_6(j+1) + c_2 s^{n-m_{j+1}} \tag{2.22}$$

Each processor takes $O(1)$ time for computing (2.22) and about $4n/p$ FP operations for computing (2.21). Our parallel algorithm totally takes about $9n/p$ FP operations and $O(\log p)$ communication steps.

Our detailed parallel algorithm for solving (1.1) is shown later. In the following parallel algorithm, i = node-id (the address of the processor), $n_0 = n_1 = \cdots = n_{k-1} = \lceil n/p \rceil$, $n_k = n_{k+1} = \cdots = n_{p-1} = \lfloor n/p \rfloor$, where $k = n \bmod p$.

1. Compute a, r, s, n_i, m_j, and m_{j+1}

$$\text{If } \alpha > |\beta + \gamma| \quad \text{then} \quad a \leftarrow \frac{\alpha + \sqrt{\alpha^2 - 4\beta\gamma}}{2} \quad \text{else} \quad a \leftarrow \frac{-\sqrt{\alpha^2 - 4\beta\gamma}}{2},$$

$$r \leftarrow \frac{\beta}{a}, \qquad s \leftarrow \frac{\gamma}{a}$$

2. Solve $A'_{n_i} \mathbf{z}^{(i)} = \mathbf{b}^{(i)}$ /*solve $L_{n_i} \mathbf{y}^{(i)} = \mathbf{b}^{(i)}$ first and then solve $U_{n_i} \mathbf{z}^{(i)} = \mathbf{y}^{(i)}$*/

3. Compute $g_i \leftarrow (1/rs - 1)\, \underline{\mathbf{z}}^{(i)}$, $h_i \leftarrow s^{n_i} (r/rs - 1)\, \underline{\mathbf{z}}^{(i)} - \bar{\mathbf{z}}^{(i)}$ /*see (2.14)*/

4. Compute $c_5(i-1)$, $c_5(p-1)$, $c_6(i+1)$, and $c_6(0)$ by using prefix sum and postfix sum algorithms (for the implementation on hypercube, we recall the function *comm* which is shown and simulated in Appendix C /*see (2.15)*/

5. Calculate $x'_1 \leftarrow -c_6(0)$, $x'_n \leftarrow -c_5(p-1)$ /*see (2.17)*/

6. Calculate $g_1 \leftarrow \alpha_1 + \gamma r + \beta_1 r^{n-1}$, $g_2 \leftarrow \gamma_2 + \beta r^{n-2} + \alpha_2 r^{n-1}$, $h_1 \leftarrow \alpha_1 s^{n-1} + \gamma s^{n-2} + \beta_1$, $h_2 \leftarrow \gamma_2 s^{n-1} + \beta s + \alpha_2$, $c_1 \leftarrow (f_1 h_2 - f_2 h_1)/(g_1 h_2 - g_2 h_1)$, $c_2 \leftarrow (g_1 f_2 - g_2 f_1)/(g_1 h_2 - g_2 h_1)$ /*see (2.18) and (2.20)*/

7. Calculate $c_7(i) \leftarrow r c_5(i-1) + c_1 r^{m_i}$, $c_8(i) \leftarrow (rs/rs - 1)\,\underline{\mathbf{z}}^{(i)} + s c_6(i+1) + c_2 s^{n-m_{i+1}}$ /*see (2.22)*/

8. $\mathbf{x} \leftarrow \mathbf{z} - c_7(i)(1, r, r^2, \ldots, r^{n_i-1})^T - c_8(i)(s^{n_i-1}, \ldots, s^2, s, 1)^T$ /*see (2.21)*/

3. THE TRUNCATED VERSION OF OUR ALGORITHM

By observing the entries of $\hat{\mathbf{p}}_j$ and $\hat{\mathbf{q}}_j$, and recalling that $|r| < 1$ and $|s| < 1$, due to the fact that $|r|^k$ and $|s|^k$ are infinitesimal when k is somewhat large, we try to truncate some small enough entries in $\hat{\mathbf{p}}_j$ and $\hat{\mathbf{q}}_j$. Let $\hat{\mathbf{p}}_j(k) = (\underbrace{0, \ldots, 0}_{m_j}, \underbrace{1, r, r^2, \ldots, r^{k-1}}_{n_j}, 0, \ldots, 0, 0, \ldots, 0)^T$ and $\mathbf{q}_j(k) = (\underbrace{0, \ldots, 0}_{m_j}, \underbrace{0, \ldots, 0, s^{k-1}, \ldots, s^2, s, 1}_{n_j}, 0, \ldots, 0)^T$, we have

$$\|\hat{\mathbf{p}}_j - \hat{\mathbf{p}}_j(k)\| \leqslant |r|^k \qquad \text{and} \qquad \|\hat{\mathbf{q}}_j - \mathbf{q}_j(k)\| \leqslant |s|^k, \tag{3.1}$$

where $\|\bullet\|$ denotes the infinite sup-norm.

Therefore, (2.19) is rewritten by

$$\mathbf{x}(k, l) = \mathbf{z} - \sum_{j=0}^{p-1} c_7(j)\,\hat{\mathbf{p}}_j(k) - \sum_{j=0}^{p-1} c_8(j)\,\hat{\mathbf{q}}_j(l) \tag{3.2}$$

where $k, l \leqslant \min(n_0, n_0, \ldots, n_{p-1})$, then it yields to

$$\frac{\|\mathbf{x} - \mathbf{x}(k, l)\|}{\|\mathbf{x}\|} \leqslant \left(\max_{0 \leqslant j \leqslant p-1} \frac{|c_7(j)|}{\|\mathbf{x}\|}\right) |r^k| + \left(\max_{0 \leqslant j \leqslant p-1} \frac{|c_8(j)|}{\|\mathbf{x}\|}\right) |s^l| \tag{3.3}$$

To find an upperbound of $\|\mathbf{x} - \mathbf{x}(k, l)\|/\|\mathbf{x}\|$ in term of α, β, γ, α_1, β_1, γ_2, α_2, n and p, we need the following lemma:

Lemma 1. Let $\mathbf{b}^{\langle 2\rangle}$ be a $q \times 1$ vector, $\pi_{ij} = \mathbf{e}_i^T A_q'^{-1} \mathbf{e}_j$, $i = 1, q$ and $j = 1, q$, and

$$A_n \mathbf{x} = \mathbf{b} = \begin{pmatrix} \mathbf{b}^{\langle 1\rangle} \\ \mathbf{b}^{\langle 2\rangle} \\ \mathbf{b}^{\langle 3\rangle} \end{pmatrix} \qquad \text{and} \qquad A'_q \mathbf{y} = \mathbf{b}^{\langle 2\rangle},$$

then

$$\frac{|\bar{\mathbf{y}}|}{\|\mathbf{x}\|} \leqslant u_q \qquad \text{and} \qquad \frac{|\mathbf{y}|}{\|\mathbf{x}\|} \leqslant v_q,$$

where

$$\begin{aligned}
u_q &= \max(u_q^{(1)}, u_q^{(2)}, u_q^{(3)}) \\
v_q &= \max(v_q^{(1)}, v_q^{(2)}, v_q^{(3)}) \\
u_q^{(1)} &= |\beta\pi_{11}| + |1 + (\alpha - a)\,\pi_{11}| + |\gamma\pi_{1q}| \\
u_q^{(2)} &= |1 + (\alpha_1 - a)\,\pi_{11}| + |\gamma\pi_{1q}| + |\beta_1\pi_{11}| \\
u_q^{(3)} &= |\gamma_2\pi_{1q}| + |\beta\pi_{11}| + |1 + (\alpha - a)\,\pi_{11}| + |(\alpha_2 - \alpha)\,\pi_{1q}| \\
v_q^{(1)} &= |\beta\pi_{q1}| + 1 + |(\alpha - a)\,\pi_{q1}| + |\gamma\pi_{qq}| \\
v_q^{(2)} &= 1 + |(\alpha_1 - a)\,\pi_{q1}| + |\gamma\pi_{qq}| + |\beta_1\pi_{q1}| \\
v_q^{(3)} &= |\gamma_2\pi_{qq}| + |\beta\pi_{q1}| + |1 + (\alpha_2 - \alpha)\,\pi_{qq}| + |(\alpha - a)\,\pi_{q1}|
\end{aligned}$$

Proof. See Appendices D and E.

By Lemma 1 and $A'_{n_i}\mathbf{z}^{(i)} = \mathbf{b}^{(i)}$, we have

$$\frac{|\bar{\mathbf{z}}^{(i)}|}{\|\mathbf{x}\|} \leqslant u_{n_i} \leqslant u \qquad \text{and} \qquad \frac{|\underline{\mathbf{z}}^{(i)}|}{\|\mathbf{x}\|} \leqslant v_{n_i} \leqslant v \tag{3.4}$$

where

$$\begin{aligned}
u &= \max(u_{n_0}, u_{n_1}, \ldots, u_{n_{p-1}}) = \max(u_{n_0}, u_{n_{p-1}}) \\
v &= \max(v_{n_0}, v_{n_1}, \ldots, v_{n_{p-1}}) = \max(v_{n_0}, v_{n_{p-1}})
\end{aligned} \tag{3.5}$$

By (2.13)–(2.17), (2.20), and (2.22), we have

$$\frac{|c_7(j)|}{\|\mathbf{x}\|} \leqslant t_7 \qquad \text{and} \qquad \frac{|c_8(j)|}{\|\mathbf{x}\|} \leqslant t_8$$

where t_7 and t_8 are referred to Appendix F. By (3.3), the relative error is given by

$$\begin{aligned}
\frac{\|\mathbf{x} - \mathbf{x}(k, l)\|}{\|\mathbf{x}\|} &\leqslant \left(\max_{0 \leqslant j \leqslant p-1} \frac{|c_7(j)|}{\|\mathbf{x}\|}\right) |r^k| + \left(\max_{0 \leqslant j \leqslant p-1} \frac{|c_8(j)|}{\|\mathbf{x}\|}\right) |s^l| \\
&\leqslant t_7\,|r^k| + t_8\,|s^l|
\end{aligned}$$

If we let

$$k = \left\lceil \log_{|r|} \frac{\xi}{2} - \log_{|r|} t_7 \right\rceil \quad \text{and} \quad \left\lceil \log_{|s|} \frac{\xi}{2} - \log_{|s|} t_8 \right\rceil \tag{3.6}$$

then ξ is an upper bound of $\|\mathbf{x} - \mathbf{x}(k, l)\|/\|\mathbf{x}\|$.

The parallel algorithm derived in this section is similar to the one listed in Section 2, the only difference is to replace step 8 to step 8′ as shown here

8′. Compute $u = \max(u_{n_0}, u_{n_{p-1}})$, $v = \max(v_{n_0}, v_{n_{p-1}})$ /*see (3.5)*/ Compute t_7, t_8 /*see Appendix F*/ Compute k, l /*see (3.6) $\mathbf{x} \leftarrow \mathbf{z} - c_7(i)$ $(\underbrace{1, r, r^2, \ldots, r^{k-1}, 0, \ldots, 0}_{n_i})^T - c_8(i)(\underbrace{0, \ldots, 0, s^{l-1}, \ldots, s^2, s, 1}_{n_i})^T$. /*see (3.2)*/ From the truncated version of our parallel algorithm for solving (1.1), it needs $5n + 2k + 2l$, $0 < k$, $l \leqslant n/p$, FP operations; needs $\log p$ communication steps. The values of k and l are dependent on α, β, γ, α_1, β_1, γ_2, α_2, n, and p.

4. EXPERIMANTAL RESULTS

We have implemented this parallel algorithm in Section 3 for solving (1.1) on the nCUBE 2/E multiprocessors [cf. nCUBE 2 Processor Manual (1993); nCUBE 2 Programmer's Guide (1993)]. We test our parallel programs on this machine using 1, 2, 4, 8, and 16 processors, respectively. To balance the loads of all nodes, we let the data be distributed evenly among the nodes. Each node whose node-id is less than $n \bmod p$ processes $\lceil n/p \rceil$ data, and the other node processes $\lfloor n/p \rfloor$ data, respectively.

We use four sets of input data to demonstrate the performance of our algorithm. In Experiment 1, we let $\alpha = 3$, $\alpha_1 = 7.8$, $\beta = 1$, $\gamma = 1$, $\beta_1 = 0.6$, and $\gamma_2 = 0.8$; b_i's are generated randomly by the program. The execution time (T_n) and speedup (T_1/T_n) are listed in Table I. The sup-norm of the residual, $\|A\mathbf{x} - \mathbf{b}\|/\|b\|$, is in the order of 10^{-16}. In Experiment 2, we let $\alpha = 2.1$, $\alpha_1 = 7.8$, $\beta = 1$, $\gamma = 1$, $\beta_1 = 0.6$, $\gamma_2 = 0.8$. The experimental data are listed in Table II. The sup-norm of the residual is also in the order of 10^{-16}. In Experiment 3, we let $\alpha = 2.001$, $\alpha_1 = 7.8$, $\beta = 1$, $\gamma = 1$, $\beta_1 = 0.6$, $\gamma_2 = 0.8$. The experimental data are listed in Table III. The sup-norm of the residual is in the order of 10^{-13}. In Experiment 4, we let $\alpha = 2.00001$, $\alpha_1 = 7.8$, $\beta = 1$, $\gamma = 1$, $\beta_1 = 0.6$, $\gamma_2 = 0.8$. The experimental data are listed in Table IV. The sup-norm of the residual is in the order of 10^{-11}. In Tables I–IV, each entry has two numbers. The first number shows the

Table I. Execution Time (T_n, in msec) and Speedup (T_1/T_n) for $r = s = -0.381966$.

n	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$
100	1095	960	927	986	1104
	1	1.14	1.18	1.11	0.99
200	1759	1292	1185	1132	1202
	1	1.36	1.48	1.55	1.46
400	3076	1955	1516	1377	1320
	1	1.57	2.03	2.23	2.33
800	5739	3274	2178	1714	1584
	1	1.74	2.63	3.35	3.62
1600	10990	5935	3472	2371	1910
	1	1.85	3.17	4.64	5.75
3200	21517	11214	6155	3689	2599
	1	1.92	3.50	5.83	8.28
6400	42560	21713	11408	6358	3958
	1	1.96	3.73	6.69	10.75
12800	84676	42760	21906	11599	6575
	1	1.98	3.87	7.30	12.88

Table II. Execution Time (T_n, in msec) and Speedup (T_1/T_n) for $r = s = -0.7298$.

n	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$
100	1356	991	930	981	1102
	1	1.37	1.46	1.38	1.23
200	2174	1551	1214	1128	1204
	1	1.40	1.79	1.93	1.81
400	3497	2382	1777	1409	1344
	1	1.47	1.97	2.48	2.60
800	6155	3706	2618	1968	1610
	1	1.66	2.35	3.13	3.82
1600	11400	6364	3907	2813	2191
	1	1.79	2.92	4.05	5.20
3200	21943	11626	6598	4133	3019
	1	1.89	3.33	5.31	7.27
6400	42985	22146	11827	6795	4358
	1	1.94	3.63	6.33	9.86
12800	85089	43199	22370	12018	7017
	1	1.97	3.80	7.08	12.13

Table III. Execution time (T_n, in msec) and Speedup (T_1/T_n for $r = s = 0.9688$.

n	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$
100	1446	1110	1026	1081	1210
	1	1.30	1.41	1.34	120
200	2558	1637	1302	1223	1279
	1	1.56	1.96	2.09	2.00
400	4780	2750	1829	1493	1434
	1	1.74	2.61	3.20	3.22
800	9254	4974	2967	2046	1715
	1	1.86	3.12	4.52	5.40
1600	17875	9446	5191	3161	2248
	1	1.89	3.44	5.65	7.95
3200	28394	18360	9666	5410	3359
	1	1.55	2.94	5.25	8.45
6400	49435	28913	18552	9857	5598
	1	1.71	2.66	5.02	8.83
12800	91512	49949	29104	18743	10074
	1	1.83	3.14	4.88	9.08

Table IV. Execution Time (T_n, in msecs) and Speedup (T_1/T) for $r = s = 0.99684$.

n	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$
100	1446	1082	1023	1088	1212
	1	1.34	1.41	1.33	1.19
200	2557	1638	1303	1218	1289
	1	1.56	1.96	2.10	1.98
400	4780	2750	1857	1518	1431
	1	1.74	2.57	3.15	3.34
800	9254	4971	2969	2077	1693
	1	1.86	3.12	4.46	5.47
1600	18173	9446	5192	3165	2241
	1	1.92	3.50	5.74	8.11
3200	36104	18637	9663	5386	3377
	1	1.94	3.74	6.70	10.69
6400	71663	36207	18557	9854	5603
	1	1.98	3.86	7.27	12.79
12800	142990	71855	36397	18756	10107
	1	1.99	3.93	7.62	14.15

longest execution time among all processors. The second number represents the speedup ratio when compared with the time required when using one processor. It is observed that the parallel algorithm works well as the size of data, say q, processed by each node is greater than 50. As q is increasing, the speedup is almost linear (speedup is proportional to the number of nodes).

The number of FP operations required in step 8′ of this parallel algorithm depends on the absolute values of r and s. In Experiment 1, both r and s are small ($r = s = -0.381966$). Both c_7 and c_8 converge to 0 very fast. The total number of FP operations performed is closed to $5n/p$ although n is very small. In Experiment 2, both the absolute values of r and s ($r = s = -0.7289$) are closer to 1 than those in Experiment 1. It needs more than $5n/p$ arithmetic operations, but needs less than $9n/p$ operations. In Experiment 3, both the absolute values of r and s are closer to 1 too ($r = s = 0.9688$), thus the final solution converges slower. The total number FP operations required is closed to $9n/p$ while n/p is small; it becomes less than $9n/p$ as n becomes larger. In Experiment 4, both the absolute values of r and s are very closed to 1 ($r = s = -0.99684$) and the total number of operations required is closed to $9n/p$.

5. CONCLUSIONS

We have presented our parallel algorithm for solving circulant tridiagonal Toeplitz linear systems. The number of FP operations required in our algorithm is ranged from $5n/p$ to $9n/p$, which has the same order as that required in the sequential algorithm [cf. Yan and Chung (1994)]. On the nCUBE 2/E multicomputer, some experimental results are illustrated to demonstrate the good performance of our stable parallel solver.

APPENDIX A

A.1. Six Equalities

From $-r\gamma + a = \alpha$ and $r = -\beta/a$, we have $(\beta\gamma/a) + a = \alpha$, i.e., $a^2 - \alpha a + \beta\gamma = 0$. From $-r\gamma + a = \alpha$, we have $\alpha + \gamma r = a$. Hence we have $\alpha r + \gamma r^2 = ar$. From $r = -\beta/a$, it gives $\beta + \alpha r + \gamma r^2 = 0$. From $-r\gamma + a = \alpha$ and $s = -\gamma/a$, we have $\alpha - a = r - r\gamma = ars$. From $r = -\beta/a$, we then have $\alpha - a = ars = -\beta s$, $\beta s + \alpha = a$, and $\beta s^2 + \alpha s = as = -\gamma$, i.e., $\beta s^2 + \alpha s + \gamma = 0$.

APPENDIX B

B.1. One Equality

Consider

$$\mathbf{w} = \hat{A}_n \begin{pmatrix} \mathbf{z}^{(0)} \\ \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}^{(p-1)} \end{pmatrix} - \begin{pmatrix} \mathbf{b}^{(0)} \\ \mathbf{b}^{(1)} \\ \vdots \\ \mathbf{b}^{(p-1)} \end{pmatrix} = \hat{A}_n \begin{pmatrix} \mathbf{z}^{(0)} \\ \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}^{(p-1)} \end{pmatrix} - \begin{pmatrix} A'_{n_0}\mathbf{z}^{(0)} \\ A'_{n_1}\mathbf{z}^{(1)} \\ \vdots \\ A'_{n_{p-1}}\mathbf{z}^{(p-1)} \end{pmatrix}$$

then it yields to

$$\begin{aligned} w_{m_i} &= (\beta z_{m_i-1} + \alpha z_{m_i} + \gamma z_{m_i+1}) - (\beta z_{m_i-1} + \alpha z_{m_i}) \\ &= \gamma z_{m_i+1} \\ &= -as\bar{\mathbf{z}}^{(i)} \\ w_{m_i+1} &= \beta z_{m_i} + \alpha z_{m_i+1} + \gamma z_{m_i+2} - (az_{m_i+1} + \gamma z_{m_i+2}) \\ &= \beta z_{m_i} + (\alpha - a)\, z_{m_i+1} \\ &= -ar\underline{\mathbf{z}}^{(i-1)} + ars\bar{\mathbf{z}}^{(i)} \qquad \text{for} \quad 1 \leqslant i \leqslant p-1 \\ w_n &= (\beta z_{n-1} + az_n) - (\beta z_{n-1} + \alpha z_n) \\ &= (a - \alpha)\, z_n \\ &= -ars\underline{\mathbf{z}}^{(p-1)} \\ w_j &= 0 \qquad \text{otherwise} \end{aligned}$$

Equivalently, we have

$$\mathbf{w} = -ars\underline{\mathbf{z}}^{(p-1)}\mathbf{e}_n - \sum_{i=1}^{p-1} [as\bar{\mathbf{z}}^{(i)}\mathbf{e}_{m_i} + (ar\underline{\mathbf{z}}^{(i-1)} - ars\bar{\mathbf{z}}^{(i)})\, \mathbf{e}_{m_i+1}]$$

APPENDIX C

C.1. The Pseudo Code of *comm* for Hypercube

From

$$c_5(j) = \sum_{i=0}^{j} r^{m_{j+1}-m_{i+1}} g_i = r^{n_j} c_5(j-1) + g_j$$

$$c_6(j) = \sum_{i=j}^{p-1} s^{m_i - m_j} h_i = s^{n_j} c_6(j+1) + h_j,$$

it follows that

$$\begin{pmatrix} c_5(j) \\ 1 \end{pmatrix} = \begin{pmatrix} r^{n_j} & g_j \\ 0 & 1 \end{pmatrix} \begin{pmatrix} c_5(j-1) \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} c_5(0) \\ 1 \end{pmatrix} = \begin{pmatrix} r^{n_0} & g_0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} c_6(j) \\ 1 \end{pmatrix} = \begin{pmatrix} s^{n_j} & h_j \\ 0 & 1 \end{pmatrix} \begin{pmatrix} c_6(j+1) \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} c_6(p-1) \\ 1 \end{pmatrix} = \begin{pmatrix} s^{n_{p-1}} & h_{p-1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Let

$$X_j = \begin{pmatrix} r^{n_j} & g_j \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad Y_j = \begin{pmatrix} s^{n_j} & h_j \\ 0 & 1 \end{pmatrix}$$

two prefix-product forms are obtained as follows:

$$\begin{pmatrix} c_5(j) \\ 1 \end{pmatrix} = X_j X_{j-1} \cdots X_0 \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

and

$$\begin{pmatrix} c_6(j) \\ 1 \end{pmatrix} = Y_j Y_{j+1} \cdots Y_{p-1} \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

The pseudo code of function "*comm*" is listed next in order to guarantee that processor i in the hypercube keeps the value of $c_5(i-1)$, $c_5(p-1)$, $c_6(i+1)$, and $c_6(0)$. Note that in the following psuedo code, an overlapping technique is employed.

Function comm

$$X \leftarrow \begin{pmatrix} r^{n_i} & g_i \\ 0 & 1 \end{pmatrix}, \quad Y \leftarrow \begin{pmatrix} s^{n_i} & h_i \\ 0 & 1 \end{pmatrix}, \quad \hat{X} \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \hat{Y} \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

for k := 0 **to** $\log p - 1$ **do**
Send X and Y to the node whose node-id is different in bit position k;
Received these values from the sender and save it in X' and Y'
if bit k of node-id is 1 **then**

$$\hat{X} \leftarrow \hat{X}X', \qquad X \leftarrow XX', \qquad Y \leftarrow Y'Y$$

else

$$\hat{Y} \leftarrow \hat{Y}Y', \qquad X \leftarrow X'X, \qquad Y \leftarrow YY'$$

endif
endfor
/**********

$$X = X_{p-1}X_{p-2}\cdots X_0, \qquad Y = Y_0 Y_1 \cdots Y_{p-1},$$
$$\hat{X} = X_{i-1}\cdots X_1 X_0, \qquad \hat{Y} = Y_{i+1}Y_{i+2}\cdots Y_{p-1}$$

**********/

$$c_5(i-1) \leftarrow \hat{x}_{12}, \qquad c_5(p-1) \leftarrow x_{12}, \qquad c_6(i+1) \leftarrow \hat{y}_{12}, \qquad c_6(0) \leftarrow y_{12}$$

Suppose $p = 4$, we have

node-id	00	01	10	11
initial	$X = X_0, \hat{X} = I$	$X = X_1, \hat{X} = I$	$X = X_2, \hat{X} = I$	$X = X_3, \hat{X} = I$
state	$Y = Y_0, \hat{Y} = I$	$Y = Y_1, \hat{Y} = I$	$Y = Y_2, \hat{Y} = I$	$Y = Y_3, \hat{Y} = I$
$k = 0$	$X = X_1X_0, \hat{X} = I$	$X = X_1X_0, \hat{X} = X_0$	$X = X_3X_2, \hat{X} = I$	$X = X_3X_2, \hat{X} = X_2$
	$Y = Y_0Y_1, \hat{Y} = Y_1$	$Y = Y_0Y_1, \hat{Y} = I$	$Y = Y_2Y_3, \hat{Y} = Y_3$	$Y = Y_2Y_3, \hat{Y} = I$
$k = 1$	$X = X_3X_2X_1X_0$	$X = X_3X_2X_1X_0$	$X = X_3X_2X_1X_0$	$X = X_3X_2X_1X_0$
	$\hat{X} = I$	$\hat{X} = X_0$	$\hat{X} = X_1X_0$	$\hat{X} = X_2X_1X_0$
	$Y = Y_0Y_1Y_2Y_3$	$Y = Y_0Y_1Y_2Y_3$	$Y = Y_0Y_1Y_2Y_3$	$Y = Y_0Y_1Y_2Y_3$
	$\hat{Y} = Y_1Y_2Y_3$	$\hat{Y} = Y_2Y_3$	$\hat{Y} = Y_3$	$\hat{Y} = I$

APPENDIX D

D.1. The Proof of Lemma 1

Three cases for the vector $\mathbf{b}$ are to be considered. Consider the first case

$$\mathbf{b}=\begin{pmatrix}\mathbf{b}^{\langle 1\rangle}\\ \mathbf{b}^{\langle 2\rangle}\\ \mathbf{b}^{\langle 3\rangle}\end{pmatrix}$$

Let $J=(0 \quad I_{q\times q} \quad 0)_{q\times n}$ and it satisfies $J\mathbf{b}=\mathbf{b}^{\langle 2\rangle}$. From $\mathbf{A}'_q\mathbf{y}=\mathbf{b}^{\langle 2\rangle}$ and $A_n\mathbf{x}=\mathbf{b}$, we have

$$\frac{|\bar{\mathbf{y}}|}{\|\mathbf{x}\|}=\frac{|\mathbf{e}_1^TA_q'^{-1}\mathbf{b}^{\langle 2\rangle}|}{\|A_n^{-1}\mathbf{b}\|}=\frac{|\mathbf{e}_1^TA_q'^{-1}J\mathbf{b}|}{\|A_n^{-1}\mathbf{b}\|}\leqslant\max_{\mathbf{u}\neq 0}\frac{|\mathbf{e}_1^TA_q'^{-1}J\mathbf{u}|}{\|A_n^{-1}\mathbf{u}\|}$$

$$=\max_{\mathbf{v}\neq 0}\frac{|\mathbf{e}_1^TA_q'^{-1}JA_n\mathbf{v}|}{\|\mathbf{v}\|}=\|\mathbf{e}_1^TA_q'^{-1}JA_n\|_\infty$$

Since

$$\begin{aligned}A_q'^{-1}JA_n&=A_q'^{-1}(0 \quad I_{q\times q} \quad 0)\,A_n\\&=A_q'^{-1}(0 \quad \beta\mathbf{e}_1 \quad A'_q+(\alpha-a)\,\mathbf{e}_1\mathbf{e}_1^T \quad \gamma\mathbf{e}_q \quad 0)\\&=(0 \quad \beta A_q'^{-1}\mathbf{e}_1 \quad I+(\alpha-a)\,A_q'^{-1}\mathbf{e}_1\mathbf{e}_1^T \quad \gamma A_q'^{-1}\mathbf{e}_q \quad 0)\end{aligned}$$

we have

$$\begin{aligned}\frac{|\bar{\mathbf{y}}|}{\|\mathbf{x}\|}&\leqslant\|\mathbf{e}_1^TA_q'^{-1}JA_n\|_\infty\\&=\|(0 \quad \beta\mathbf{e}_q^TA_q'^{-1}\mathbf{e}_1 \quad (1+(\alpha-a)\,\mathbf{e}_1^TA_q'^{-1}\mathbf{e}_1)\,\mathbf{e}_1^T \quad \gamma\mathbf{e}_1^TA_q'^{-1}\mathbf{e}_q \quad 0)\|_\infty\\&\leqslant|\beta\pi_{11}|+|1+(\alpha-a)\,\pi_{11}|+|\gamma\pi_{1q}|=u_q^{(1)}\end{aligned}$$

and

$$\begin{aligned}\frac{|\mathbf{y}|}{\|\mathbf{x}\|}&\leqslant\|\mathbf{e}_q^TA_q'^{-1}JA_n\|_\infty\\&=\|(0 \quad \beta\mathbf{e}_q^TA_q'^{-1}\mathbf{e}_1 \quad \mathbf{e}_q^T+(\alpha-a)\,\mathbf{e}_q^TA_q'^{-1}\mathbf{e}_1\mathbf{e}_1^T \quad \gamma\mathbf{e}_q^TA_q'^{-1}\mathbf{e}_q \quad 0)\|_\infty\\&\leqslant|\beta\pi_{q1}|+1+|(\alpha-a)\,\pi_{q1}|+|\gamma\pi_{qq}|=v_q^{(1)}\end{aligned}$$

Consider the second case

$$\mathbf{b}=\begin{pmatrix}\mathbf{b}^{\langle 2\rangle}\\ \mathbf{b}^{\langle 3\rangle}\end{pmatrix}$$

Let $J=(I_{q\times q}\quad 0)_{q\times n}$ and it satisfies $J\mathbf{b}=\mathbf{b}^{\langle 2\rangle}$. Then it yields to

$$\begin{aligned}A_q'^{-1}JA_n&=A_q'^{-1}(I_{q\times q}\quad 0)\,A_n\\&=A_q'^{-1}(A_q'+(\alpha_1-a)\,\mathbf{e}_1\mathbf{e}_1^T\quad \gamma\mathbf{e}_q\quad 0\quad \beta_1\mathbf{e}_1)\\&=(I+(\alpha_1-a)\,A_q'^{-1}\mathbf{e}_1\mathbf{e}_1^T\quad \gamma A_q'^{-1}\mathbf{e}_q\quad 0\quad \beta_1 A_q'^{-1}\mathbf{e}_1)\end{aligned}$$

Therefore, we have

$$\begin{aligned}\frac{|\bar{\mathbf{y}}|}{\|\mathbf{x}\|}&\leqslant\|\mathbf{e}_1^TA_q'^{-1}JA_n\|_\infty\\&=\|((1+(\alpha_1-a)\,\mathbf{e}_1^TA_q'^{-1}\mathbf{e}_1)\,\mathbf{e}_1^T\quad \gamma\mathbf{e}_1^TA_q'^{-1}\mathbf{e}_q\quad 0\quad \beta_1\mathbf{e}_1^TA_q'^{-1}\mathbf{e}_1)\|_\infty\\&\leqslant|1+(\alpha_1-a)\,\pi_{11}|+|\gamma\pi_{1q}|+|\beta_1\pi_{11}|=u_q^{(2)}\end{aligned}$$

and

$$\begin{aligned}\frac{|\underline{\mathbf{y}}|}{\|\mathbf{x}\|}&\leqslant\|\mathbf{e}_q^TA_q'^{-1}JA_n\|_\infty\\&=\|(\mathbf{e}_q^T+(\alpha_1-a)\,\mathbf{e}_q^TA_q'^{-1}\mathbf{e}_1\mathbf{e}_1^T\quad \gamma\mathbf{e}_q^TA_q'^{-1}\mathbf{e}_q\quad 0\quad \beta_1\mathbf{e}_q^TA_q'^{-1}\mathbf{e}_1)\|_\infty\\&\leqslant 1+|(\alpha_1-a)\,\pi_{q1}|+|\gamma\pi_{qq}|+|\beta_1\pi_{q1}|=v_q^{(2)}\end{aligned}$$

Consider the third case

$$\mathbf{b}=\begin{pmatrix}\mathbf{b}^{\langle 1\rangle}\\ \mathbf{b}^{\langle 2\rangle}\end{pmatrix}$$

Let $J=(0\quad I_{q\times q})$ and it satisfies $J\mathbf{b}=\mathbf{b}^{\langle 2\rangle}$. We than have

$$\begin{aligned}A_q'^{-1}JA_n&=A_q'^{-1}(0\quad I_{q\times q})\,A_n\\&=A_q'^{-1}(\gamma_2\mathbf{e}_q\quad 0\quad \beta\mathbf{e}_1\quad A_q'+(\alpha-a)\,\mathbf{e}_1\mathbf{e}_1^T+(\alpha_2-\alpha)\,\mathbf{e}_q\mathbf{e}_q^T)\\&=(\gamma_2A_q'^{-1}\mathbf{e}_q\quad 0\quad \beta A_q'^{-1}\mathbf{e}_1\\&\qquad I+(\alpha-a)\,A_q'^{-1}\mathbf{e}_1\mathbf{e}_1^T+(\alpha_2-\alpha)\,A_q'^{-1}\mathbf{e}_q\mathbf{e}_q^T)\end{aligned}$$

Therefore, we have

$$\frac{|\bar{\mathbf{y}}|}{\|\mathbf{x}\|} \leqslant \|\mathbf{e}_1^T A_q'^{-1} J A_n\|_\infty$$

$$= \|(\gamma_2 \mathbf{e}_1^T A_q'^{-1} \mathbf{e}_q \quad 0 \quad \beta \mathbf{e}_1^T A_q'^{-1} \mathbf{e}_1$$

$$(1 + (\alpha - a)\, \mathbf{e}_1^T A_q'^{-1} \mathbf{e}_1)\, \mathbf{e}_1^T + (\alpha_2 - \alpha)\, \mathbf{e}_1^T A_q'^{-1} \mathbf{e}_q \mathbf{e}_q^T)\|_\infty$$

$$\leqslant |\gamma_2 \pi_{1q}| + |\beta \pi_{11}| + |1 + (\alpha - a)\, \pi_{11}| + |(\alpha_2 - \alpha)\, \pi_{1q}| = u_q^{(3)}$$

and

$$\frac{|\underline{\mathbf{y}}|}{\|\mathbf{x}\|} \leqslant \|\mathbf{e}_q^T A_q'^{-1} J A_n\|_\infty$$

$$= \|(\gamma_2 \mathbf{e}_q^T A_q'^{-1} \mathbf{e}_q \quad 0 \quad \beta \mathbf{e}_q^T A_q'^{-1} \mathbf{e}_1$$

$$(\alpha - a)\, \mathbf{e}_q^T A_q'^{-1} \mathbf{e}_1 \mathbf{e}_1^T + (1 + (\alpha_2 - \alpha)\, \mathbf{e}_q^T A_q'^{-1} \mathbf{e}_q)\, \mathbf{e}_q^T)\|_\infty$$

$$\leqslant |\gamma_2 \pi_{qq}| + |\beta \pi_{q1}| + |1 + (\alpha_2 - \alpha)\, \pi_{qq}| + |(\alpha - a)\, \pi_{q1}| = v_q^{(3)}$$

Finally, it yield to $|\bar{\mathbf{y}}|/\|\mathbf{x}\| \leqslant u_q$ and $|\underline{\mathbf{y}}|/\|\mathbf{x}\| \leqslant u_q$, where $u_q = \max(u_q^{(1)}, u_q^{(2)}, u_q^{(3)})$ and $v_q = \max(v_q^{(1)}, v_q^{(2)}, v_q^{(3)})$.

APPENDIX E

E.1. The evaluation of $\mathbf{e}_1^T A_q'^{-1} \mathbf{e}_1$, $\mathbf{e}_q^T A_q'^{-1} \mathbf{e}_1$, $\mathbf{e}_1^T A_q'^{-1} \mathbf{e}_q$, and $\mathbf{e}_q^T A_q'^{-1} \mathbf{e}_q$

To evaluate $\mathbf{e}_1^T A_q'^{-1} \mathbf{e}_1$ and $\mathbf{e}_q^T A_q'^{-1} \mathbf{e}_1$, we first solve $A_q' \mathbf{p} = \mathbf{e}_1$. That is, it wants to solve the recurrence relation: $\beta p_{i-1} + \alpha p_i + \gamma p_{i+1} = 0$ for $2 \leqslant i \leqslant q-1$ with the boundary conditions:

$$a p_1 + \gamma p_2 = 1 \qquad \text{and} \qquad \beta p_{q-1} + \alpha p_q = 0$$

From $\beta + \alpha r + \gamma r^2 = 0$ and $\beta s^2 + \alpha s + \gamma = 0$, r and $1/s$ are the zeros of characteristic polynomial of this recurrence relation. Subsequently, we obtain $p_k = c_1 r^{k-1} + c_2 s^{q-k}$. Putting it into boundary conditions, we have

$$a(c_1 + c_2 s^{q-1}) + \gamma(c_1 r + c_2 s^{q-2}) = 1$$

and

$$\beta(c_1 r^{q-2} + c_2 s) + \alpha(c_1 r^{q-1} + c_2) = 0$$

that is,

$$(a+\gamma r)\,c_1 = 1 \qquad \text{and} \qquad r^{q-2}(\beta+\alpha r)\,c_1 + (\alpha+\beta s)\,c_2 = 0$$

We then have

$$c_1 = \frac{1}{a+\gamma r} \qquad \text{and} \qquad c_2 = \frac{-r^{q-2}(\beta+\alpha r)}{(a+\gamma r)(\alpha+\beta s)}$$

Consequently, it follows that

$$\mathbf{e}_1^T A_q'^{-1}\mathbf{e}_1 = p_1 = c_1 + c_2 s^{q-1} = \frac{1}{a+\gamma r} - \frac{r^{q-2}s^{q-1}(\beta+\alpha r)}{(a+\gamma r)(\alpha+\beta d)}$$

$$\mathbf{e}_n^T A_q'^{-1}\mathbf{e}_1 = p_q = c_1 r^{q-1} + c_2 = \frac{r^{q-1}}{a+\gamma r} - \frac{r^{q-2}(\beta+\alpha r)}{(a+\gamma r)(\alpha+\beta s)}$$

To evaluate $\mathbf{e}_1^T A_q'^{-1}\mathbf{e}_q$ and $\mathbf{e}_q^T A_q'^{-1}\mathbf{e}_q$, we first solve $A_q'\mathbf{p} = \mathbf{e}_q$. Similarly, let $p_k = c_1 r^{k-1} + c_2 s^{q-k}$, we have

$$(a+\gamma r)\,c_1 = 0 \qquad \text{and} \qquad r^{q-2}(\beta+\alpha r)\,c_1 + (\alpha+\beta s)\,c_2 = 1$$

After some similar algebraic computations, it yields to

$$c_1 = 0 \qquad \text{and} \qquad c_2 = \frac{1}{\alpha+\beta s}$$

Consequently, it follows that

$$\mathbf{e}_1^T A_q'^{-1}\mathbf{e}_q = p_1 = c_1 + c_2 s^{q-1} = \frac{s^{q-1}}{\alpha+\beta s}$$

$$\mathbf{e}_q^T A_q'^{-1}\mathbf{e}_q = p_q = c_1 r^{q-1} + c_2 = \frac{1}{\alpha+\beta s}$$

APPENDIX F

F.1. A Upper Bound for the Relative Error

By (2.14) and (3.4), we have

$$\frac{|g_i|}{\|\mathbf{x}\|} = \frac{1}{|1-rs|}\,\frac{|\underline{\mathbf{z}}^{(i)}|}{\|\mathbf{x}\|} \leqslant t_1 \qquad \text{and} \qquad \frac{|h_i|}{\|\mathbf{x}\|} \leqslant \frac{|r|}{|1-rs|}\,\frac{|\underline{\mathbf{z}}^{(i)}|}{\|\mathbf{x}\|} + \frac{|\bar{\mathbf{z}}^{(i)}|}{\|\mathbf{x}\|} \leqslant t_2,$$

where

$$t_1 = \frac{v}{|1-rs|} \qquad \text{and} \qquad t_2 = \frac{|rv|}{|1-rs|} + u$$

By (2.15), it follows that

$$\frac{|c_5(j)|}{\|\mathbf{x}\|} \leqslant \sum_{i=0}^{j} \frac{|g_i|}{\|\mathbf{x}\|} \leqslant pt_1 \qquad \text{and} \qquad \frac{|c_6(j)|}{\|\mathbf{x}\|} \leqslant \sum_{i=j}^{p-1} \frac{|h_i|}{\|\mathbf{x}\|} \leqslant pt_2$$

By (2.13), it yields to

$$\frac{|c_3(j)|}{\|\mathbf{x}\|} = \frac{|rc_5(j-1)|}{\|\mathbf{x}\|} \leqslant |prt_1|$$

$$\frac{|c_4(j)|}{\|\mathbf{x}\|} \leqslant \frac{|rs|}{|rs-1|} \frac{|\underline{\mathbf{z}}^{(j)}|}{\|\mathbf{x}\|} + \frac{|sc_6(j+1)|}{\|\mathbf{x}\|} \leqslant |rst_1| + |spt_2|$$

By (2.17)

$$\frac{|x_1'|}{\|\mathbf{x}\|} \leqslant \frac{|c_6(0)|}{\|\mathbf{x}\|} \leqslant pt_2 \qquad \text{and} \qquad \frac{|x_n'|}{\|\mathbf{x}\|} \leqslant \frac{|c_5(p-1)|}{\|\mathbf{x}\|} \leqslant pt_1$$

By (2.16), we have

$$\frac{|f_1|}{\|\mathbf{x}\|} \leqslant |\alpha_1 - a| \frac{|x_1'|}{\|\mathbf{x}\|} + |\beta_1| \frac{|x_n'|}{\|\mathbf{x}\|} \leqslant t_3$$

and

$$\frac{|f_2|}{\|\mathbf{x}\|} \leqslant |\gamma_2| \frac{|x_1'|}{\|\mathbf{x}\|} + |\alpha_2 - a| \frac{|x_n'|}{\|\mathbf{x}\|} \leqslant t_4$$

where

$$t_3 = |\alpha_1 - a|\, pt_2 + |\beta_1|\, pt_1 \qquad \text{and} \qquad t_4 = |\gamma_2|\, pt_2 + |\alpha_2 - a|\, pt_1$$

By (2.20), it follows that

$$\frac{|c_1|}{\|\mathbf{x}\|} \leqslant \frac{|f_1 h_2 - f_2 h_1|}{|g_1 h_2 - g_2 h_1|\, \|\mathbf{x}\|} \leqslant t_2 \qquad \text{and} \qquad \frac{|c_2|}{\|\mathbf{x}\|} \leqslant \frac{|g_1 f_2 - g_2 f_1|}{|g_1 h_2 - g_2 h_1|\, \|\mathbf{x}\|} \leqslant t_6$$

where

$$t_5 = \frac{|t_3 h_2| + |t_4 h_1|}{|g_1 h_2 - g_2 h_1|} \qquad \text{and} \qquad t_6 = \frac{|g_1 t_4| + |g_2 t_3|}{|g_1 h_2 - g_2 h_1|}$$

By (2.22), it yields to

$$\frac{|c_7(j)|}{\|\mathbf{x}\|} \leqslant \frac{|c_3(j)| + |c_1|}{\|\mathbf{x}\|} \leqslant t_7 \qquad \text{and} \qquad \frac{|c_8(j)|}{\|\mathbf{x}\|} \leqslant \frac{|c_4(j)| + |c_2|}{\|\mathbf{x}\|} \leqslant t_8$$

where

$$t_7 = |prt_1| + t_5 \qquad \text{and} \qquad t_8 = |rst_1| + |spt_2| + t_6$$

REFERENCES

Hockney, R. W. (1965). A fast direct solution of Poisson's equation using Fourier analysis. *J. ACM* **12**, 95–113.

Widlund, O. B. (1972). On the use of fast methods for separable finite difference equations for the solution of general elliptic problems, in *Sparse Matrices and Their Applications*, Rose. D. J., and Willoughby, R. A. (eds.), Plenum Press, New York, pp. 121–131.

Fisher, D., Golub, G., Hald, O., Levia, C., and Winlund, O. (1974). On Fourier–Toeplitz methods for separable elliptic problems. *Mathematics of Computation* **28**, 349–368.

Smith, G. D. (1985). *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Third Edition, Oxford University Press.

Chung, K. L., and Yan, W. M. (1994). A fast algorithm for cubic B-spline curve fitting. *Comput. Graphics* **18** (3), 327–334.

Hirsh, R. (1975). Higher order accurate difference solutions of fluid mechanics problems by a compact differencing technique. *J. Comput. Phy.* **19**, 90–109.

Yan, W. M., and Chung, K. L. (1994). A fast algorithm for solving special tridiagonal systems, *Computing* **52**, 203–211.

Ranka, S., and Sahni, S. (1990). *Hupercube Algorithms with Applications to Image Processing and Pattern Recognition*, Springer-Verlag, Chap. 2, pp. 29–30.

nCUBE Company, (1993). *nCUBE 2 Processor Manual*, Foster City, California.

nCUBE Company, (1993). *nCUBE 2 Programmer's Guide*, Foster City, California.

Kim, H. J., and Lee, J. G. (1990). A parallel algorithm solving a tridiagonal Toeplitz linear system, *Parallel Computing* **13**, 289–294.