ELSEVIER

Parallel Computing 21 (1995) 1241–1267

PARALLEL COMPUTING

Application and accuracy of the parallel diagonal dominant algorithm ☆

Xian-He Sun *

Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803-4020, USA

Received 21 May 1993; revised 1 April 1994, 27 January 1995

Abstract

The Parallel Diagonal Dominant (PDD) algorithm is an efficient tridiagonal solver. In this paper, a detailed study of the PDD algorithm is given. First the PDD algorithm is extended to solve periodic tridiagonal systems and its scalability is studied. Then the reduced PDD algorithm, which has a smaller operation count than that of the conventional sequential algorithm for many applications, is proposed. Accuracy analysis is provided for a class of tridiagonal systems, the symmetric and skew-symmetric Toeplitz tridiagonal systems. Implementation results show that the analysis gives a good bound on the relative error, and the PDD and reduced PDD algorithms are good candidates for emerging massively parallel machines.

Keywords: Parallel processing; Parallel numerical algorithms; Scalable computing; Tridiagonal systems; Toeplitz systems

1. Introduction

Solving tridiagonal systems is one of the key issues in computational fluid dynamics (CFD) and many other scientific applications [20,11]. Many methods used for the solution of partial differential equations (PDEs) rely on solving a sequence of tridiagonal systems. The alternating direction implicit (ADI) method [16] requires solution of tridiagonal systems alternately in each coordinate direction. Discretization of partial differential equations by compact difference schemes

☆ This research was supported in part by the National Aeronautics and Space Administration under NASA contract NAS1-19480 and NAS1-1672/MMP.

* Email: sun@bit.csc.lsu.edu

SSDI 0167-8191(95)00018-6

[18] also leads to a sequence of tridiagonal systems. Tridiagonal systems also arise in multigrid methods, wavelet collocation method, and in ADI or line-SOR preconditioners for conjugate gradient methods. In addition to PDE's, tridiagonal systems also arise in many other applications [1].

Solving tridiagonal systems is inexpensive on sequential machines. However, because of their sequential data dependencies, tridiagonal systems are more difficult to solve efficiently on parallel computers. For this reason, intensive research has been done on the development of efficient parallel tridiagonal solvers. Many algorithms have been proposed [13,8]. Among them, the recursive doubling reduction method (RCD), developed by Stone [15], and the cyclic reduction or odd-even reduction method (OER), developed by Hockney [9], are able to solve an n-dimensional tridiagonal system in $O(\log n)$ time using n processors. These are effective algorithms for fine grained computing. Later, several algorithms were proposed for medium and coarse grain computing, i.e. for the case of $p < n$ or $p \ll n$, where p is the number of processors available [5,10,22]. The algorithm of Lawrie and Sameh [10] and the algorithm of Wang [22] can be considered substructuring methods (or, in a more general term, divided-and-conquer method). These algorithms partition the original problem into sub-problems. The sub-problems are then solved in parallel, and their solutions combined to form the final solution.

Recently, Sun, Zhang, and Ni [20] proposed three new parallel algorithms for solving tridiagonal systems. All three algorithms are substructuring methods and are based on Sherman-Morrison matrix modification formula [3]. Two of these, the proposed parallel partition LU (PPT) algorithm and the parallel hybrid (PPH) algorithm are fast and able to incorporate limited pivoting. The PPT algorithm is a good candidate when the number of processors, p, is small, while the PPH algorithm is a better choice when p is large.

The third algorithm proposed is the parallel diagonal dominant (PDD) algorithm, designed for strictly diagonally dominant problems. This PDD algorithm is the most efficient. Compared with other tridiagonal solvers, which require at least $O(\log p)$ communication phases, the PDD algorithm has only a small fixed communication cost independent of the number of processors, and requires only a slightly more computation than that in the best sequential algorithm. In fact, the PDD algorithm is perfectly scalable, in the sense that the communication cost and the computation overhead do not increase with the problem size or with the number of processors available [19].

Scalability has become an important metric of parallel algorithms [6,19]. The PDD algorithm is perfectly scalable and highly efficient, making it an ideal algorithm on massively parallel architectures, for problems to which it is applicable. However, the PDD algorithm is relatively new and applicable only in certain conditions. In this paper we give a detailed study of the PDD algorithm. We study application of the PDD algorithm and provide a formal accuracy analysis for Toeplitz tridiagonal systems. The PDD algorithm described here is slightly different from the algorithm proposed in [20]. A detailed study is provided here for new applications, such as periodic systems, and systems with multiple right hand sides.

More importantly, as a result of this study, we were led to a new algorithm, the reduced PDD algorithm. This algorithm, also described here, maintains the same communication cost as the PDD algorithm, but has a smaller operation count. In fact, its operation count is even smaller than that of the Thomas algorithm [16], the conventional sequential algorithm, for periodic systems with multiple right hand sides. This low operation count makes the reduced PDD algorithm an effective algorithm for many systems arising in PDEs.

This paper is organized as follows. Section 2 introduces the sequential and parallel PDD algorithms. The applications of the PDD algorithm are discussed in Section 3. This section also gives the variant of the PDD algorithm needed for periodic systems and the reduced PDD algorithm. Section 4 gives an accuracy analysis for both the PDD and reduced PDD algorithms. Experimental results on an Intel/iPSC860 multicomputer are presented in Section 5. Finally, Section 6 provides conclusions and final remarks.

2. The parallel diagonal dominant algorithm

We are interested in solving a tridiagonal linear system of equations

$$\mathbf{Ax} = \mathbf{d}, \tag{1}$$

where A is a tridiagonal matrix of order n

$$A = \begin{bmatrix} b_0 & c_0 & & & & \\ a_1 & b_1 & c_1 & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & a_{n-2} & b_{n-2} & c_{n-2} \\ & & & & a_{n-1} & b_{n-1} \end{bmatrix} = [a_i, b_i, c_i], \tag{2}$$

$x = (x_0, \cdots, x_{n-1})^T$ and $d = (d_0, \cdots, d_{n-1})^T$. We assume that A, x, and d have real coefficients. Extension to the complex case is straightforward.

2.1. The matrix partition technique

To solve Eq. (1) efficiently on parallel computers, we partition A into submatrices. For convenience we assume that $n = p \cdot m$, where p is the number of processors available. The matrix A in Eq. (2) can be written as

$$A = \tilde{A} + \Delta A,$$

where $\tilde{A}$ is a block diagonal matrix with diagonal submatrices $A_i (i = 0, \cdots, p-1)$. The submatrices $A_i (i = 0, \cdots, p-1)$ are $m \times m$ tridiagonal matrices. Let e_i be a column vector with its ith $(0 \le i \le n-1)$ element being one and all the other entries being zero. We have

$$\Delta A = [a_m e_m, c_{m-1} e_{m-1}, a_{2m} e_{2m}, c_{2m-1} e_{2m-1}, \cdots, c_{(p-1)m-1} e_{(p-1)m-1}] \begin{bmatrix} e_{m-1}^T \\ e_m^T \\ \cdot \\ \cdot \\ e_{(p-1)m-1}^T \\ e_{(p-1)m}^T \end{bmatrix} = VE^T,$$

where both V and E are $n \times 2(p-1)$ matrices. Thus, we have

$$A = \tilde{A} + VE^T.$$

The Sherman and Morrison [14] matrix modification formula for rank-one updates, subsequently generalized by Woodbury [23], enables us to solve Eq. (1), assuming that all of the A_i's are invertible. We have

$$x = A^{-1} d = (\tilde{A} + VE^T)^{-1} d, \tag{3}$$

$$x = \tilde{A}^{-1} d - \tilde{A}^{-1} V (1 + E^T \tilde{A}^{-1} V)^{-1} E^T \tilde{A}^{-1} d. \tag{4}$$

Note that I is an identity matrix and $Z = I + E^T \tilde{A}^{-1} V$ is a pentadiagonal matrix of order $2(p-1)$. Let

$$\tilde{A}\tilde{x} = d \tag{5}$$

$$\tilde{A} Y = V \tag{6}$$

$$h = E^T \tilde{x} \tag{7}$$

$$Z = I + E^T Y \tag{8}$$

$$Zy = h \tag{9}$$

$$\Delta x = Yy. \tag{10}$$

Eq. (4) becomes

$$x = \tilde{x} - \Delta x. \tag{11}$$

In Eqs. (5) and (6), $\tilde{x}$ and Y are solved by the LU decomposition method. By the structure of $\tilde{A}$ and V, this is equivalent to solving

$$A_i [\tilde{x}^{(i)}, v^{(i)}, w^{(i)}] = [d^{(i)}, a_{im} e_0, c_{(i+1)m-1} e_{m-1}], \tag{12}$$

$i = 0, \cdots, p-1$. Here $\tilde{x}^{(i)}$ and $d^{(i)}$ are the ith block of $\tilde{x}$ and d, respectively, and $v^{(i)}$, $w^{(i)}$ are potentially nonzero column vectors of the ith row block of Y. Eq. (12) implies that we only need to solve three linear systems of order m with the same LU decomposition for each i $(i = 0, \cdots, p-1)$. In addition, we can skip the first $m-1$ forward substitutions for the third system, since the first $m-1$ components of the vector on the right-hand side are all zeros. h and Z are obtained without computation.

2.2. *The PDD algorithm*

Solving Eq. (9) is the major computation involved in the conquer part of our algorithms. Several different approaches have been proposed for solving this equation, resulting in several different algorithms for solving tridiagonal systems [20]. The matrix Z in Eq. (9) has the form

$$Z = \begin{bmatrix} 1 & w_{m-1}^{(0)} & 0 & & & & & \\ v_0^{1} & 1 & 0 & w_0^{(1)} & & & & \\ v_{m-1}^{(1)} & 0 & 1 & w_{m-1}^{(1)} & 0 & & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & 1 & 0 & w_0^{(p-2)} \\ & & & & v_{m-1}^{(p-2)} & 0 & 1 & w_{m-1}^{(p-2)} \\ & & & & & 0 & v_0^{(p-1)} & 1 \end{bmatrix}$$

where $v^{(i)}$, $w^{(i)}$ for $i = 0, \cdots, p-1$ are solutions of Eq. (12) and the 1's come from the identity matrix I. In practice, the magnitude of the last component of $v^{(i)}$, $v_{m-1}^{(i)}$, and the first component of $w^{(i)}$, $w_0^{(i)}$, may be smaller than machine accuracy when $p \ll n$, especially for diagonal dominant tridiagonal systems. In this case, $w_0^{(i)}$ and $v_{m-1}^{(i)}$ can be dropped, and Z becomes a block diagonal system consisting of $(p-1)$ independent 2×2 blocks. Thus, Eq. (9) can be solved efficiently on parallel computers, which leads to the highly efficient *parallel diagonal dominant* (PDD) algorithm.

In the sequential PDD algorithm, since Y has at most two nonzero entries in every row, and Z is a diagonal block matrix with 1's as diagonal elements, (9) takes five arithmetic operations per row, and the evaluation of (10) takes four operations per row. Thus we conclude that the sequential PDD algorithm takes $17n - 9n/p - 4p - 9$ arithmetic operations.

The PDD algorithm, using p processors, consists of the following steps:

Step 1. Allocate A_i, $d^{(i)}$, and elements a_{im}, $c_{(i+1)m-1}$ to the ith node, where $0 \le i \le p-1$.
Step 2. Solve (12). All computations can be executed in parallel on p processors.
Step 3. Send $\tilde{x}_0^{(i)}$, $v_0^{(i)}$ from the ith node to the $(i-1)$th node, for $i = 1, \cdots, p-1$.
Step 4. Solve

$$\begin{bmatrix} 1 & w_{m-1}^{(i)} \\ v_0^{(i+1)} & 1 \end{bmatrix} \begin{pmatrix} y_{2i} \\ y_{2i+1} \end{pmatrix} = \begin{pmatrix} \tilde{x}_{m-1}^{(i)} \\ \tilde{x}_0^{(i+1)} \end{pmatrix}$$

in parallel on the ith node for $0 \le i \le p-2$. Then send y_{2i} from the ith node to the $(i+1)$th node, for $i = 0, \cdots, p-2$.

Step 5. Compute (10) and (11). We have

$$\Delta x^{(i)} = [v^{(i)}, w^{(i)}]\begin{pmatrix} y_{(2i-1)} \\ y_{2i} \end{pmatrix}, \; x^{(i)} = \tilde{x}^{(i)} - \Delta x^{(i)}.$$

In all of these operations, each processor communicates only with its two neighboring processors.

Though recent advances in interprocess communication, such as circuit switching and wormhole routing, have reduced communication costs, it remains an intrinsic overhead in parallel computing. Moreover, the improvement in communication capabilities has been quite modest compared with the dramatic improvements in processing speed. For most distributed-memory computers, the time to communicate with nearest neighbors is found to vary linearly with problem size [4]. Let S be the number of bytes to be transferred. Then the transfer time to communicate with a neighbor can be expressed as $\alpha + S\beta$, where α is a fixed startup time and β is the incremental transmission time per byte. Assuming 4 bytes are used for each real number, Steps 3 and 4 take $\alpha + 8\beta$ and $\alpha + 4\beta$ time respectively. The parallel PDD algorithm has $17n/p - 4$ parallel operations and has communication time $2(\alpha + 6\beta)$.

2.3. Scalability analysis

As parallel machines with more and more processors are becoming available, the performance metric *scalability* is becoming more and more important. The question is how an algorithm will perform when the problem size is scaled linearly with the number of processors. Let $T(p, W)$ be the execution time for solving a system with W work (problem size) on p processors. The ideal situation would be that when both the number of processors and the amount of work are scaled up by a factor of N, the execution time remains unchanged:

$$T(N \times p, N \times W) = T(p, W) \tag{13}$$

How to define problem size, in general, is a question still being debated. However, it is commonly agreed that the floating point (flop) operation count is a good estimate of problem size for scientific computations. To eliminate the effect of numerical inefficiencies in parallel algorithms, in practice the flop count is based upon some practical optimal sequential algorithm. In our case, the Thomas algorithm [16], the LU decomposition method for tridiagonal systems, was chosen as the sequential algorithm. It takes $8n - 7$ floating point operations, where the 7 can be neglected for large n. As the problem size W increases N times to W', we have

$$W' = N \times 8n = 8n'$$
$$n' = N \cdot n.$$

Let τ_{comp} represent the unit of a computation operation normalized to the communication time. The time required to solve (1) by the PDD algorithm with p processors is

$$T(p, W) = \left(17\frac{n}{p} - 4\right)\tau_{comp} + 2(\alpha + 6\beta),$$

and

$$\begin{aligned} T(N \times p, N \times W) &= \left(17\frac{n'}{N \cdot p} - 4\right)\tau_{comp} + 2(\alpha + 6\beta) \\ &= \left(17\frac{N \cdot n}{N \cdot p} - 4\right)\tau_{comp} + 2(\alpha + 6\beta) \\ &= \left(17\frac{n}{p} - 4\right)\tau_{comp} + 2(\alpha + 6\beta) \\ &= T(p, W). \end{aligned}$$

Thus the PDD algorithm is perfectly scalable. Similar arguments can be applied to periodic systems (see Section 3) with the same result.

Eq. (13) is true if and only if the average unit speed of the given computing system is a constant, where average unit speed is defined as the quotient of the achieved speed of the given computing system and the number of processors. Scalability has been formally defined [19] as the ability to maintain the average unit speed. Let W be the amount of work of an algorithm when p processors are employed in a machine, and let W' be the amount of work of the algorithm when $p' = N \cdot p$ processors are employed to maintain the average speed, then the scalability from system size p to system size $N \cdot p$ of the algorithm-machine combination is defined as

$$\psi(p, N \times p) = \frac{N \cdot p \cdot W}{p \cdot W'} = \frac{N \cdot W}{W'}.$$

The average unit speed can be represented as

$$A\_S(p, W) = \frac{W}{p \cdot T(p, W)},$$

where W is the problem size, p is the number of processors, and $T(p, W)$ is the corresponding execution time. From our previous discussion of the PDD algorithm, when $W' = N \cdot W$, we have $T(N \times p, W') = T(p, W)$. Therefore

$$\begin{aligned} A\_S(N \times p, W') &= \frac{W'}{N \cdot p \cdot T(N \times p, W')} = \frac{W'}{N \cdot p \cdot T(p, W)} \\ &= \frac{N \cdot p \cdot W}{N \cdot T(p, W)} = \frac{W}{p \cdot T(p, W)}. \end{aligned}$$

That is $W' = N \cdot W$ has maintained the average unit speed, and the scalability is

$$\psi(p, N \times p) = \frac{N \cdot W}{W'} = \frac{N \cdot W}{N \cdot W} = 1.$$

Thus the PDD algorithm is perfectly scalable.

3. Special applications

In this section, we first discuss some tridiagonal systems arising in CFD applications, the *symmetric* and *skew-symmetric Toeplitz tridiagonal systems*. Then two variants of the PDD algorithm, the *reduced PDD algorithm* and the PDD algorithm for periodic systems, will be presented.

3.1. Toeplitz tridiagonal systems

A Toeplitz tridiagonal matrix has the form

$$A = \begin{bmatrix} \eta & \gamma & & & \\ \lambda & \eta & \gamma & & \\ & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \gamma \\ & & & \lambda & \eta \end{bmatrix} = [\lambda, \eta, \gamma].$$

Symmetric Toeplitz tridiagonal systems often arise in solving partial differential equations and in other scientific applications. The compact finite difference scheme [18] is a relative new scheme for discretizing PDE's. Because of its simplicity and high accuracy, it is beginning to be widely used in practice.

Using the compact scheme, the general approximation of a first derivative has the form:

$$\beta f'_{i-2} + \alpha f'_{i-1} + f'_i + \alpha f'_{i+1} + \beta f'_{i+2}$$
$$= c\frac{f_{i+3} - f_{i-3}}{6h} + b\frac{f_{i+2} - f_{i-2}}{4h} + a\frac{f_{i+1} - f_{i-1}}{2h}$$

Letting

$$\alpha = \frac{1}{3}, \quad \beta = 0, \quad a = \frac{14}{9}, \quad b = \frac{1}{9}, \quad c = 0,$$

the scheme becomes formally sixth order accurate and the resulting system is $[\frac{1}{3}, 1, \frac{1}{3}]$, a symmetric Toeplitz tridiagonal system. Similarly, the general approximation of a second derivative has the form

$$\beta f''_{i-2} + \alpha f''_{i-1} + f''_i + \alpha f''_{i+1} + \beta f''_{i+2}$$
$$= c\frac{f_{i+3} - 2f_i + f_{i-3}}{9h^2} + b\frac{f_{i+2} - 2f_i + f_{i-2}}{4h^2} + a\frac{f_{i+1} - 2f_i + f_{i-1}}{h^2}.$$

For

$$\alpha = \frac{2}{11}, \quad \beta = 0, \quad a = \frac{12}{11}, \quad b = \frac{3}{11}, \quad c = 0,$$

a sixth order difference scheme is obtained, and the tridiagonal system is symmetric and Toeplitz, $[\frac{2}{11}, 1, \frac{2}{11}]$.

Now discretizing in time, the one dimensional wave equation $u_t = a \cdot u_x$ and the heat equation $u_t = a \cdot u_{xx}$, where u_t, u_x and u_{xx} are the differential of u on time and on space, can be represented as

$$u^{n+1} = u^n + \Delta t \cdot a \cdot u_x^n,$$

and

$$u^{n+1} = u^n + \Delta t \cdot a \cdot u_{xx}^n$$

respectively. Using the compact scheme, u_x^n and u_{xx}^n are defined by symmetric Toeplitz tridiagonal systems. Therefore, the solutions can be obtained by solving a sequence of symmetric Toeplitz tridiagonal systems.

Skew-symmetric Toeplitz tridiagonal systems also arise in solving PDEs [16]. For instance, to solve the advection equation $u_t + a \cdot u_x = f$, we begin with the formula

$$u_t = \frac{u(t+k, x) - u(t, x)}{k} + O(k^2)$$

for u_t evaluated at $(t + \frac{1}{2}k, x)$. We also use the relation

$$\begin{aligned} u_x\left(t + \frac{1}{2}k, x\right) &= \frac{u_x(t+k, x) + u_x(t, x)}{2} + O(k^2) \\ &= \frac{1}{2}\left[\frac{u(t+k, x+h) - u(t+k, x-h)}{2h} + \frac{u(t, x+h) - u(t, x-h)}{2h}\right] \\ &\quad + O(k^2) + O(h^2). \end{aligned}$$

Using these approximations for $u_t + a \cdot u_x = f$ about $(t + \frac{1}{2}k, x)$, we obtain

$$\frac{v_m^{n+1} - v_m^n}{k} + a\frac{v_{m+1}^{n+1} - v_{m-1}^{n+1} + v_{m+1}^n - v_{m-1}^n}{4h} = \frac{f_m^{n+1} + f_m^n}{2}$$

or, equivalently,

$$\frac{a\lambda}{4}v_{m+1}^{n+1} + v_m^{n+1} - \frac{a\lambda}{4}v_{m-1}^{n+1} = -\frac{a\lambda}{4}v_{m+1}^n + v_m^n + \frac{a\lambda}{4}v_{m-1}^n + \frac{k}{2}\left(f_m^{n+1} + f_m^n\right),$$

where $\lambda = k/h$. The left side is an *skew-symmetric Toeplitz tridiagonal matrix*, $A = [\frac{a\lambda}{4}, 1, \frac{-a\lambda}{4}]$.

3.2. Periodic tridiagonal systems

Many PDE's arising in real applications have periodic boundary conditions. For instance, to study a physical phenomenon in an infinite region, we often model only a small subdomain, applying periodic boundary conditions on the boundary. The resulting linear systems have the form of

$$A=\begin{bmatrix} b_0 & c_0 & & & & & a_0 \\ a_1 & b_1 & c_1 & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & a_{n-2} & b_{n-2} & c_{n-2} \\ c_{n-1} & & & & & a_{n-1} & b_{n-1} \end{bmatrix},$$

and are called *periodic tridiagonal systems*. On sequential machines, periodic tridiagonal systems are solved by combining the solutions of two different right hand sides [7], which increases the operation count from $8n-7$ to $14n-16$.

The PDD algorithm can be extended to periodic tridiagonal systems. The difference is that, after dropping $w_0^{(i)}$, and $v_{m-1}^{(i)}$, the matrix Z becomes a periodic system of order $2p$:

$$Z=\begin{bmatrix} 1 & w_{(m-1)}^{(0)} & & & & v_0^{(0)} \\ v_0^{(1)} & 1 & 0 & & & \\ & 0 & 1 & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & w_{m-1}^{(p-2)} \\ w_{m-1}^{(p-1)} & & & & v_0^{(p-1)} & 1 \end{bmatrix}$$

The dimension of Z is slightly higher than in the non-periodic case, which simply makes the load on the 0th and $(p-1)$th processor identical to the load on all of the other processors. The parallel computation time remains the same. For periodic systems, the communication at step 3 and 4 changes from one dimensional array communication to ring communication. The communication time is also unchanged for any architecture supporting ring communication. Fig. 1 shows the communication pattern of the PDD algorithm for periodic systems.

3.3. The reduced PDD algorithm

In the last step, Step 5, of the PDD algorithm, the final solution, x, is computed by combining the intermediate results concurrently on each processor,

$$x^{(k)}=\tilde{x}^{(k)}-y_{(2k-1)}v^{(k)}-y_{2k}w^{(k)},$$

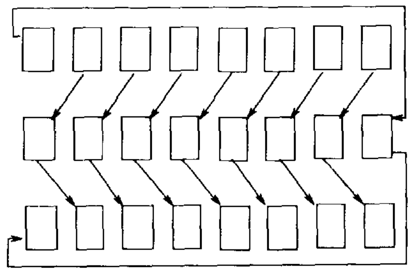

Fig. 1. Communication pattern for solving periodic systems.

which requires $4(n-1)$ operations in total and $4m$ parallel operations, if $p = n/m$ processors are used. The PDD algorithm drops the first element of w, w_0, and the last element of v, v_{m-1}, in solving Eq. (9). In Section 4.1–4.2 we will show that, for symmetric and skew-symmetric Toeplitz tridiagonal systems, the w_0 and v_{m-1} can be dropped when m is large without affecting the accuracy of the final solution. Furthermore, we will show (Eq. (27))

$$v = \frac{1}{\lambda\left(a + b\sum_{i=0}^{m-1} b^{2i}\right)} \left(\sum_{i=0}^{m-1} b^{2i}, \sum_{i=1}^{m-1} b^{2i}/(-b), \cdots, (-b)^{m-1} \right)^T.$$

So, when m is large enough, we may drop v_i, $i = j, j+1, \cdots, m-1$, and w_i, $i = 0, 1, \cdots, j-1$, for some integer $j > 0$, while maintaining the required accuracy. If we replace v_i by $\tilde{v}_i$, where $\tilde{v}_i = v_i$ for $i = 0, 1, \cdots, j-1$, $\tilde{v}_i = 0$, for $i = j, \cdots, m-1$; and replace w by $\tilde{w}$, where $\tilde{w}_i = w_i$ for $i = j, \cdots, m-1$, and $\tilde{w}_i = 0$, for $i = 0, 1, \cdots, j-1$; and use $\tilde{v}$, $\tilde{w}$ in Step 5, we have Step 5'

$$\begin{aligned} \Delta x^{(k)} &= [\tilde{v}, \tilde{w}] \begin{pmatrix} y_{(2k-1)} \\ y_{2k} \end{pmatrix} \\ x^{(k)} &= \tilde{x}^{(k)} - \Delta x^{(k)}. \end{aligned} \tag{14}$$

This requires only $4\frac{j}{p}$ parallel operation. Replacing Step 5 of the PDD algorithm by Step 5', we get the reduced PDD algorithm which requires $13\frac{n}{p} - 4(j+1)$ parallel computations. The key question here is how to find the smallest integer $j > 0$ which will maintain the required accuracy. This question is answered in Section 4.3, where a formal accuracy study of the reduced PDD algorithm is given and a simple formula provided to compute the value of j.

4. Accuracy analysis

The PDD algorithm is highly efficient, perfectly scalable, but is only applicable when the intermediate results $v_{m-1}^{(i)}$, $w_0^{(i)}$, $0 \le i \le p-1$, can be dropped. However this dropping may lead to an inaccurate solution. Thus an accuracy study is essential in applying the PDD algorithm. A preliminary study of the accuracy of the PDD algorithm has already been done [24,2]. However, that study was for the general case and provides only sufficient conditions to guarantee a given accuracy. Unfortunately, the conditions given in [24,2] are difficult to verify, and the accuracy bound given is quite loose. In this section we focus on a particular class of tridiagonal systems, *symmetric and skew-symmetric Toeplitz tridiagonal systems*, where a tighter analysis can be given.

There are four steps in our analysis. First, we give the decay rate of $w_0^{(i)}$, $v_{m-1}^{(i)}$, $i = 0, \cdots, p-1$, for symmetric Toeplitz tridiagonal systems. These are the entries treated as zeros by the PDD algorithm. Second, the accuracy of the PDD algorithm is studied, again for symmetric Toeplitz tridiagonal systems. Next, we analyze the accuracy of the reduced PDD algorithm. Finally, we extend the above results to skew-symmetric Toeplitz tridiagonal systems.

4.1. The decay rate of v_{m-1} and w_0

Symmetric Toeplitz tridiagonal systems have the form $A = [\lambda, \eta, \lambda] = \lambda[1, c, 1]$, where $c = \eta/\lambda$. We assume the matrix A is diagonal dominant. That is we assume $|c| > 2$. To study the accuracy of the solution of $Ax = b$, we first study the matrix

$$\tilde{B} = \begin{pmatrix} a & 1 & & & \\ 1 & c & 1 & & \\ & 1 & \cdot & \cdot & \\ & & \cdot & \cdot & 1 \\ & & & 1 & c \end{pmatrix} = \begin{pmatrix} 1 & & & & \\ b & 1 & & & \\ & b & \cdot & & \\ & & \cdot & \cdot & \\ & & & b & 1 \end{pmatrix} \begin{pmatrix} a & 1 & & & \\ & a & \cdot & & \\ & & \cdot & \cdot & \\ & & & \cdot & 1 \\ & & & & a \end{pmatrix},$$

where a and b are the real solutions of

$$b + a = c, \quad b \cdot a = 1. \tag{15}$$

Since $a \cdot b = 1$ and $|c| > 2$, we may further assume that $|a| > 1$, and $|b| < 1$.

The LDL^T decomposition of $\tilde{B}$ is

$$\tilde{B} = [b, 1, 0] \times [0, a, 0] \times [0, 1, b].$$

Thus

$$\begin{aligned} \tilde{B}^{-1} &= [0, 1, b]^{-1} \times [0, a, 0]^{-1} \times j[b, 1, 0]^{-1} \\ &= \begin{pmatrix} 1 & -b & b^2 & \cdot & (-b)^{n-1} \\ & 1 & -b & \cdot & (-b)^{n-2} \\ & & \cdot & \cdot & \cdot \\ & & & 1 & -b \\ & & & & 1 \end{pmatrix} \begin{pmatrix} a^{-1} & & & \\ & a^{-1} & & \\ & & \cdot & \\ & & & \cdot \\ & & & & a^{-1} \end{pmatrix} \\ &\quad \times \begin{pmatrix} 1 & & & & \\ -b & 1 & & & \\ b^2 & -b & 1 & & \\ \cdot & \cdot & \cdot & \cdot & \\ (-b)^{n-1} & \cdot & \cdot & -b & 1 \end{pmatrix}. \end{aligned} \tag{16}$$

Let $d=(1,0,\cdots,0)^T$. Then

$$\tilde{B}^{-1}d=\frac{1}{a}\left(\sum_{i=0}^{n-1}b^{2i},\ \sum_{i=1}^{n-1}b^{2i}/(-b),\ \sum_{i=2}^{n-1}b^{2i}/b^2,\ldots,\ \sum_{i=n-1}^{n-1}b^{2i}/(-b)^{n-1}\right)^T.$$

Let

$$\Delta B=\begin{pmatrix} b & 0 & \cdot & \cdot & \cdot & 0\\ 0 & 0 & \cdot & \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot\\ 0 & 0 & \cdot & \cdot & \cdot & 0\end{pmatrix}=\begin{pmatrix} b\\ 0\\ \cdot\\ \cdot\\ 0\end{pmatrix}(1,0,\cdots,0)=\tilde{V}\tilde{E}^T, \tag{17}$$

and

$$B=\tilde{B}+\Delta B=[1,c,1].$$

Then, by the matrix modification formula (4), the solution of $By=d$ is

$$\begin{aligned} y=B^{-1}d&=\left(\tilde{B}+\tilde{V}\tilde{E}^T\right)^{-1}d\\ &=\tilde{B}^{-1}d-\tilde{B}^{-1}\tilde{V}\left(I+\tilde{E}^T\tilde{B}^{-1}\tilde{V}\right)^{-1}\tilde{E}^T\tilde{B}^{-1}d,\end{aligned}$$

where

$$\left(I+\tilde{E}^T\tilde{B}^{-1}\tilde{V}\right)^{-1}=\frac{a}{a+b\sum_{i=0}^{n-1}b^{2i}},$$

$$\tilde{E}^T\tilde{B}^{-1}d=\frac{\sum_{i=0}^{n-1}b^{2i}}{a},$$

$$\tilde{B}^{-1}\tilde{V}=\frac{b}{a}\left(\sum_{i=0}^{n-1}b^{2i},\sum_{i=1}^{n-1}b^{2i}/(-b),\cdots,\sum_{i=n-1}^{n-1}b^{2i}/(-b)^{n-1}\right)^T,$$

and

$$\tilde{B}^{-1}\tilde{V}\left(I+\tilde{E}^T\tilde{B}^{-1}\tilde{V}\right)^{-1}\tilde{E}^T\tilde{B}^{-1}d=\frac{b}{a}\cdot\frac{\sum_{i=0}^{n-1}b^{2i}}{a+b\sum_{i=0}^{n-1}b^{2i}}\begin{pmatrix}\sum_{i=0}^{n-1}b^{2i}\\ \cdot\\ \cdot\\ (-b)^{n-1}\end{pmatrix}.$$

The last element of y is

$$y_{n-1}=\frac{(-b)^{n-1}}{a}-\frac{(-b)^{n-1}}{a}\cdot\frac{b\sum_{i=0}^{n-1}b^{2i}}{a+b\sum_{i=0}^{n-1}b^{2i}}=\frac{(-b)^{n-1}}{a}\left(\frac{a}{a+b\sum_{i=0}^{n-1}b^{2i}}\right) \tag{18}$$

$$=\frac{(-b)^{n-1}}{a}\left(\frac{1}{1+b^2\sum_{i=0}^{n-1}b^{2i}}\right)\quad(\textit{note } a\cdot b=1). \tag{19}$$

Thus,

$$|y_{n-1}| \le \frac{|b|^{n-1}}{|a|} = |b|^n.$$

The first element of y is

$$y_0 = \frac{\sum_{i=0}^{n-1} b^{2i}}{a}\left(\frac{1}{1+b^2\sum_{i=0}^{n-1} b^{2i}}\right) = \frac{b(1-b^{2n})}{1-b^{2(n+1)}},$$

$$|y_0| = \left|\frac{b(1-b^{2n})}{1-b^{2(n+1)}}\right| < |b|.$$

For the original system $Ax = d$, $A = \lambda[1, c, 1]$, the first element of x is

$$x_0 = \frac{y_0}{\lambda}.$$

The last element of x is

$$x_{n-1} = \frac{y_{n-1}}{\lambda}.$$

Using similar arguments, we can prove that for $d = (0, \ldots 0, 1)^T$, $Ax = d$ has solution

$$x_i = \frac{y_{n-(i+1)}}{\lambda}. \tag{20}$$

In particular

$$x_{n-1} = \frac{y_0}{\lambda}, \quad x_0 = \frac{y_{n-1}}{\lambda}.$$

Combining these we have the following lemma:

Lemma 1. *For any diagonal dominant, symmetric Toeplitz tridiagonal matrix* $A = [\lambda, \eta, \lambda]$, *the first element and the last element of the solution of equations* $Ax = d$ *is less than* $|b^{m-1}/\lambda a|$ *for* $d = (0, \ldots, 0, 1)$ *and* $d = (1, 0, \ldots, 0)$ *respectively, where* a *and* b *are the solutions of Eq.* (15) *and* n *is the order of matrix* A.

The decay rate analysis given in this section is based on the matrix modification formula (4), which is the basic tool used in this study. Different approaches are available. For instance, the convergence rate of the sequences taken from the diagonals of the LU factorization of a tridiagonal, Toeplitz matrix [12] can be used, instead of the modification formula, in the analysis to achieve the same results.

4.2. Accuracy of the PDD algorithm

Since for Toeplitz tridiagonal systems, each submatrix A_i, $i = 0, \cdots, p-1$, has the same structure as A, based on Lemma 1, we have the following theorem:

Theorem 1. *For any diagonal dominant, symmetric Toeplitz tridiagonal matrix* $A = [\lambda, \eta, \lambda]$, *if* b^{m-1}/a *is less than machine accuracy, where* a *and* b *are the solutions of equation* (15), *then the PDD algorithm approximates the true solution to within machine accuracy.*

Proof. By the structure of matrix $\tilde{A}$ and V, for any symmetric Toeplitz matrix $A = [\lambda, \eta, \lambda]$, solving Eq. (6) is equivalent to solving

$$A_i[v^{(i)}, w^{(i)}] = [\lambda e_0, \lambda e_{m-1}], \tag{21}$$

$i = 0, \cdots, p-1$ (see Eq. (12)). Solving Eq. (21) is in turn equivalent to solving

$$A'_i[v^{(i)}, w^{(i)}] = [e_0, e_{m-1}],$$

$i = 0, \cdots, p-1$, where $A'_i = [1, \eta/\lambda, 1]$. By lemma 1, if b^{m-1}/a is less than machine accuracy, the last element of solution $v^{(i)}$ and the first element of solution $w^{(i)}$, $i = 0, \cdots, p-1$, are less than machine accuracy. That is these elements can be dropped from matrix Z (see Eq. (9)) without influencing the accuracy of the final solutions, which concludes our proof. □

Theorem 1 says that if v_{m-1}, w_0 are less than machine accuracy, the PDD algorithm gives a satisfactory solution. In most scientific applications, the accuracy requirement is much weaker than machine accuracy. We now study how the decay rate of v_{m-1}, w_0 influences the accuracy of the final solution. Our study starts at the matrix partition formula (4).

Let

$$y = (I + E^T\tilde{A}^{-1}V)^{-1}E^T\tilde{A}^{-1}d. \tag{22}$$

Substituting y into Eq. (4), we have

$$\begin{aligned} x &= \tilde{A}^{-1}d - \tilde{A}^{-1}Vy, \\ E^Tx &= E^T\tilde{A}^{-1}d - E^T\tilde{A}^{-1}V \cdot y \\ &= (I + E^T\tilde{A}^{-1}V)y - E^T\tilde{A}^{-1}V \cdot y = y. \end{aligned} \tag{23}$$

Let y^* be the corresponding solution of the PDD algorithm,

$$y^* = (I + E^T\tilde{A}^{-1}V - D)^{-1}E^T\tilde{A}^{-1}d,$$

where D is the $2(p-1) \times 2(p-1)$ matrix which contains all the $v_{m-1}^{(i)}$, $w_0^{(i)}$ elements. Combined with Eq. (22) we have

$$(I + E^T\tilde{A}^{-1}V)y - (I + E^T\tilde{A}^{-1}V - D)y^* = 0.$$

That is

$$(y^* - y) = (I + E^T\tilde{A}^{-1}V - D)^{-1}D \cdot y.$$

Let x^* be the corresponding final solution of the PDD algorithm. Then

$$x^* = \tilde{A}^{-1}d - \tilde{A}^{-1}V \cdot y^*,$$

$$x - x^* = \tilde{A}^{-1}V(y^* - y)$$

$$= \tilde{A}^{-1}V\left(I + E^T\tilde{A}^{-1}V - D\right)^{-1} D \cdot E^T x.$$

Thus,

$$\frac{\| x - x^* \|}{\| x \|} \le \| \tilde{A}^{-1}V\left(I + E^T\tilde{A}_V - D\right)^{-1} DE^T \|. \tag{24}$$

The inequality (24) holds for general tridiagonal systems. In the following we assume the special structure of symmetric Toeplitz tridiagonal system in computing the norm of the right hand side. We use the 1-norm in our study. As discussed in the last section,

$$\left(I + E^T\tilde{A}^{-1}V - D\right)^{-1} = \begin{pmatrix} Z_0^{-1} & & & & \\ & Z_1^{-1} & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & Z_{p-1}^{-1} \end{pmatrix},$$

where Z_i are 2×2 matrices:

$$Z_i = \begin{pmatrix} 1 & w_{m-1}^{(i)} \\ v_0^{(i)} & 1 \end{pmatrix}$$

For symmetric Toeplitz tridiagonal systems $v_0^{(i)} = w_{m-1}^{(i)} = v_0^{(0)} = \tilde{a}$, and $v_{m-1}^{(i)} = w_0^{(i)} = v_{m-1}^{(0)} = \tilde{b}$, for i = 0,$\cdots$, p − 1. So, for our applications,

$$Z_i = Z_1 = \begin{pmatrix} 1 & \tilde{a} \\ \tilde{a} & 1 \end{pmatrix}, \quad Z_1^{-1} = \frac{1}{1 - \tilde{a}^2}\begin{pmatrix} 1 & -\tilde{a} \\ -\tilde{a} & 1 \end{pmatrix}.$$

$D \cdot E^T$ stretches D from a $2(p-1) \times 2(p-1)$ matrix to a $2(p-1) \times n$ matrix. Each column of $D \cdot E^T$ is either identically zero or contains only one non-zero element, $\tilde{b}$. Also, $(I + E^T\tilde{A}^{-1}V - D)^{-1}DE^T$ is a $2(p-1) \times n$ matrix. Each of its columns is either identically zero or contains at most two non-zero elements c_1, c_2, where

$$\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = Z_1^{-1}\begin{pmatrix} 0 \\ w_0^{(i)} \end{pmatrix} = Z_1^{-1}\begin{pmatrix} 0 \\ \tilde{b} \end{pmatrix} = \frac{\tilde{b}}{1 - \tilde{a}^2}\begin{pmatrix} -\tilde{a} \\ 1 \end{pmatrix},$$

and

$$\begin{pmatrix} c_2 \\ c_1 \end{pmatrix} = Z_1^{-1}\begin{pmatrix} v_{m-1}^{(i)} \\ 0 \end{pmatrix} = Z_1^{-1}\begin{pmatrix} \tilde{b} \\ 0 \end{pmatrix} = \frac{\tilde{b}}{1 - \tilde{a}^2}\begin{pmatrix} 1 \\ -\tilde{a} \end{pmatrix}.$$

For our application $A_i = A_1$, and $a_0^{(i)} = c_{m-1}^{(i)} = \lambda$, $i = 0, \cdots, p-1$. So, $v^{(i)} = v, w^{(i)} = w, i = 0, \cdots, p-1$ (see Eq. (12)). $(\tilde{A}^{-1}V)(I + E^T\tilde{A}^{-1}V - D)^{-1}D \cdot E^T$ is an $n \times n$ matrix, with each column being either a zero column or containing only c_1w, c_2v, or c_2w, c_1v respectively. Thus,

$$\begin{aligned}
&\| \tilde{A}^{-1}V\left(I + E^T A_V - D\right)^{-1} DE^T \| \\
&\quad \leq max\{ \| c_2 v \| + \| c_1 w \|, \| c_1 v \| + \| c_2 w \| \} \\
&\quad \leq | c_2 | \| w \| + | c_1 | \| v \| \quad (note \ \| w \| = \| v \|, \ Eq.(20)) \\
&\quad = (| c_2 | + | c_1 |) \| v \| = \frac{| \tilde{b} |}{|1 - \tilde{a}^2|}(1 + | \tilde{a} |) \| v \| = \frac{| \tilde{b} |}{|1 - | \tilde{a} ||} \| v \|.
\end{aligned} \tag{25}$$

From the results given in Section 4.1 (see Eq. (18)),

$$| \tilde{a} | = \left| \frac{b(1 - b^{2m})}{\lambda(1 - b^{2(m+1)})} \right| \leq \left| \frac{b}{\lambda} \right|, \quad | \tilde{b} | \leq \left| \frac{b^m}{\lambda} \right|.$$

Defining,

$$v = \frac{1}{\lambda a}\left(1 - \frac{b\Sigma_{i=0}^{m-1} b^{2i}}{a + b\Sigma_{i=0}^{m-1} b^{2i}}\right)\left(\sum_{i=0}^{m-1} b^{2i}, \sum_{i=1}^{m-1} b^{2i}/(-b), \cdots, (-b)^{m-1}\right)^T \tag{26}$$

$$= \frac{1}{\lambda\left(a + b\Sigma_{i=0}^{m-1} b^{2i}\right)}\left(\sum_{i=0}^{m-1} b^{2i}, \cdots, (-b)^{m-1}\right)^T, \tag{27}$$

we have

$$\begin{aligned}
\| v \| &= \left| \frac{1 - b^2}{\lambda(a - b^{2m+1})} \right| \left| \sum_{i=0}^{m-1} \right| \left| \frac{(-b)^i(1 - b^{2(m-i)})}{1 - b^2} \right| \\
&\leq \left| \frac{1 - b^2}{\lambda(a - b^{2m+1})} \right| \frac{(1 + | b |^{m+1})(1 - | b |^m)}{(1 - b^2)(1 - | b |)} \\
&\leq \frac{1}{| \lambda a |} \left| \frac{(1 + | b |^{m+1})}{1 - b^{2(m+1)}} \right| \left| \frac{(1 - | b |^m)}{(1 - | b |)} \right| \\
&\leq \frac{1}{| \lambda a |} \cdot \frac{1 - | b |^m}{1 - | b |^{m+1}} \cdot \frac{1}{1 - | b |} \quad (note \ | a | > 1, | b | < 1) \\
&\leq \frac{1}{| \lambda | (| a | - 1)}.
\end{aligned} \tag{28}$$

Combining the inequalities (24), (25), and (28) we obtain the final result

$$\frac{\| x - x^* \|}{\| x \|} \leq \frac{| \tilde{b} |}{| \lambda(1 - | \tilde{a} |) | \times (| a | - 1)}, \tag{29}$$

and

$$\frac{\| x - x^* \|}{\| x \|} \le \frac{|b|^m}{|\lambda^2(1-|\tilde{a}|)|(|a|-1)} = \frac{|b|^m}{\left|\lambda\left(|\lambda| - \left|\frac{b(1-b^{2m})}{1-b^{2(m+1)}}\right|\right)\right|(|a|-1)}. \tag{30}$$

Inequality (29) shows how the values of v_{m-1} and w_0 influence the accuracy of the final results. Inequality (30) gives an error bound of the PDD algorithm. When $|b/\lambda| < 1$, inequality (30) can be simplified to

$$\frac{\| x - x^* \|}{\| x \|} \le \frac{|b|^m}{|\lambda|(|\lambda| - |b|)(|a|-1)}.$$

4.3. Accuracy of the reduced PDD algorithm

Let $\tilde{v}$, $\tilde{w}$ be the vectors defined in Eq. (14), let $\tilde{V}$ be the corresponding matrix in Eq. (6) consisting of all the $2(p-1)$ vectors, and let x' be the solution of the reduced PDD algorithm. Then

$$x' = \tilde{A}^{-1}d - \tilde{A}^{-1}\tilde{V}(I + E^T\tilde{A}^{-1}V)E^T\tilde{A}^{-1}d.$$

As in Section 4.2, we let $y = (I + E^T\tilde{A}^{-1}V)E^T\tilde{A}^{-1}d$. Notice that x^* is the solution of the PDD algorithm. By Eqs. (4) and (23),

$$x' - x^* = (\tilde{A}^{-1}\tilde{V} - \tilde{A}^{-1}V)y = (\tilde{A}^{-1}\tilde{V} - \tilde{A}^{-1}V)E^Tx.$$

Therefore, for a given integer $j > 0$,

$$\frac{\| x' - x^* \|}{\| x \|} \le \|(\tilde{A}^{-1}\tilde{V} - \tilde{A}^{-1}V)\| \cdot \| E^T \| = \| \tilde{A}^{-1}\tilde{V} - \tilde{A}^{-1}V \| = \| \tilde{v} - v \|$$

$$= \left|\frac{1}{\lambda\left(a + b\sum_{i=0}^{m-1} b^{2i}\right)}\right| \left|\sum_{i=j}^{m-1}\right| \left|\frac{(-b)^i(1-b^{2(m-i)})}{1-b^2}\right|.$$

Since

$$\sum_{i=j}^{m-1}\left|\frac{b^i(1-b^{2(m-i)})}{1-b^2}\right| \le \frac{1}{|1-b^2|}\left(\sum_{i=j}^{m-1}|b|^i + \sum_{i=j}^{m-1}|b|^{2m-i}\right)$$

$$= \frac{1}{|1-b^2|}\left(\frac{|b|^j(1-|b|^m)}{1-|b|} + |b|^m\frac{1-|b|^{m-j+1}}{1-|b|}\right)$$

$$= \frac{\left(|b|^j(1-|b|^m) + |b|^m(1-|b|^{m-j+1})\right)}{|1-b^2|(1-|b|)},$$

$$\frac{\| x' - x^* \|}{\| x \|} \le \left|\frac{1}{\lambda a}\right| \left|\frac{1-b^2}{1-b^{2(m+1)}}\right| \cdot \frac{\left(|b|^j(1-|b|^m) + |b|^m(1-|b|^{m-j+1})\right)}{|(1-b^2)|(1-|b|)}$$

$$\le \frac{(|b|^j + |b|^m)}{|\lambda(|a|-1)}.$$

By (30), inequality (32) gives the error bound of the reduced PDD algorithm.

$$\frac{\| x - x' \|}{\| x \|} \le \frac{\| x - x^* \|}{\| x \|} + \frac{\| x^* - x' \|}{\| x \|} \tag{31}$$

$$\le \frac{| b |^m}{\left| \lambda \left(| \lambda | - \left| \frac{b(1 - b^{2m})}{1 - b^{2(m+1)}} \right| \right) \right| (| a | - 1)} + \frac{| b |^j + | b |^m}{| \lambda | (| a | - 1)}. \tag{32}$$

For a given error tolerance $\epsilon > 0$, the right hand side of inequality (32)

$$\frac{| b |^m}{\left| \lambda \left(| \lambda | - \left| \frac{b(1 - b^{2m})}{1 - b^{2(m+1)}} \right| \right) \right| (| a | - 1)} + \frac{| b |^j + | b |^m}{| \lambda | (| a | - 1)} < \epsilon$$

if and only if

$$j > \frac{\log\left(| \lambda | (| a | - 1) \left(\epsilon - \frac{| b |^m}{| \lambda | (| a | - 1)} \left(1 + 1 / \left(| \lambda | - \left| \frac{b(1 - b^{2m})}{1 - b^{2(m+1)}} \right| \right) \right) \right) \right)}{\log | b |}. \tag{33}$$

Inequality (33) gives a lower bound of the number of variables, j, need to be modified in Eq. (14), for a given error tolerance $\epsilon > 0$.

When $| b / \lambda | < 1$

$$\frac{\| x - x' \|}{\| x \|} \le \frac{| b |^m}{| \lambda | (| a | - 1)} \left(1 + \frac{1}{| \lambda | - | b |} \right) + \frac{| b |^j}{| \lambda | (| a | - 1)},$$

and we get a simpler inequality for the minimal number j,

$$j > \left(\frac{\log(| \lambda | (| a | - 1) \left(\epsilon - | b |^m | \lambda | (| a | - 1) \left(1 + \frac{1}{| \lambda | - | b |} \right) \right)}{\log | b |} \right). \tag{34}$$

When $| b^m |$ is less than machine accuracy, inequality (33) becomes the same as inequality (34) and we have an even simpler formula:

$$j > \frac{\log(| \lambda | (| a | - 1) \epsilon}{\log | b |} \tag{35}$$

4.4. Skew-symmetric Toeplitz tridiagonal systems

The accuracy analysis given by Sections 4.1–4.2 is for symmetric Toeplitz tridiagonal systems. In this section we extend the results to skew-symmetric Toeplitz tridiagonal systems. We assume that $m = n/p$ is an even number.

A skew-symmetric Toeplitz tridiagonal matrix A has the form $A=[-\lambda, \eta, \lambda] = \lambda \cdot [-1, c, 1]$. Let $B=[-1, c, 1]$. Then, for the corresponding matrix $\tilde{B}$ (see Section 4.1, Eq. (16))

$$\tilde{B}=[b, 1, 0] \times [0, a, 1] \times [0, 1, -b],$$

where a, b are the solutions of

$$b+a=c, \quad b \cdot a=-1. \tag{36}$$

Compared with the symmetric case, the difference is that we have $-b$ in the matrix [0, 1, $-$b] and $b \cdot a=-1$ in Eq. (36). Following the steps given in the study of symmetric systems, we have computed the vectors of v and w in Eq. (12),

$$\begin{aligned}
v &= \frac{1}{\lambda a}\left(1-\frac{b\Sigma_{i=0}^{m-1}(-1)^i b^{2i}}{a+b\Sigma_{i=0}^{m-1}(-1)^i b^{2i}}\right) \\
&\quad \times \left(\sum_{i=0}^{m-1}(-1)^i b^{2i}, \sum_{i=1}^{m-1}(-1)^i b^{2i}/(b), \cdots, (-b)^{m-1}\right)^T \\
&= \frac{1}{\lambda\left(a+b\Sigma_{i=0}^{m-1}(-1)^i b^{2i}\right)}\left(\sum_{i=0}^{m-1}(-1)^i b^{2i}, \cdots, (-b)^{m-1}\right)^T; \qquad (37) \\
w &= \frac{1}{\lambda\left(a+b\Sigma_{i=0}^{m-1}(-1)^i b^{2i}\right)} \\
&\quad \times \left((-1)^{m-1}(-b)^{m-1}, (-1)^{m-2}\sum_{i=m-2}^{m-1}(-1)^i b^{2i}/b^i, \cdots, \sum_{i=0}^{m-1}(-1)^i b^{2i}\right)^T.
\end{aligned}$$

We can see for skew-symmetric Toeplitz tridiagonal systems $v_0^{(i)}=w_{m-1}^{(i)}=v_0^{(0)}=\tilde{a}$, and $v_{m-1}^{(i)}=-w_0^{(i)}=v_{m-1}^{(0)}=\tilde{b}$, for $i=0, \cdots, p-1$. Thus, the inequality (25) remains true for skew-symmetric cases.

By Eq. (37), we have

$$\tilde{b}=v_{m-1}^{(i)}=\frac{(-b)^{m-1}}{\lambda\left(a+b\Sigma_{i=0}^{m-1}(-1)^i b^{2i}\right)}=\frac{(-b)^m \cdot (1+b^2)}{\lambda(1+b^{2(m+1)})},$$

$$|\tilde{b}| \le \frac{|b^m(1+b^2)|}{|\lambda|};$$

and

$$\tilde{a}=v_0^{(i)}=\frac{\Sigma_{i=0}^{m-1}(-1)^i b^{2i}}{\lambda\left(a+b\Sigma_{i=0}^{m-1}(-1)^i b^{2i}\right)}=\frac{-b \cdot (1-b^{2m})}{\lambda(1+b^{2(m+1)})},$$

$$|\tilde{a}|=\left|\frac{-b \cdot (1-b^{2m})}{\lambda(1+b^{2(m+1)})}\right| \le \frac{|b|}{\lambda}.$$

For the bound of the norm of vector v (see Eq. (28)), when $b \cdot a = -1$,

$$\| v \| \le \left| \frac{1}{\lambda a} \left(\frac{a}{a + b\sum_{i=0}^{m-1}(-1)^i b^{2i}} \right) \right| \left| \frac{(1+|b|)^{m+1}(1-|b|^{2m})}{(1+b^2)(1-|b|)} \right|$$

$$\le \frac{1}{|\lambda a(1+b^{2(m+1)})|} \cdot \frac{1-|b|^{2(m+1)}}{1-|b|} \le \frac{1}{|\lambda(|a|-1)|}.$$

The corresponding relative error is

$$\frac{\| x - x^* \|}{\| x \|} \le \frac{|\tilde{b}|}{|\lambda(1-|\tilde{a}|)(|a|-1)|}$$

in terms of $\tilde{a}$ and $\tilde{b}$; and

$$\frac{\| x - x^* \|}{\| x \|} \le \frac{|b|^m(1+b^2)}{|\lambda^2(1-|\tilde{a}|)(|a|-1)|} = \frac{|b|^m(1+b^2)}{|\lambda(|\lambda| - \left|\frac{b(1-b^{2m})}{1+b^{2(m+1)}}\right|(|a|-1)|}$$

in terms of a and b. When $|b|/|\lambda| < 1$, we have

$$\frac{\| x - x^* \|}{\| x \|} \le \frac{|b|^m(1+b^2)}{|\lambda(|\lambda|-|b|)(|a|-1)|}.$$

5. Experimental results

Table 1 gives the computation and communication count of the PDD algorithm. Tridiagonal systems arising in both ADI and in the compact scheme methods are multiple right-side systems. They are usually 'kernels' in much larger codes. It is often more efficient to use a parallel tridiagonal solver for these systems than to

Table 1
Computation and communication counts of the PDD algorithm

System	Matrix	Best sequential	The PDD	
			Computation	Communication
Single system	Non-periodic	$8n-7$	$17\frac{n}{p}-4$	$2\alpha+12\beta$
	Periodic	$14n-16$	$17\frac{n}{p}-4$	$2\alpha+12\beta$
Multiple right sides	Non-periodic	$(5n-3)\cdot n1$	$\left(9\frac{n}{p}+1\right)\cdot n1$	$(2\alpha+8n1\cdot\beta)$
	Periodic	$(7n-1)\cdot n1$	$\left(9\frac{n}{p}+1\right)\cdot n1$	$(2\alpha+8n1\cdot\beta)$

Table 2
Computation and communication counts of the reduced PDD

System	The Reduced PDD	
	Computation	Communication
Single system	$11\frac{n}{p}+6j-4$	$2\alpha+12\beta$
Multiple right sides	$\left(5\frac{n}{p}+4j+1\right)\cdot n1$	$(2\alpha+8n1\cdot\beta)$

remap data among processors to be able to a serial solve, especially for distributed-memory machines where communication cost is high. The computation and communication count for solving multiple right-side systems is also listed in Table 1, in which the factorization of matrix $\tilde{A}$ and computation of Y are not considered (see Eqs. (5) and (6) in Section 2). Parameter $n1$ is the number of right hand sides. Note for multiple right-side systems, the communication cost increases with the number of right hand sides. Table 2 gives the computation and communication counts of the reduced PDD algorithm. As for the PDD algorithm, it has the same parallel computation and communication counts for periodic and non-periodic systems. The computation saving of the reduced PDD algorithm is not only in step 5, the final modification step. Since we only need j elements of vector v and w for the final modification in the reduced PDD algorithm (see Eq. (14) in Section 3), we only need to compute j elements for each column of V in solving Eq. (6). The integer j is given by (33), (34), or (35) depending on the particular circumstance. In general, j is quite small. For instance, when error tolerance ϵ equals 10^{-4}, j equals either 10 or 7 for λ, the magnitude of the off diagonal elements equals 1/3 or 1/4 respectively. The integer j reduces to 4 for $0<\lambda\leq 1/9$. Notice that when $j<n/2$, the reduced PDD algorithm has a smaller operation count than that of Thomas algorithm for periodic systems with multiple right hand sides.

While the accuracy analyses given in this study are for Toeplitz tridiagonal systems, the PDD algorithm and the reduced PDD algorithm can be applied for solving general tridiagonal systems. The computation counts given in Table 1 and 2 are for general tridiagonal systems. For symmetric Toeplitz tridiagonal systems, a fast method proposed by Malcolm and Palmer [12] has a smaller computation count than Thomas algorithm for systems with single right hand side. It only requires $5n+2k-3$ arithmetic, where k is a decay parameter depending on the diagonal dominancy of the system. Formulas are available to compute the upper and lower bounds of parameter k [12]. The computation saving of Malcolm and Palmer's method is in the the LU decomposition. For systems with multiple right hand sides, in which the factorization cost is not considered, the Malcolm and Palmer's method and Thomas method have the same computation count. Table 3 gives the computation and communication counts of the PDD and reduced PDD algorithms based on Malcolm and Palmer's algorithm. The computation counts of the two algorithms have reduced with the fast method being used in solving the sub-systems. Table 3 is for solving systems with single right hand side only. For

Table 3
Computation and communication counts for symmetric toeplitz systems

Algorithm	Matrix	Best sequential	Parallel algorithm	
			Computation	Communication
PDD Algorithm	Non-periodic	$5n+2k-3$	$14\frac{n}{p}+2k$	$2\alpha+12\beta$
	Periodic	$11n+2k-12$	$14\frac{n}{p}+2k$	$2\alpha+12\beta$
reduced PDD Alg.	Non-periodic	$5n+2k-3$	$8\frac{n}{p}+2k+6j$	$2\alpha+8\beta$
	Periodic	$11n+2k-12$	$8\frac{n}{p}+2k+6j$	$2\alpha+8\beta$

systems with multiple right hand sides, the computation counts remain the same as in Table 1 and 2 for PDD and reduced PDD algorithm respectively.

As an illustration of the algorithm and theoretical results given by previous sections, a sample matrix is tested here. This sample matrix

$$A=\left[\frac{1}{3},\ 1,\ \frac{1}{3}\right] \tag{38}$$

arises in the compact scheme, we have

$$A=\left[\frac{1}{3},\ 1,\ \frac{1}{3}\right]=\frac{1}{3}\cdot[1,3,1]=\frac{1}{3}\cdot([b,1,0]\times[0,a,0]\times[0,1,b]-\Delta B),$$

where ΔB is given by Eq. (17), and

$$\lambda=\frac{1}{3},\quad c=3,\quad a=\frac{3+\sqrt{5}}{2},\quad b=\frac{3-\sqrt{5}}{2}. \tag{39}$$

The PDD algorithm was first implemented on a Sun Sparc4 to solve the corresponding non-periodic system of $Ax=d$ for accuracy checking. Then the algorithm, without taking the advantage of the Toeplitz structure, was implemented on a 32-node Intel/iPSC860 to measure the speedup over the Thomas algorithm [7], the standard sequential algorithm for periodic tridiagonal systems. For accuracy checking, all the measured and predicted data have been converted to a common logarithm scale to make the difference visible. Fig. 2 depicts the decay rate of v_{m-1} of matrix A, where the x-coordinate is the order of the sub-system A_i and the y-coordinate is the value of v_{m-1}. We can see that the theoretical bound given in Section 4.1 matches the measured value closely.

Accuracy comparisons of the PDD and the reduced PDD algorithms are given in Fig. 3 and Fig. 4 for periodic and non-periodic systems $Ax=d$ respectively. For the accuracy comparisons, the right-side vector, d, was randomly generated. The x-coordinate is the order of matrix A_i, and the y-coordinate is the relative error in the 1-norm. These two figures show that our accuracy analysis provides a very good bound.

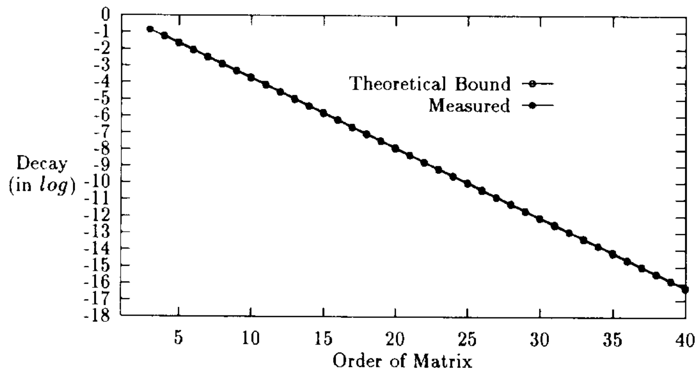

Fig. 2. Measured and predicted decay rate.

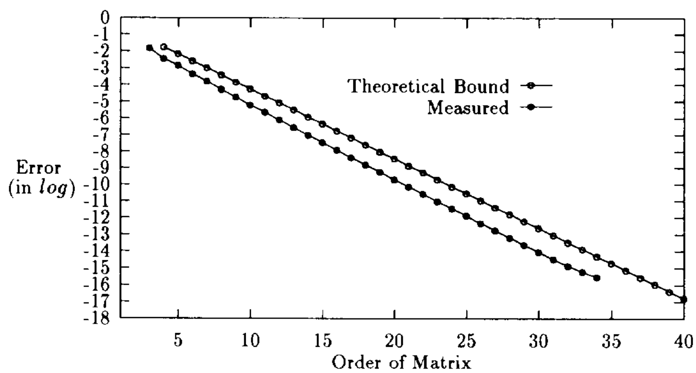

Fig. 3. Measured and predicted accuracy of the PDD algorithm.

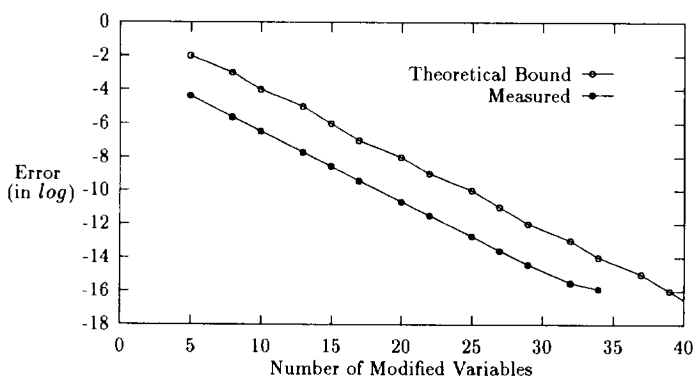

Fig. 4. Measured and predicted accuracy of the reduced PDD.

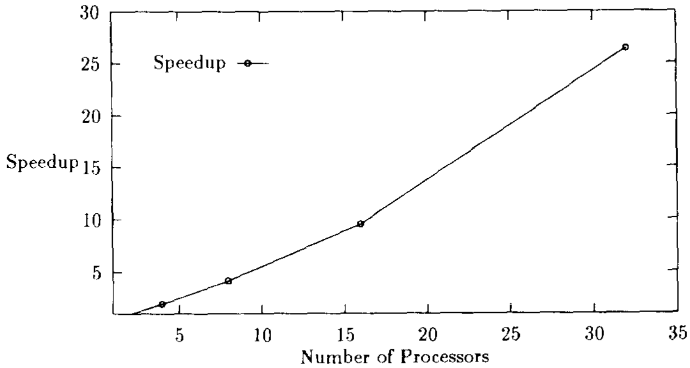

Fig. 5. Measured speedup over the Thomas algorithm. Single system of order 6400.

Figs. 5 and 6 give the speedup of the PDD algorithm over the Thomas algorithm for the corresponding periodic system. Here, speedup is defined as the single processor execution time of the conventional sequential algorithm over the parallel execution time of the parallel algorithm. For single system (a system with single right hand side), the order of matrix A is limited by the machine memory to $n = 6400$. For multiple right-sides, the system is limited to $n = 128$ and $n1 = 4096$. From Fig. 5 we can see that the speedup of solving a single system increases linearly with the number of processors. By the relation between speedup and scalability [21], it confirms the ideal scalability of the PDD algorithm. Fig. 6 shows that the linear increasing property does not hold for multiple right-side systems. The lower speedup is due to the reduction of the matrix size and the increase of the number of right hand sides. As seen in Table 1, the communication cost increases linearly with the number of right hand sides. Since the Intel/iPSC860 has a very high (communication speed)/(computation speed) ratio, we can expect

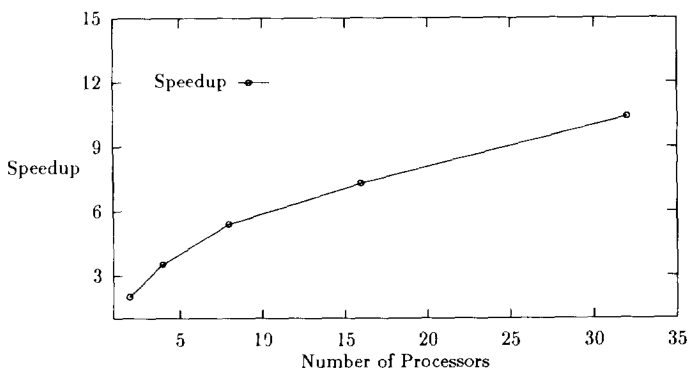

Fig. 6. Measured speedup over the Thomas algorithm. 4096 systems of order 128, factorization time not included.

a better speedup on an Intel Paragon or even on an Intel/iPSC2 [17] multicomputer.

6. Conclusion

A detailed study has been given for the Parallel Diagonal Dominant (PDD) tridiagonal algorithm. The PDD algorithm presented is slightly different from that originally proposed [20] and is extended to periodic systems. Based on our study, the reduced PDD algorithm was also introduced. While maintain the communication cost of the PDD algorithm, the reduced PDD algorithm has reduced the operation count considerably. An accuracy analysis was provided for a class of tridiagonal systems, the symmetric and skew-symmetric Toeplitz tridiagonal systems. Implementation results were provided for both our accuracy analysis and for the algorithms. These results showed that the accuracy analysis provides a very tight bound and that the algorithms are quite efficient for both single and multiple right-side systems. Thus the algorithms are good candidate for large scale computing, where the number of processors and the problem size are large. They are a good choice for the emerging massively parallel machines. While the discussion here was based on distributed-memory machines, the result can be easily applied to shared-memory machines as well.

The PDD algorithm and the reduced PDD algorithm proposed in this paper can be extended to band and block tridiagonal systems. Unfortunately our accuracy analysis, which gives a good, simple, relative error bound, is for symmetric and skew-symmetric Toeplitz tridiagonal systems only. It is unlikely that the analysis can be extended to the general case by similar techniques.

Acknowledgements

The author is indebted to several individuals at ICASE for their help with this research. In particular, Gordon Erlebacher brought the issue of designing fast symmetric Toeplitz tridiagonal solvers for compressible and uncompressible flow. John R. Van Rosendale offered valuable suggestions and comments which improved the technical quality and presentation of this paper. The author is also grateful to the editor and referees for their helpful comments on the revision of this paper.

References

[1] K.-L. Chung and L.-J. Shen, Vectorized algorithm for b-spline curve fitting on Cary X-MP EA/16se in: *Proc. Supercomputing '92* (1992) 166-169.

[2] S. Demko, W. Moss and P. Smith, Decay rates for inverses of band matrices, *Mathemat. Comput.*, 43 (168) (1984) 491-499.

[3] I. Duff, A. Erisman and J. Reid, *Direct Methods for Sparse Matrices* (Clarendon Press, Oxford, 1986).

[4] T. Dunigan, Performance of the intel ipsc/860 and ncube 6400 hypercubes, *Parallel Comput.* (17) (Dec. 1991) 1285–1302.

[5] O. Egecioglu, D. Koc and A. Laub, A recursive doubling algorithm for solution of tridiagonal systems on hypercube multiprocessors *J. Comp. and Appl. Math.* 27 (1989).

[6] J. Gustafson, G. Montry and R. Benner, Development of parallel methods for a 1024-processor hypercube, *SIAM J. Sci. and Stat. Computing* 9 (4) (July 1988) 609–638.

[7] C. Hirsch, *Numerical Computation of Internal and External Flows*. (John Wiley, 1988).

[8] C. Ho and S. Johnsson, Optimizing tridiagonal solvers for alternating direction methods on boolean cube multiprocessors, *SIAM J. Sci. and Stat. Computing* 11 (3) (1990) 563–592.

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

[10] D. Lawrie and A. Sameh, The computation and communication complexity of a parallel banded system solver, *ACM Trans. Math. Soft.* 10 (2) (June 1984) 185–195.

[11] T. Li, H. Zhang and X.-H Sun, Parallel homotopy algorithm for symmetric tridiagonal eigenvalue problem, *SIAM J. Sci. and Stat. Computing* 12 (May 1991).

[12] M.A. Malcolm and J. Palmer, A fast method for solving a class of tridiagonal linear systems, *Commun. ACM* 17 (1) (1974) 14–17.

[13] J. Ortega and R. Voigt, Solution of partial differential equations on vector and parallel computers, *SIAM Rev.* (June 1985) 149–240.

[14] J. Sherman and W. Morrison, Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix, *Ann. Math. Stat.* 20 621 (1949).

[15] H. Stone, An efficient parallel algorithm for the solution of a tridiagonal linear system of equations *J. ACM* 20 (1) (Jan. 1973) 27–38.

[16] J.C. Strikwerda, *Finite Difference Schemes and Partial Differential Equations* (Wadsworth & Brooks/Cole, Mathematics Series, 1989).

[17] X.-H. Sun and J. Gustafson, Toward a better parallel performance metric, *Parallel Comput.* 17 (Dec. 1991) 1093–1109.

[18] X.-H. Sun and R. Joslin, A massively parallel algorithm for compact finite difference schemes, in: *Proc. Int. Conf. on Parallel Processing* (1994) III282–III289.

[19] X.-H. Sun and D. Rover, Scalability of parallel algorithm-machine combinations, *IEEE Trans. Parallel Distributed Systems* (June 1994) 599–613.

[20] X.-H. Sun, H. Zhang and L. Ni, Efficient tridiagonal solvers on multicomputers, *IEEE Trans. Comput.* 41 (3) (1992) 286–296.

[21] X.-H. Sun and J. Zhu, Shared virtual memory and generalized speedup, in: *Proc. Eighth Int. Parallel Processing Symp.* (April 1994) 637–643.

[22] H. Wang, A parallel method for tridiagonal equations, *ACM Trans. Math. Software* 7 (June 1981) 170–183.

[23] M. Woodbury, Inverting modified matrices, Memorandum 42, Statistics Research Group, Princeton University, 1950.

[24] H. Zhang, On the accuracy of the parallel diagonal dominat algorithm, *Parallel Comput.* 17 (1991) 265–272.