Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 752651, 8 pages
http://dx.doi.org/10.1155/2014/752651

Research Article

Parallel Algorithm with Parameters Based on Alternating Direction for Solving Banded Linear Systems

Xinrong Ma,[1,2] Sanyang Liu,[1] Manyu Xiao,[3] and Gongnan Xie[4]

[1] *Department of Applied Mathematics, Xidian University, Xi'an 710071, China*
[2] *Department of Applied Mathematics, Xianyang Normal University, Xianyang 712000, China*
[3] *Department of Applied Mathematics, Northwestern Polytechnical University, Xi'an 710072, China*
[4] *School of Mechanical Engineering, Northwestern Polytechnical University, Xi'an 710072, China*

Correspondence should be addressed to Manyu Xiao; manyuxiao@gmail.com

Received 3 December 2013; Accepted 30 January 2014; Published 7 April 2014

Academic Editor: Massimo Scalia

An efficient parallel iterative method with parameters on distributed-memory multicomputer is investigated for solving the banded linear equations in this work. The parallel algorithm at each iterative step is executed using alternating direction by splitting the coefficient matrix and using parameters properly. Only it twice requires the communications of the algorithm between the adjacent processors, so this method has high parallel efficiency. Some convergence theorems for different coefficient matrices are given, such as a Hermite positive definite matrix or an M-matrix. Numerical experiments implemented on HP rx2600 cluster verify that our algorithm has the advantages over the multisplitting one of high efficiency and low memory space, which has a considerable advantage in CPU-times costs over the BSOR one. The efficiency for Example 1 is better than BSOR one significantly. As to Example 2, the acceleration rates and efficiency of our algorithm are better than the PEk inner iterative one.

1. Introduction

In recent years, the high-performance parallel computing technology has been rapidly developed. The large sparse banded linear systems are frequently encountered when finite difference or finite element methods are used to discretize partial differential equations in many practice scientific and engineering computing problems, especially in computational fluid dynamics (CFD). While many problems can be efficiently resolved on sequential computers but are difficult to solve on parallel computers, the communications take a significant part of the total execution time. So we need more efforts to investigate more efficient parallel algorithm to improve the experimental results.

The parallel algorithms on the large sparse linear systems have been widely investigated in [1–8]. Specifically, the multisplitting algorithm in [1] is a popular method at present. In [3], the authors provide a method for solving block-tridiagonal linear systems in which local lower and upper triangular incomplete factors are combined into an effective approximation for global incomplete lower and upper triangular factors of coefficient matrix based on two-dimensional domain decomposition with small overlapping. The algorithm is applicable to any preconditioner of incomplete type. Duan et al. presented a parallel strategy based on the Galerkin principle for solving block-tridiagonal linear systems in [4]. In [5], a parallel direct algorithm based on Divide-and-Conquer principle and the decomposition of the coefficient matrix is investigated for solving the block-tridiagonal linear systems on distributed-memory multicomputers. The communication of the algorithm is only twice between the adjacent processors. In [7], a direct method for solving circular-tridiagonal block linear systems is presented. Some parallel algorithms for solving the linear systems can be found in [9–14]. The algorithm in this paper is discussed on the basis of the advantages of the one in [2].

The goal of this paper is to develop an efficient, stable parallel iterative method on distributed-memory multicomputer and to give some theoretical analysis. We appropriately choose the splitting matrices $\mathbf{W}$ and $\mathbf{V}$ to establish

the iterative scheme. Two examples have been done on the HP rx2600 cluster; the experimental results indicate that the parallel algorithm has advantages over the multisplitting one of high parallel speedup and efficiency.

The content of this paper is as follows. In Section 2, the parallel iterative algorithm is described. In Section 3, the parallel iterative process is discussed. The analysis of convergence is done in Section 4. The numerical results are shown in Section 5. In Section 6, the conclusion is presented.

2. Parallel Algorithm

Let a banded linear equation $\mathbf{AX} = \mathbf{b}$ be represented as

$$\begin{pmatrix} \mathbf{A}_1 & \mathbf{B}_1 & & & & \\ \mathbf{C}_2 & \mathbf{A}_2 & \mathbf{B}_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \mathbf{C}_{n-1} & \mathbf{A}_{n-1} & \mathbf{B}_{n-1} & \\ & & & \mathbf{C}_n & \mathbf{A}_n & \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{n-1} \\ \mathbf{x}_n \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_{n-1} \\ \mathbf{b}_n \end{pmatrix}, \tag{1}$$

where $\mathbf{A}_i$ is a $d_i \times d_i$ matrix, $\mathbf{B}_i$ and $\mathbf{C}_i$ are $d_i \times d_{i+1}$ and $d_i \times d_{i-1}$ matrices, respectively, and $\mathbf{x}_i$ and $\mathbf{b}_i$ are d_i-dimensional real column vectors. In general, assuming that there are p processors available and $n = 2hp$ ($h \geq 1$, $h \in Z^+$), we denote the ith processor by P_i (for $i = 1, 2, \ldots, p$) and split the coefficient matrix $\mathbf{A}$ into $\mathbf{A} = \mathbf{W} + \mathbf{V}$.

Then, we use the alternating direction iterative scheme in [2] and obtain the new iterative scheme

$$(\mathbf{I} + \tau\mathbf{W})(\mathbf{I} + \tau\mathbf{V})\left(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right) = -\alpha\tau\left(\mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}\right); \tag{2}$$

here $\mathbf{I} + \tau\mathbf{W}$ and $\mathbf{I} + \tau\mathbf{V}$ are nonsingular matrices and $\alpha = 2$. And hence (2) is changed into

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)} - (\mathbf{I} + \tau\mathbf{V})^{-1}(\mathbf{I} + \tau\mathbf{W})^{-1} \\ &\quad \times \left[2\tau\left(\mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}\right)\right] \\ &= \left(\mathbf{I} - 2\tau(\mathbf{I} + \tau\mathbf{V})^{-1}(\mathbf{I} + \tau\mathbf{W})^{-1}\mathbf{A}\right)\mathbf{x}^{(k)} \\ &\quad + 2\tau(\mathbf{I} + \tau\mathbf{V})^{-1}(\mathbf{I} + \tau\mathbf{W})^{-1}\mathbf{b} \\ &= \mathbf{B}_\omega\mathbf{x}^{(k)} + \mathbf{g}; \end{aligned} \tag{3}$$

here, $\mathbf{B}_\omega = \mathbf{I} - 2\tau(\mathbf{I} + \tau\mathbf{V})^{-1}(\mathbf{I} + \tau\mathbf{W})^{-1}\mathbf{A}$ is the so-called iterative matrix and $\mathbf{g} = 2\tau(\mathbf{I} + \tau\mathbf{V})^{-1}(\mathbf{I} + \tau\mathbf{W})^{-1}\mathbf{b}$.

Obviously, the matrices $\mathbf{I} + \tau\mathbf{W}$ and $\mathbf{I} + \tau\mathbf{V}$ should be nonsingular and the definition of $\mathbf{W}$ and $\mathbf{V}$ is the most important key of solving the linear systems by (3) in this paper. If $\mathbf{W}$ and $\mathbf{V}$ are suitable, the algorithm would have good parallelism and low CPU-times costs. So we choose $\mathbf{W}$ and $\mathbf{V}$ as follows

$$\mathbf{W} = \begin{pmatrix} \alpha\mathbf{A}_1 & \mathbf{B}_1 & & & & & \\ & \beta\mathbf{A}_2 & & & & & \\ & \mathbf{C}_3 & \alpha\mathbf{A}_3 & \mathbf{B}_3 & & & \\ & & & \beta\mathbf{A}_4 & & & \\ & & & & \ddots & & \\ & & & & \mathbf{C}_{2hp-1} & \alpha\mathbf{A}_{2hp-1} & \mathbf{B}_{2hp-1} \\ & & & & & & \beta\mathbf{A}_{2hp} \end{pmatrix},$$

$$\mathbf{V} = \begin{pmatrix} (1-\alpha)\mathbf{A}_1 & & & & & & & \\ \mathbf{C}_2 & (1-\beta)\mathbf{A}_2 & \mathbf{B}_2 & & & & & \\ & & (1-\alpha)\mathbf{A}_3 & & & & & \\ & & \mathbf{C}_4 & (1-\beta)\mathbf{A}_4 & \mathbf{B}_4 & & & \\ & & & & \ddots & & & \\ & & & & & (1-\alpha)\mathbf{A}_{2hp-1} & & \\ & & & & & \mathbf{C}_{2hp} & (1-\beta)\mathbf{A}_{2hp} \end{pmatrix}. \tag{4}$$

From (3), let $\mathbf{y} = (\mathbf{I} + \tau\mathbf{W})^{-1}(\mathbf{A}\mathbf{x}^{(k)} - \mathbf{b})$; we obtain

$$(\mathbf{I} + \tau\mathbf{W})\,\mathbf{y} = \mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}; \tag{5}$$

then the detailed calculation procedure is as follows:

$$\begin{aligned} &\left(\mathbf{I} + \tau\beta\mathbf{A}_{(i-1)2h+j}\right)\mathbf{y}_{(i-1)2h+j} \\ &\quad = \mathbf{C}_{(i-1)2h+j}\mathbf{x}^{(k)}_{(i-1)2h+j-1} + \mathbf{A}_{(i-1)2h+j}\mathbf{x}^{(k)}_{(i-1)2h+j} \\ &\qquad + \mathbf{B}_{(i-1)2h+j}\mathbf{x}^{(k)}_{(i-1)2h+j+1} - \mathbf{b}_{(i-1)2h+j} \quad (i = 2, 4, \ldots, 2h), \\ &\left(\mathbf{I} + \tau\alpha\mathbf{A}_{(i-1)2h+j}\right)\mathbf{y}_{(i-1)2h+j} \\ &\quad = \mathbf{C}_{(i-1)2h+j}\left(\mathbf{x}^{(k)}_{(i-1)2h+j-1} - \tau\mathbf{y}_{(i-1)2h+j-1}\right) \\ &\qquad + \mathbf{A}_{(i-1)2h+j}\mathbf{x}^{(k)}_{(i-1)2h+j} \\ &\qquad + \mathbf{B}_{(i-1)2h+j}\left(\mathbf{x}^{(k)}_{(i-1)2h+j+1} - \tau\mathbf{y}_{(i-1)2h+j+1}\right) \\ &\qquad - \mathbf{b}_{(i-1)2h+j} \quad (i = 1, 3, \ldots, 2h-1); \end{aligned} \tag{6}$$

here, $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_i, \ldots, \mathbf{y}_p)^T$ and $\mathbf{y}_i$ is a $2h$-dimentional row vector.

Let $\mathbf{z} = (\mathbf{I} + \tau\mathbf{V})^{-1}\mathbf{y}$; then we have $(\mathbf{I} + \tau\mathbf{V})\mathbf{z} = \mathbf{y}$, and

$$\begin{aligned}
&\left[\mathbf{I} + \tau(1-\alpha)\mathbf{A}_{(i-1)2h+j}\right]\mathbf{z}_{(i-1)2h+j} = \mathbf{y}_{(i-1)2h+j},\\
&\qquad\qquad j = (1, 3, \ldots, 2h-1),\\
&\left[\mathbf{I} + \tau(1-\beta)\mathbf{A}_{(i-1)2h+j}\right]\mathbf{z}_{(i-1)2h+j}\\
&\quad = \mathbf{y}_{(i-1)2h+j} - \tau\mathbf{C}_{(i-1)2h+j}\mathbf{z}_{(i-1)2h+j-1}\\
&\qquad - \tau\mathbf{B}_{(i-1)2h+j}\mathbf{z}_{(i-1)2h+j+1}, \quad (j = 2, 4, \ldots, 2h),
\end{aligned} \tag{7}$$

where $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_i, \ldots, \mathbf{z}_p)^T$ and $\mathbf{z}_i$ is a $2h$-dimentional row vector. Then according to the aforementioned formulas, we get $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - 2\tau\mathbf{z}$.

3. Process of Parallel Iterative Algorithm

Here, we show the storage method and computational procedure of the parallel algorithm as follows.

3.1. Storage Method. The coefficient matrix is divided into $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_{k_l+1}, \ldots, \mathbf{A}_{k_l+k_u+1}$ from left to right as banded order. Let vectors $\mathbf{A}_1 = (0, \ldots, 0, a_{k_l+1,1}, a_{k_l+2,2}, \ldots, a_{n,n-k_l})^{\mathbf{T}}, \ldots, \mathbf{A}_{k_l+1} = (a_{11}, a_{22}, \ldots, a_{nn})^{\mathbf{T}}, \ldots, \mathbf{A}_{k_l+k_u+1} = (a_{1,k_u+1}, a_{2,k_u+2}, \ldots, a_{n-k_u,n}, 0, \ldots, 0)^{\mathbf{T}}$.

The corresponding relationship is as follows:

$$\mathbf{A} = \begin{pmatrix}
a_{11} & \cdots & a_{1,k_u+1} & 0 & \cdots & \cdots & \cdots & \cdots & 0\\
a_{21} & a_{22} & \cdots & a_{2,k_u+2} & 0 & \cdots & \cdots & \cdots & 0\\
\vdots & \vdots & \ddots & & \ddots & \ddots & & & \vdots\\
a_{k_l+1,1} & a_{k_l+1,2} & \cdots & a_{k_l+1,k_l+1} & \cdots & a_{k_l+1,k_l+k_u+2} & 0 & \cdots & 0\\
\vdots & \ddots & & & \ddots & & \ddots & & \\
\vdots & & \ddots & & \ddots & \ddots & & \ddots & \vdots\\
0 & \cdots & 0 & a_{n-k_u,n-k_u-k_l} & \cdots & \cdots & a_{n-k_u,n-k_u} & \cdots & a_{n-k_u,n}\\
\vdots & & \vdots & & \ddots & & & \ddots & \vdots\\
0 & \cdots & 0 & \cdots & \cdots & a_{n,n-k_l} & \cdots & a_{n,n-1} & a_{nn}
\end{pmatrix} \tag{8}$$

$$\Longrightarrow \begin{array}{c}
\begin{matrix} \mathbf{A}_1 & \cdots & \mathbf{A}_{k_l+1} & \cdots & \mathbf{A}_{k_l+k_u+1} \end{matrix}\\
\begin{matrix} \Downarrow & \cdots & \Downarrow & \cdots & \Downarrow \end{matrix}\\
\begin{pmatrix}
0 & \cdots & a_{11} & \cdots & a_{1,k_u+1}\\
\vdots & & \vdots & & \vdots\\
a_{k_l+1,1} & \cdots & a_{k_l+1,k_l+1} & \cdots & a_{k_l+1,k_l+k_u+2}\\
\vdots & \cdots & \vdots & \cdots & \vdots\\
a_{n,n-k_l} & & \cdots & & 0
\end{pmatrix}
\end{array}
\begin{matrix} \\ \\ \\ \mathbf{0} \\ \\ \\ \\ \end{matrix}
\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \end{matrix}
\begin{matrix} \\ \\ \\ \mathbf{0} \\ \\ \\ \\ \end{matrix}
\Bigg).$$

Then, assign m ($m = n/p$) rows to each processor. The processor stores the corresponding vectors $\mathbf{b}_i$, $\mathbf{x}_i$ with $i = 1, 2, \ldots, p$. Here k_u and k_l are upper-band width and lower-band width, respectively. In such a case, this saves much of the memory space although programming is difficult. Note that if n is not divisible by p, some processors store $[n/p] + 1$ rows-block of $\mathbf{A}$, sequentially, and others store $[n/p]$ rows-block; meanwhile, each processor stores the corresponding vectors of $\mathbf{x}^{(0)}$and $\mathbf{b}$. Thereby, it makes load of each processor approach balance and shorten wait time.

3.2. Cycle Process. (1) P_i performs a parallel communication to obtain $\mathbf{x}^{(k)}_{(i-1)2m}$, $\mathbf{x}^{(k)}_{(i)2m+1}$ and then computes

$$\mathbf{y}_i = -2\tau\left(\mathbf{I} + \tau\mathbf{W}_i\right)^{-1}\left(\mathbf{A}_i\mathbf{x}_i^{(k)} - \mathbf{b}_i\right) \tag{9}$$

and implements LU discretization one-step, where $\mathbf{W}_i$, $\mathbf{A}_i$, $\mathbf{b}_i$, and $\mathbf{x}_i$ are the ith (for $i = 1, 2, \ldots, p$) block of $\mathbf{W}$, $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{x}$, respectively.

(2) P_i performs one parallel communication to obtain $\mathbf{y}_{(i-1)2m}$ and then computes

$$\mathbf{x}_i^{(k+1)} = \mathbf{x}_i^{(k)} + \left(\mathbf{I} + \tau\mathbf{V}_i\right)^{-1}\mathbf{y}_i \tag{10}$$

and implements LU discretization one-step; here $\mathbf{V}_i$ is the ith (for $i = 1, 2, \ldots, p$) block of $\mathbf{V}$.

(3) On the P_i processor, judge whether the inequality $\|\mathbf{x}_i^{(k+1)} - \mathbf{x}_i^{(k)}\| < \varepsilon$ (ε is error bound, $i = 1, 2, \ldots, p$) holds. Stop if these inequalities hold on every processor, or return to (1) and continue cycling until all inequalities are satisfied.

4. Analysis of Convergence

To perform the theoretical analysis on convergence of the parallel algorithm, we introduce the definition and several lemmata.

Symbol and Definition

(i) $R^{n\times n}$ represents the space of $n \times n$ real matrices.

(ii) $\mathbf{I}_r$ represents the unit matrix of order r.

(iii) $\mathbf{W}^H$, $\mathbf{V}^H$ represent the conjugate transpose matrix of $\mathbf{W}$, $\mathbf{V}$, respectively.

(iv) $\mathbf{W}^{-1}$ represents the inverse matrix of $\mathbf{W}$.

Definition 1 (see [15]). Suppose $\mathbf{A} \in R^{n\times n}$ and $\mathbf{A} = \mathbf{Q} - \mathbf{S}$, where $\mathbf{Q}^{-1} \geq \mathbf{0}$ and $\mathbf{S} \geq \mathbf{0}$; then $\mathbf{A} = \mathbf{Q} - \mathbf{S}$ is called normal splitting of matrix $\mathbf{A}$.

Definition 2 (see [15]). Suppose $\mathbf{A} \in R^{n\times n}$ and $\mathbf{A} = \mathbf{Q} - \mathbf{S}$, where $\mathbf{Q}^{-1}\mathbf{S} \geq \mathbf{0}$; then $\mathbf{A} = \mathbf{Q} - \mathbf{S}$ is called weak normal splitting of matrix $\mathbf{A}$.

Definition 3 (see [15]). Suppose $\mathbf{A} \in R^{n\times n}$ and $\mathbf{A} = \mathbf{Q} - \mathbf{S}$, where $\mathbf{Q}^H + \mathbf{S}$ is a Hermite positive definite matrix; then $\mathbf{A} = \mathbf{Q} - \mathbf{S}$ is called P-normal splitting of matrix $\mathbf{A}$.

Definition 4 (see [15]). Let $\mathbf{A} = (a_{ij}) \in \mathbf{R}^{n\times n}$, if $a_{ij} \leq 0$ $(i \neq j)$ and $\mathbf{A}^{-1} \geq \mathbf{0}$; then the matrix $\mathbf{A}$ is an M-matrix.

Here, we give some theoretical analysis for convergence of the parallel iterative algorithm.

Lemma 5 (see [9]). *Let $\mathbf{A} \in R^{n\times n}$, if the splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$ is a weak normal splitting or normal splitting of coefficient matrix $\mathbf{A}$; then $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$ if and only if $\mathbf{A}^{-1} \geq \mathbf{0}$.*

Lemma 6 (see [10]). *Let $\mathbf{A}$ be an M-matrix. If any element of $\mathbf{A}$ increases while outside elements of the main diagonal keep nonpositive, then the transformation matrix $\mathbf{B}$ is also an M-matrix and $\mathbf{B}^{-1} \leq \mathbf{A}^{-1}$.*

Lemma 7 (see [15]). *Let $\mathbf{A} \in C^{n\times n}$ be a nonsingular Hermite matrix. If $\mathbf{A} = \mathbf{M} - \mathbf{N}$ is a P-normal splitting of the matrix $\mathbf{A}$, then $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$ if and only if $\mathbf{A}$ is a positive definite matrix.*

Theorem 8. *Let $\mathbf{A} \in R^{n\times n}$ be a Hermite positive definite matrix. If $\tau > 0$, $\beta = 1/2$, and $0 \leq \alpha \leq 1$, then the iterative scheme (3) is convergent for all vector $\mathbf{x}^{(0)}$.*

Proof. Since $(\mathbf{I} + \tau\mathbf{W})(\mathbf{I} + \tau\mathbf{V})(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -2\tau(\mathbf{A}\mathbf{x}^{(k)} - \mathbf{b})$ and

$$\begin{aligned}\mathbf{B}_\omega &= \left(\mathbf{I} - 2\tau(\mathbf{I} + \tau\mathbf{V})^{-1}(\mathbf{I} + \tau\mathbf{W})^{-1}\mathbf{A}\right)\\ &= (\mathbf{I} + \tau\mathbf{V})^{-1}(\mathbf{I} + \tau\mathbf{W})^{-1}(\mathbf{I} - \tau\mathbf{W})(\mathbf{I} - \tau\mathbf{V}),\end{aligned} \tag{11}$$

we have $\mathbf{A} = \mathbf{M} - \mathbf{N}$; here $\mathbf{M} = (\mathbf{I} + \tau\mathbf{V})(\mathbf{I} + \tau\mathbf{W})$, $\mathbf{N} = (\mathbf{I} - \tau\mathbf{W})(\mathbf{I} - \tau\mathbf{V})$,

$$\begin{aligned}\mathbf{M}^H + \mathbf{N} &= (\mathbf{I} + \tau\mathbf{V})^H(\mathbf{I} + \tau\mathbf{W})^H + (\mathbf{I} - \tau\mathbf{W})(\mathbf{I} - \tau\mathbf{V})\\ &= \mathbf{I} + \tau\mathbf{A} + \tau^2\mathbf{V}^H\mathbf{W}^H + \mathbf{I} - \tau\mathbf{A} + \tau^2\mathbf{W}\mathbf{V}\\ &= 2\mathbf{I} + \tau^2\left(\mathbf{W}\mathbf{V} + \mathbf{V}^{\mathbf{H}}\mathbf{W}^{\mathbf{H}}\right).\end{aligned} \tag{12}$$

Since

$$\mathbf{WV} = \begin{pmatrix} \Lambda_1 & (1-\beta)\mathbf{D}_1 & \mathbf{E}_1 & & & & & & \\ \beta\mathbf{D}_1^{\mathbf{H}} & \Lambda_2 & \beta\mathbf{F}_2 & & & & & & \\ \mathbf{E}_1^{\mathbf{H}} & (1-\beta)\mathbf{F}_2^{\mathbf{H}} & \Lambda_3 & (1-\beta)\mathbf{D}_3 & \mathbf{E}_3 & & & & \\ & & \beta\mathbf{D}_3^{\mathbf{H}} & \Lambda_4 & \beta\mathbf{F}_4 & & & & \\ & & \mathbf{E}_3^{\mathbf{H}} & (1-\beta)\mathbf{F}_4^{\mathbf{H}} & \Lambda_5 & (1-\beta)\mathbf{D}_5 & \mathbf{E}_5 & & \\ & & & & \ddots & & & & \\ & & & & & \beta\mathbf{D}_{2hp-3}^{\mathbf{H}} & \Lambda_{2hp-2} & \beta\mathbf{F}_{2hp-2} & \\ & & & & & \mathbf{E}_{2hp-3}^{\mathbf{H}} & (1-\beta)\mathbf{F}_{2hp-2}^{\mathbf{H}} & \Lambda_{2hp-1} & (1-\beta)\mathbf{D}_{2hp-1} \\ & & & & & & & \beta\mathbf{D}_{2hp-1}^{\mathbf{H}} & \Lambda_{2hp} \end{pmatrix}, \tag{13}$$

here

$$\Lambda_1 = \alpha(1-\alpha)\mathbf{A}_1^2 + \mathbf{B}_1\mathbf{B}_1^{\mathbf{H}},$$

$$\Lambda_i = \begin{cases} \beta(1-\beta)\mathbf{A}_i^2 & (i = 2, 4, \ldots, 2n) \\ \alpha(1-\alpha)\mathbf{A}_i^2 + \mathbf{B}_{i-1}^{\mathbf{H}}\mathbf{B}_{i-1} + \mathbf{B}_i\mathbf{B}_i^{\mathbf{H}} & (i = 3, 5, \ldots, 2hp-1), \end{cases}$$

$$\mathbf{D}_i = \mathbf{B}_i\mathbf{A}_{i+1}, \quad \mathbf{E}_i = \mathbf{B}_i\mathbf{B}_{i+1} \quad (i = 1, 2, \ldots, 2hp),$$

$$\mathbf{F}_i = \mathbf{A}_i\mathbf{B}_i \quad (i = 2, 4, \ldots, 2hp), \tag{14}$$

and let

$$\mathbf{U} = \begin{pmatrix} \mathbf{0} & \mathbf{B}_1 & & & & & \\ \frac{1}{2}\mathbf{A}_2 & & & & & & \\ \mathbf{C}_3 & \mathbf{0} & \mathbf{B}_3 & & & & \\ & & \ddots & & & & \\ & & & \mathbf{C}_{2hp-1} & \mathbf{0} & \mathbf{B}_{2hp-1} & \\ & & & & & \frac{1}{2}\mathbf{A}_{2hp} \end{pmatrix}; \tag{15}$$

then we have

$$\mathbf{WV} + \mathbf{V^H W^H} - 2\mathbf{UU^H} = \begin{pmatrix} 2\Lambda_1 - \mathbf{Q}_1 & & & & \\ & 2\Lambda_2 - \mathbf{Q}_2 & & & \\ & & \ddots & & \\ & & & 2\Lambda_{2n-1} - \mathbf{Q}_{2n-1} & \\ & & & & 2\Lambda_{2n} - \mathbf{Q}_{2n} \end{pmatrix}; \tag{16}$$

here

$$\mathbf{Q}_1 = 2\mathbf{B}_1\mathbf{B}_1^{\mathbf{H}},$$

$$\mathbf{Q}_i = \begin{cases} \frac{1}{2}\mathbf{A}_i^2 & (i = 2, 4, \ldots, 2hp) \\ 2\mathbf{B}_{i-1}^{\mathbf{H}}\mathbf{B}_{i-1} + 2\mathbf{B}_i\mathbf{B}_i^{\mathbf{H}} & (i = 3, 5, \ldots, 2hp-1), \end{cases}$$

$$2\Lambda_i - \mathbf{Q}_i = \begin{cases} 2\alpha(1-\alpha)\mathbf{A}_i^2 & (i = 1, 3, \ldots, 2hp-1) \\ \left[2\beta(1-\beta) - \frac{1}{2}\right]\mathbf{A}_i^2 & (i = 2, 4, \ldots, 2hp). \end{cases} \tag{17}$$

Obviously, $\mathbf{WV} + \mathbf{V^H W^H} - 2\mathbf{UU^H}$ is a semipositive definite matrix or a positive definite matrix. Hence the matrix

$$\begin{aligned} \mathbf{M}^H + \mathbf{N} &= 2\mathbf{I} + \tau^2\left(\mathbf{WV} + \mathbf{V}^H\mathbf{W}^H\right) \\ &= 2\mathbf{I} + \tau^2\left(\mathbf{WV} + \mathbf{V}^H\mathbf{W}^H - 2\mathbf{UU}^H\right) + 2\tau^2\mathbf{UU}^H \end{aligned} \tag{18}$$

is a Hermite positive definite matrix.

Therefore, $\mathbf{A} = \mathbf{M} - \mathbf{N}$ is a P-normal splitting of the matrix $\mathbf{A}$, and then $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$ by Lemma 7; we know that our algorithm iterative scheme is convergent. □

By the theorem, we know that the parallel algorithm is convergent if $\mathbf{A}$ is a Hermite positive definite matrix.

Theorem 9. *Let $\mathbf{A} \in R^{n\times n}$ be an M-matrix. If $0 < \tau \le \min\{1/\alpha, 1/\beta, 1/(1-\alpha), 1/(1-\beta)\}\min(1/a_{ii})$ for $i = 1, 2, \ldots, 2hp$, here $0 < \alpha, \beta < 1$ and a_{ii} is the diagonal element of $\mathbf{A}$; then the iterative scheme (3) is convergent for all vector $\mathbf{x}^{(0)}$.*

Proof. Since $\mathbf{M} = (\mathbf{I} + \tau\mathbf{V})(\mathbf{I} + \tau\mathbf{W})$, $\mathbf{N} = (\mathbf{I} - \tau\mathbf{W})(\mathbf{I} - \tau\mathbf{V})$, and

$$\mathbf{I} + \tau\mathbf{V} = \begin{pmatrix} \mathbf{I} + (1-\alpha)\tau\mathbf{A}_1 & & & & & & & \\ \tau\mathbf{C}_2 & \mathbf{I} + \tau(1-\beta)\mathbf{A}_2 & \tau\mathbf{B}_2 & & & & & \\ & & \mathbf{I} + (1-\alpha)\tau\mathbf{A}_3 & & & & & \\ & & \tau\mathbf{C}_4 & \mathbf{I} + \tau(1-\beta)\mathbf{A}_4 & \tau\mathbf{B}_4 & & & \\ & & & & \ddots & & & \\ & & & & & \mathbf{I} + (1-\alpha)\tau\mathbf{A}_{2hp-1} & & \\ & & & & & \tau\mathbf{C}_{2hp} & \mathbf{I} + \tau(1-\beta)\mathbf{A}_{2hp} \end{pmatrix}, \tag{19}$$

we have

$$(\mathbf{I} + \tau\mathbf{V})^{-1} = \begin{pmatrix} (\mathbf{I} + (1-\alpha)\tau\mathbf{A}_1)^{-1} & & & & & & & \\ \mathbf{Q}_2 & (\mathbf{I} + (1-\beta)\tau\mathbf{A}_2)^{-1} & \mathbf{F}_2 & & & & & \\ & & (\mathbf{I} + (1-\alpha)\tau\mathbf{A}_3)^{-1} & & & & & \\ & & \mathbf{Q}_4 & (\mathbf{I} + (1-\beta)\tau\mathbf{A}_4)^{-1} & \mathbf{F}_4 & & & \\ & & & & \ddots & & & \\ & & & & & \left(\mathbf{I} + (1-\alpha)\tau\mathbf{A}_{2hp-1}\right)^{-1} & & \\ & & & & & \mathbf{Q}_{2hp} & \left(\mathbf{I} + (1-\beta)\tau\mathbf{A}_{2hp}\right)^{-1} \end{pmatrix}. \tag{20}$$

Here

$$\mathbf{Q}_i = -\tau\left(\mathbf{I} + \left(1-\beta\right)\tau\mathbf{A}_i\right)^{-1}\mathbf{C}_i\left(\mathbf{I} + \left(1-\alpha\right)\tau\mathbf{A}_{i-1}\right)^{-1}, \quad \left(i = 2, 4, \ldots, 2hp\right),$$
$$\mathbf{F}_i = -\tau\left(\mathbf{I} + \left(1-\beta\right)\tau\mathbf{A}_i\right)^{-1}\mathbf{B}_i\left(\mathbf{I} + \left(1-\alpha\right)\tau\mathbf{A}_{i+1}\right)^{-1}, \quad \left(i = 2, 4, \ldots, 2hp-2\right). \tag{21}$$

Hence, we know that $(\mathbf{I} + (1-\alpha)\tau\mathbf{A}_{i-1})$, $(\mathbf{I} + (1-\alpha)\tau\mathbf{A}_{i+1})$, and $(\mathbf{I} + (1-\beta)\tau\mathbf{A}_i)$, $(i = 1, 2, \ldots, 2hp)$, are all M-matrices by Lemma 6. Then $(\mathbf{I} + (1-\alpha)\tau\mathbf{A}_{i-1})^{-1} \geq \mathbf{0}$, $(\mathbf{I} + (1-\alpha)\tau\mathbf{A}_{i+1})^{-1} \geq \mathbf{0}$, $(\mathbf{I} + (1-\beta)\tau\mathbf{A}_i)^{-1} \geq \mathbf{0}$, $\mathbf{Q}_i \geq \mathbf{0}$, and $\mathbf{F}_i \geq \mathbf{0}$; we obtain $(\mathbf{I} + \tau\mathbf{V})^{-1} \geq \mathbf{0}$. Similarly, we can obtain $(\mathbf{I} + \tau\mathbf{W})^{-1} \geq \mathbf{0}$, and $\mathbf{M}^{-1} \geq \mathbf{0}$.

Since $0 < \tau \leq \min\{1/\alpha, 1/\beta, 1/(1-\alpha), 1/(1-\beta)\}\min(1/a_{ii})$ for $i = 1, 2, \ldots, 2hp$, we have $(\mathbf{I} - \tau\mathbf{W}) \geq \mathbf{0}$ and $(\mathbf{I} - \tau\mathbf{V}) \geq \mathbf{0}$. That is, $\mathbf{N} \geq \mathbf{0}$ is obtained and $\mathbf{A} = \mathbf{M} - \mathbf{N}$ is a normal splitting. Since $\mathbf{A}$ is an M-matrix, then $\mathbf{A}^{-1} \geq \mathbf{0}$; we know that $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$ by Lemma 5, and the iterative scheme (3) is convergent. □

By the theorem, we know that the parallel algorithm is convergent if $\mathbf{A}$ is an M-matrix and $0 < \tau \leq \min\{1/\alpha, 1/\beta, 1/(1-\alpha), 1/(1-\beta)\}\min(1/a_{ii})$ for $i = 1, 2, \ldots, 2hp$.

5. Numerical Examples

We performed two numerical experiments on the HP rx2600 cluster. The results are shown as follows.

Example 1. Consider a banded linear system $\mathbf{AX} = \mathbf{b}$; here

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{B}_1 & & & \\ \mathbf{C}_2 & \mathbf{A}_2 & \mathbf{B}_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{C}_{m-1} & \mathbf{A}_{m-1} & \mathbf{B}_{m-1} \\ & & & \mathbf{C}_m & \mathbf{A}_m \end{pmatrix},$$
$$\mathbf{A}_i = \begin{pmatrix} 15.1 & -3.5 & -6.9 \\ -2.7 & 20.1 & -4.8 \\ -15.7 & -5.3 & 25.1 \end{pmatrix}, \tag{22}$$
$$\mathbf{B}_i = \mathbf{C}_i = \begin{pmatrix} -3 & & \\ & -2 & \\ & & -4 \end{pmatrix}, \qquad \mathbf{b}_i = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

Let initialization value $\mathbf{x}_i^{(0)} = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^{\mathbf{T}}$ and $m = 80000$. We apply this algorithm with the optimal relaxation factor, the multisplitting method, and BSOR method to the systems on the HP rx2600 cluster. Here P is the number of processor, T is the run times (seconds), the S is speedup (T of one processor/T of all processors), L is iteration times, E is the efficiency ($E = S/P$), and the error $\varepsilon = 1 \times 10^{-10}$. See Tables 1, 2, and 3 and Figures 1 and 2.

Table 1: The results for model 1 (the algorithm in the paper ($\tau = 0.9, \alpha = \beta = 1/2$)).

P	1	2	4	8
T	2.9921	1.5179	0.8028	0.6492
S		1.9712	3.7271	4.6089
E		0.9856	0.9318	0.5761
L	39	39	39	39

Table 2: The results for model 1 (the multisplitting method).

P	1	2	4	8
T	7.9544	7.1352	3.3874	2.2426
S		1.1148	2.3482	3.5470
E		0.5745	0.5871	0.4434
L	160	224	224	224

Table 3: The results for model 1 (BSOR method ($\omega = 1.85$)).

P	1	2	4	8
T	17.0168	8.9928	4.6551	3.8031
S		1.8923	3.6555	4.4745
E		0.9461	0.9139	0.5593
L	495	532	532	532

Example 2. Consider an elliptic partial differential equation

$$C_x\frac{\partial^2 u}{\partial x^2} + C_y\frac{\partial^2 u}{\partial y^2} + \left(C_1\sin 2\pi x + C_2\right)\frac{\partial u}{\partial x} + \left(D_1\sin 2\pi x + D_2\right)\frac{\partial u}{\partial x} + Eu = 0, \quad 0 \leq x, \quad y \leq 1, \tag{23}$$

equipped with the boundary conditions $u|_{x=0} = u|_{x=1} = 10 + \cos\pi y$, $u|_{y=0} = u|_{y=1} = 10 + \cos\pi x$; here C_x, C_y, C_1, C_2, D_1, D_2, and E are all constants.

We denote $C_x = C_y = E = 1$, $C_1 = C_2 = D_1 = D_2 = 0$. Using the finite difference method, we obtain two block-tridiagonal linear systems on condition that the step sizes $h = 1/100$. Then, we apply this algorithm with the optimal relaxation factor, BSOR method, PEk method, and the multisplitting algorithm to the systems on the HP rx2600 cluster. The numerical results are shown in Tables 4, 5, 6, and 7 and Figures 3 and 4.

6. Results Analysis

From Table 1 to Table 7, we can get the following conclusion.

(i) It can be known that the results of the parallel algorithm verify the results of the theoretical analysis. The conditions in the theorems are only sufficient conditions.

(ii) By the numerical results, it can be known that the parallel one has good parallelism.

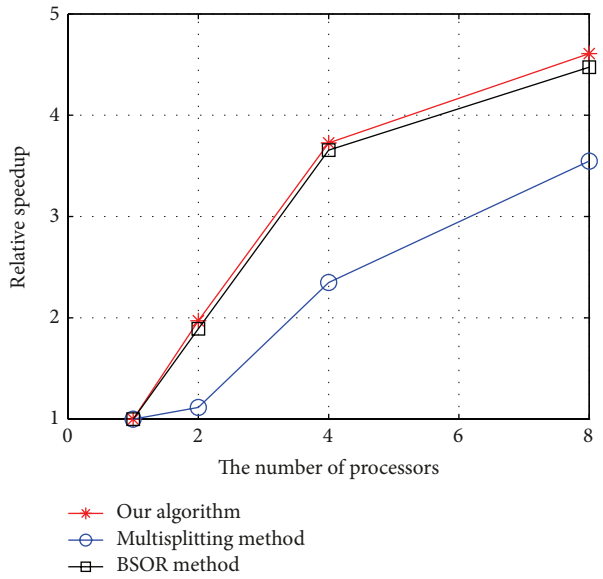

Figure 1: The parallel speedup for Example 1.

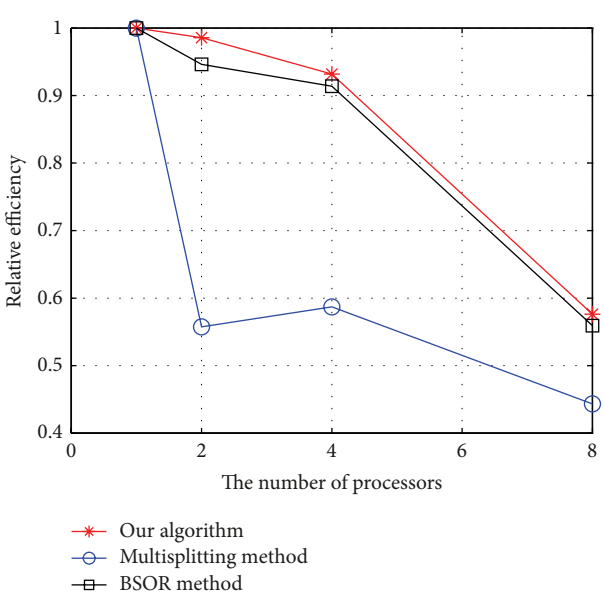

Figure 2: The parallel efficiency for Example 1.

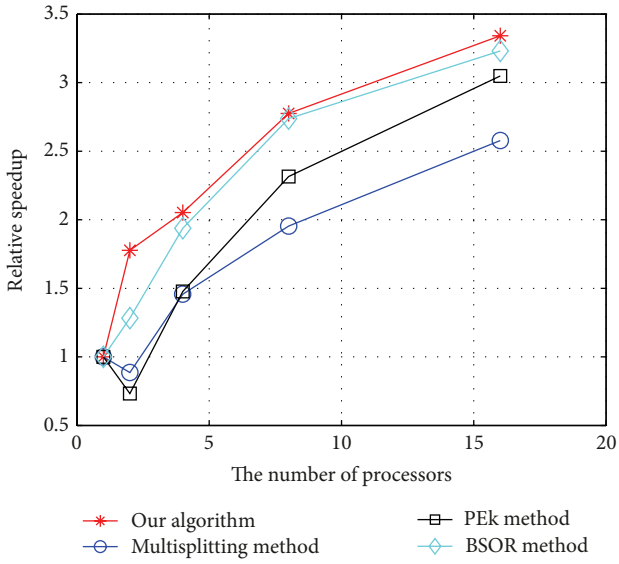

Figure 3: The parallel speedup for Example 2.

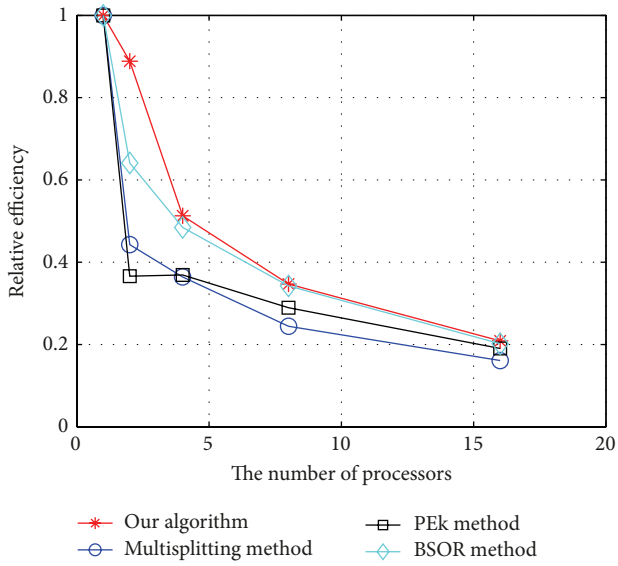

Figure 4: The parallel efficiency for Example 2.

(iii) As to Examples 1 and 2, the results of the examples show that the efficiency of the algorithm is better than the multisplitting ones. Our algorithm has good parallel speedup the same as BSOR methods to the examples. As to Example 2, the efficiency of the algorithm is also better than PEk methods.

(iv) The parallel algorithm is easily implemented on parallel computer and more flexible and simple than [1] in practice.

7. Conclusions

An efficient parallel iterative method on a distributed-memory multicomputer has been presented for solving the large banded linear systems. We make full use of the decomposition of the coefficient matrix to choose **W** and **V** to save computational cost. The storage strategy can save memory space. Only twice it requires the communications of the algorithm between the adjacent processors. Theoretical analysis and experiment show that the algorithm in this paper has good parallelism and high efficiency. The results also confirm correctness of convergence theorems. When the coefficient

Table 4: The results for model 2 (the algorithm in the paper ($\tau = 8.0$, $\alpha = \beta = 1/2$)).

P	1	2	4	8	16
T	11.3091	6.3632	5.5117	4.0755	3.3842
S		1.7773	2.0152	2.7749	3.3417
E		0.8886	0.5130	0.3469	
E		0.8886	0.5130	0.3469	0.2089
L	1177	1177	1177	1177	1186
Δ	$0.8163e-10$	$0.8163e-10$	$0.8163e-10$	$08163e-10$	$0.8140e-10$

Table 5: The results for model 2 (the multisplitting method).

P	1	2	4	8	16
T	15.3559	17.3404	10.5411	7.8602	5.9567
S		0.8856	1.4568	1.9536	2.5779
E		0.8886	0.5130	0.3469	
E		0.4428	0.3642	0.2442	0.1611
L	310	824	975	1335	1556

Table 6: The results for model 2 (PEk method ($k = \mathbf{2.7}$)).

P	1	2	4	8	16
T	14.6964	20.0765	9.9533	6.3488	4.8215
S		0.7320	1.4765	2.3148	3.0481
E		0.8886	0.5130	0.3469	
E		0.3660	0.3691	0.2894	0.1905
L	159	444	444	444	444

Table 7: The results for model 2 (BSOR method).

P	1	2	4	8	16
T	27.7668	21.6576	14.3278	10.1420	8.5949
S		1.2821	1.9380	2.7378	3.2306
E		0.8886	0.5130	0.3469	
E		0.6410	0.4845	0.3422	0.2019
L	660	1039	1337	2101	2175

matrix is a Hermite positive definite matrix or an M-matrix, we know that the parallel algorithm is convergent if the given conditions are established. Our algorithm has an advantage over the multisplitting one of high efficiency.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant nos. 11002117 and 11302173 and Xianyang Normal University Research Foundation under Grant nos. 09XSYK209 and 09XSYK204.

References

[1] B. Zhang, T. Gu, and Z. Mo, *Principles and Methods of Numerical Parallel Computation*, National Defence Industry Press, 1999.

[2] Q. Lü and T. Ye, "An improve parallel algorithm for solving linear equations involving block tridiagonal coefficient matrix," *Journal of Northwestern Polytechnical University*, vol. 4, no. 2, pp. 314–317, 1996.

[3] J. Wu, J. Song, W. Zhang, and X. Li, "Parallel incomplete factorization preconditioning of block tridiagonal linear systems with 2-D domain decomposition," *Chinese Journal of Computational Physics*, vol. 26, no. 2, pp. 191–199, 2009.

[4] Z. Duan, Y. Yang, Q. Lv, and X. Ma, "Parallel strategy for solving block-tridiagonal linear systems," *Computer Engineering and Applications*, vol. 47, no. 13, pp. 46–49, 2011.

[5] Y. Fan, *The Parallel Algorithms for Solving the Large Scale Linear Systems with Typical Structure*, Northwestern Polytechnical University Press, Xi'an, China, 2009.

[6] Z.-G. Luo and X.-M. Li, "Parallel algorithm for block-tridiagonal linear systems on distributed-memory multicomputers," *Chinese Journal of Computers*, vol. 23, no. 10, pp. 1028–1034, 2000.

[7] S. M. El-Sayed, "A direct method for solving circulant tridiagonal block systems of linear equations," *Applied Mathematics and Computation*, vol. 165, no. 1, pp. 23–30, 2005.

[8] X. Cui and Q. Lü, "A parallel algorithm for block-tridiagonal linear systems," *Applied Mathematics and Computation*, vol. 173, no. 2, pp. 1107–1114, 2006.

[9] R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1962.

[10] J. Hu, *Iterative Method of Linear Algebraic Equations*, Science Press, Beijing, China, 1999.

[11] A. Frommer and D. B. Szyld, "Weighted max norms, splittings, and overlapping additive Schwarz iterations," *Numerische Mathematik*, vol. 83, no. 2, pp. 259–278, 1999.

[12] J. Feng, G. Che, and Y. Nie, *Principle of Numerical Analysis*, Science Press, Beijing, China, 2002.

[13] P. Bjørstad and M. Luskin, *Parallel Solution of Partial Differential Equations*, Springer, New York, NY, USA, 2000.

[14] W. H. Reed and T. R. Hill, "Triangle mesh methods for the Neutron transport equation," Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.

[15] Y. P. Cheng, *Matrix Theory*, Northwestern polytechnical University Press, 2002.