J. Math. Study
doi: 10.4208/jms.v48n1.15.01

Vol. **48**, No. 1, pp. 1-17
March 2015

Modulus-based GSTS Iteration Method for Linear Complementarity Problems

Min-Li Zeng[1,2], Guo-Feng Zhang[1,*]

[1] *School of Mathematics and Statistics, Lanzhou University, Lanzhou 730000, Gansu Province, P.R. China.*
[2] *College of Mathematics, Putian University, Putian 351100, Fujian Province, P.R. China.*

Received 3 October, 2014; Accepted (in revised version) 14 December, 2014

Abstract. In this paper, a modulus-based generalized skew-Hermitian triangular splitting (MGSTS) iteration method is present for solving a class of linear complementarity problems with the system matrix either being an H_+-matrix with non-positive off-diagonal entries or a symmetric positive definite matrix. The convergence of the MGSTS iteration method is studied in detail. By choosing different parameters, a series of existing and new iterative methods are derived, including the modulus-based Jacobi (MJ) and the modulus-based Gauss-Seidel (MGS) iteration methods and so on. Experimental results are given to show the effectiveness and feasibility of the new method when it is employed for solving this class of linear complementarity problems.

AMS subject classifications: 65F10, 65F50, 65G40, 90C33

Key words: GSTS method, modulus-based iteration method, linear complementarity problem, H_+-matrix, symmetric positive definite matrix.

1 Introduction

Consider the following linear complementarity problem

$$\vec{w} := A\vec{z} + \vec{q} \geq 0, \quad \vec{z} \geq 0 \quad \text{and} \quad \vec{z}^T\vec{w} = 0, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a large sparse matrix, $\vec{z} \in \mathbb{R}^n$ is an unknown vector and $\vec{q} = (q_1, q_2, \cdots, q_n)^T \in \mathbb{R}^n$ is a given vector. In the sequel, we abbreviate the linear complementarity problem

*Corresponding author.
Email addresses: gf_zhang@lzu.edu.cn(G.-F. Zhang), zengml12@lzu.edu.cn (M.-L. Zeng)

 ©2015 Global-Science Press

(1.1) as LCP($\vec{q}$,A). The notation "$\geq$" means the componentwise defined partial ordering between two vectors and "T" in the superscript denotes the transpose of a vector.

The LCP($\vec{q}$,A) arises in many scientific computing and engineering applications, e.g., the contact problem, the Nash equilibrium point of a bimatrix game, the free boundary problem and the optimal stopping in Markov chain and so on. For more details, see [6, 17, 19] and the references therein.

To get the numerical solution for the large and sparse LCP($\vec{q}$,A), many efficient methods have been presented based on linear algebraic equations, for example, the projected iterative methods with the system matrix being symmetric positive definite(SPD), symmetric positive semi-definite and diagonally dominant ([1, 16, 18]), the modulus-based iterative method ([2, 4–13, 15, 17, 20, 21]) and so on. The main drawback of the projected methods is that we have to project the iterative solution onto the space $\mathbb{R}^n_+ = \{x \in \mathbb{R}^n | x \geq 0\}$, which is a costly and complicated work in actual implementations. Especially, it is much more difficult when the system matrix is nonsymmetric or some zero entries appear on the diagonal position.

Recently, Bai in [5] presented a modulus-based matrix splitting iteration method. The method not only covers the known modulus iteration methods and the corresponding modified variants, but also yields a series of modulus-based relaxation methods. For example, the MJ, the MGS, the modulus-based SOR method (MSOR) ([15]), the modified modulus method ([11]) and the non-stationary extrapolated modulus algorithm ([12]). Besides, if the system matrix is an H_+-matrix, the improved modulus-based matrix splitting iteration method turns to the scaled extrapolated modulus algorithms ([13]) and the two-step modulus-based matrix splitting iteration methods ([20]), respectively.

In this paper, based on the generalized skew-Hermitian triangular splitting (GSTS) iteration method ([14]) and the modulus-based matrix splitting iteration methods ([5]), we present a modulus-based GSTS (MGSTS) iteration method for solving large sparse LCP($\vec{q}$,A). By choosing different parameter matrices, we derive a series of existing and new iterative methods, including MJ, MGS, AMJ(the accelerated MJ), AMGS(the accelerated MGS) and AMSOR (the accelerated MSOR) methods. Experimental results are given to show the effectiveness and feasibility of the new method when it is employed for solving the linear complementarity problems with the system matrix either H_+-matrix with non-positive off-diagonal entries or symmetric positive definite.

The paper is organized as follows. In Section 2, some necessary notations and definitions are introduced, some modulus-based matrix splitting iteration methods are reviewed. Then the MGSTS iteration method for solving large sparse LCP($\vec{q}$,A) is established and some special modulus-based methods are given, respectively. In Section 3, when the system matrix is an H_+-matrix with non-positive off-diagonal entries or a symmetric positive definite matrix, the convergence conditions are presented. In Section 4, numerical examples are given to show the performance of the proposed method. Finally in Section 5, we end this paper with some concluding remarks.

2 The MGSTS iteration method

First we will introduce some notations and concepts. Let $A=(a_{ij})$ and $B=(b_{ij})$ be two real $m\times n$ matrices. Then $A\geq B(A>B)$ means $a_{ij}\geq b_{ij}(a_{ij}>b_{ij})$ for all $1\leq i\leq m$ and $1\leq j\leq n$. Let $|A|=(|a_{ij}|)\in\mathbb{R}^{m\times n}$ be the absolute value of the matrix A, and A^T be the transpose of A.

We call a real matrix A is Z-matrix if the off-diagonal entries of A are non-positive; an M-matrix if A is a Z-matrix and $A^{-1}\geq 0$; an H-matrix if its comparison matrix $\langle A\rangle=(\langle a\rangle_{ij})\in\mathbb{R}^{n\times n}$ is an M-matrix, where

$$\langle a\rangle_{ij}=\begin{cases}|a_{ij}|, & \text{for} \quad i=j\\ -|a_{ij}|, & \text{for} \quad i\neq j\end{cases} \quad i,j=1,2,\ldots,n.$$

Particularly, a matrix A is called an H_+-matrix if A is an H-matrix with positive diagonal entries.

Given a matrix $A\in\mathbb{R}^{n\times n}$. A splitting $A=M-N$ is called an M-splitting if M is a nonsingular M-matrix and $N\geq 0$; an H-compatible splitting if $\langle A\rangle=\langle M\rangle-|N|$. It is known that if $A=M-N$ is an M-splitting and A is nonsingular M-matrix, then $\rho(M^{-1}N)<1$, where $\rho(M^{-1}N)$ denotes the spectral radius of the matrix $M^{-1}N$. For more details, we refer to [5].

Lemma 2.1. ([5]) *Let $A=M-N$ be a splitting of the matrix $A\in\mathbb{R}^{n\times n}$, Ω_1, Ω_2, $\Omega:=\Omega_1+\Omega_2$ and Γ be $n\times n$ positive diagonal matrices. For the LCP($\vec{q}$,A)* (1.1)*, the following statements hold true:*

- *(i) if $(\vec{w},\vec{z})$ is a solution of the LCP($\vec{q}$,A), then $\vec{x}=\frac{1}{2}(\Gamma^{-1}\vec{z}-\Omega^{-1}\vec{w})$ satisfies the implicit fixed-point equation*

$$(M\Gamma+\Omega_1)\vec{x}=(N\Gamma-\Omega_2)\vec{x}+(\Omega-A\Gamma)|\vec{x}|-q. \tag{2.1}$$

- *(ii) if $\vec{x}$ satisfies the implicit fixed-point equation* (2.1)*, then*

$$\vec{z}=\Gamma(|\vec{x}|+\vec{x}) \quad \textit{and} \quad \vec{w}=\Omega(|\vec{x}|-\vec{x}) \tag{2.2}$$

  is a solution of LCP($\vec{q}$,A).

By taking $\vec{z}=\frac{1}{\gamma}(|\vec{x}|+\vec{x}),\vec{w}=\frac{1}{\gamma}\Omega(|\vec{x}|-\vec{x})$ and $A=M-N$, the LCP($\vec{q}$,A) can be equivalently written into a system of fixed-point equations([5])

$$(\Omega+M)\vec{x}=N\vec{x}+(\Omega-A)|\vec{x}|-\gamma\vec{q}. \tag{2.3}$$

Based on (2.3), Bai ([5]) established the following modulus-based matrix splitting iteration method for solving LCP($\vec{q}$,A):

$$(\Omega+M)\vec{x}^{(k+1)}=N\vec{x}^{(k)}+(\Omega-A)|\vec{x}^{(k)}|-\gamma\vec{q}, \tag{2.4}$$
$$\vec{z}^{(k+1)}:=\frac{1}{\gamma}(|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)}).$$

Zheng in [22] further established the accelerated modulus-based iteration method as follows.

$$(M_1+\Omega)\vec{x}^{(k+1)}=N_1\vec{x}^{(k)}+(\Omega-M_2)|\vec{x}^{(k)}|+N_2|\vec{x}^{(k+1)}|-\gamma\vec{q}, \tag{2.5a}$$
$$\vec{z}^{(k+1)}:=\frac{1}{\gamma}(|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)}). \tag{2.5b}$$

Here, $A=M_1-N_1=M_2-N_2$ are two splittings of the matrix $A\in\mathbb{R}^{n\times n}$, Ω is a $n\times n$ positive diagonal matrix and γ is a positive constant.

Now if we let $A=M-N$ be a splitting of the matrix A with

$$M=\frac{1}{\tau}B(\omega_1,\omega_2),\quad N=\frac{1}{\tau}C(\omega_1,\omega_2,\tau), \tag{2.6}$$

where

$$B(\omega_1,\omega_2)=(B_c+\omega_1K_L)B_c^{-1}(B_c+\omega_2K_U), \tag{2.7a}$$
$$C(\omega_1,\omega_2,\tau)=B_c-\tau A_H+\omega_1\omega_2K_LB_c^{-1}K_U-(\tau-\omega_1)K_L-(\tau-\omega_2)K_U, \tag{2.7b}$$

with $B_c\in\mathbb{R}^{n\times n}$ being a symmetric positive definite matrix, K_L and K_U being the strictly lower triangular and the strictly upper triangular matrix, respectively, ω_1 and ω_2 being two acceleration parameters.

Then we present our modulus-based GSTS (MGSTS) iteration method for LCP($\vec{q}$,A) as follows.

MGSTS iteration method: Given an initial vector $\vec{x}^{(0)}\in\mathbb{R}^n$, a $n\times n$ positive diagonal matrix Ω, a symmetric positive definite matrix B_c and three positive parameters ω_1, ω_2 and τ to obtain the matrices M and N defined by (2.6). For $k=0,1,2,\cdots$ until the iteration sequence $\{\vec{z}^{(k)}\}_{k=0}^{+\infty}$ is convergent, compute $\vec{x}^{(k+1)}\in\mathbb{R}^n$ by solving the linear system

$$(M+\Omega)\vec{x}^{(k+1)}=N\vec{x}^{(k)}+(\Omega-A)|\vec{x}^{(k)}|-\gamma\vec{q}, \tag{2.8}$$

and set

$$\vec{z}^{(k+1)}:=\frac{1}{\gamma}\Big(|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)}\Big).$$

Remark 1. If we split $A=A_H+A_S$, where

$$A_H=\frac{1}{2}(A+A^T),\quad A_S=\frac{1}{2}(A-A^T)=K_L+K_U,$$

K_L and K_U are the strictly lower-triangular and the strictly upper-triangular parts of A_S. Given a symmetric positive definite matrix B_c. Then we obtain the generalized skew-Hermitian triangular splitting (GSTS) matrix as

$$M=\frac{1}{\tau}B(\omega_1,\omega_2)=\frac{1}{\tau}(B_c+\omega_1K_L)B_c^{-1}(B_c+\omega_2K_U). \tag{2.9}$$

Remark 2. If we split $A=D-L-U$, and let $B_c:=D$, $K_L:=L$ and $K_U:=U$, where D, L and U are the diagonal part, the minus strictly lower-triangular part and the minus strictly upper-triangular part of A, respectively. Then we have

$$M=\frac{1}{\tau}B(\omega_1,\omega_2)=\frac{1}{\tau}(D+\omega_1L)D^{-1}(D+\omega_2U). \tag{2.10}$$

Remark 3. If we split $A=M_1-N_1=M_2-N_2$, and let M_1 and N_1 be defined by (2.6a) and (2.6b), respectively. Then we obtain an accelerated modulus-based GSTS (AMGSTS) iteration method from the MGSTS iteration method when we compute $\vec{x}^{(k+1)}\in\mathbb{R}^n$ by solving the linear system (2.5a).

By choosing different matrices B_c and Ω and accelerate parameters ω_1, ω_2, τ and γ, we can easily get a series of iterative algorithms from the MGSTS iteration method for solving the LCP($\vec{q}$,A) (1.1).

Case 1. If $B_c=A$, $\omega_1=\omega_2=0$, $\tau=1$, $\gamma=1$ and $\Omega=I$. Then the MGSTS iteration method reduces to the modulus iteration method ([5])

$$(I+A)\vec{x}^{(k+1)}=(I-A)|\vec{x}^{(k)}|-\vec{q}$$

with $\vec{z}^{(k+1)}=|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)}$.

Case 2. If $B_c=\alpha I$, $\omega_1=\omega_2=0$, $\tau=1$, $\gamma=1$ and $\Omega=\alpha I$. Then the MGSTS iteration method becomes the modified modulus iteration method in [11] as

$$(\alpha I+A)\vec{x}^{(k+1)}=(\alpha I-A)|\vec{x}^{(k)}|-\vec{q}$$

with $\vec{z}^{(k+1)}=|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)}$.

Case 3. If $B_c=D$, $\omega_1=-1$,$\omega_2=0$, $\tau=1$ and $\gamma=2$, the AMGSTS iteration method yields the accelerated modulus-based Jacobi iteration method ([22]), denoted as AMJ,

$$(D+\Omega)\vec{x}^{(k+1)}=(L+U)\vec{x}^{(k)}+(\Omega-M_2)|\vec{x}^{(k)}|+N_2|\vec{x}^{(k+1)}|-2\vec{q}$$

with $\vec{z}^{(k+1)}=\frac{1}{2}(|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)})$.

Case 4. If $B_c=D$, $\omega_1=-1$,$\omega_2=0$, $\tau=1$ and $\gamma=2$, the AMGSTS iteration method reduces to the accelerated modulus-based Gauss-Seidel iteration method ([22]), denoted as AMGS,

$$(D+\Omega-L)\vec{x}^{(k+1)}=U\vec{x}^{(k)}+(\Omega-M_2)|\vec{x}^{(k)}|+N_2|\vec{x}^{(k+1)}|-2\vec{q}$$

with $\vec{z}^{(k+1)}=\frac{1}{2}(|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)})$.

Case 5. If $B_c=\frac{1}{\alpha}D$, $\omega_1=-1$,$\omega_2=0$, $\tau=1$ and $\gamma=2$, the AMGSTS iteration method yields the accelerated modulus-based SOR iteration method ([22]) , denoted as AMSOR,

$$(D+\Omega-\alpha L)\vec{x}^{(k+1)}=((1-\alpha)D+\alpha U)\vec{x}^{(k)}+(\Omega-\alpha M_2)|\vec{x}^{(k)}|+\alpha N_2|\vec{x}^{(k+1)}|-2\alpha\vec{q}$$

with $\vec{z}^{(k+1)}=\frac{1}{2}(|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)})$.

Case 6. If $B_c=\frac{1}{\alpha}D$, $\omega_1=-\frac{\beta}{\alpha}$,$\omega_2=0$, $\tau=1$ and $\gamma=2$, the MGSTS iteration method becomes the modulus-based AOR iteration method ([15])

$$(D+\Omega-\beta L)\vec{x}^{(k+1)}=((1-\alpha)D+(\alpha-\beta)L+\alpha U)\vec{x}^{(k)}+(\Omega-\alpha A)|\vec{x}^{(k)}|-2\alpha\vec{q}$$

with $\vec{z}^{(k+1)}=\frac{1}{2}(|\vec{x}^{(k+1)}|+\vec{x}^{(k+1)})$.

We can give some new MGSTS iterative methods by choosing different parameters. Some choices for the parameters are listed in Table 1. In Table 1, $A_H:=\frac{1}{2}(A+A^T)$, $A_S:=$

Table 1: Some choices of parameters.

Method	B_c	K_L	K_U	p	ω_1	ω_2	τ
MGSTS(1)	diag(A)	tril(A)	triu(A)	p_{exp}	1	$\omega_{2,exp}$	1
MGSTS(2)	A_H	tril(A_S)	triu(A_S)	p_{exp}	0	$\omega_{2,exp}$	0.6
MGSTS(3)	A_H	tril(A_S)	triu(A_S)	p_{exp}	$\omega_{1,exp}$	0	0.6
MGSTS(4)	A_H	tril(A_S)	triu(A_S)	0.01	*	*	0.6
MGSTS(5)	A_H	tril(A_S)	triu(A_S)	0.01	1.3	0.7	0.6
MGSTS(6)	A_H	tril(A_S)	triu(A_S)	6	1.3	0.7	0.6

$\frac{1}{2}(A-A^T)$, diag(A) is the diagonal parts of A. tril(A) and triu(A) represent the strictly lower-triangular part and the strictly upper-triangular part of the matrix A, respectively. P_{exp}, $\omega_{1,exp}$ and $\omega_{2,exp}$ are the experimental optimal parameters.

3 Convergence analysis of the MGSTS iteration method

In this section, we concentrate on the convergence of the MGSTS iteration method with the splitting matrix M defined in (2.9) and the system matrix A of the LCP($\vec{q}$,A) being an H_+-matrix with non-positive off-diagonal entries or a symmetric positive definite matrix.

If the vector pair $(\vec{z}^*,\vec{w}^*)\in\mathbb{R}^n_+\times\mathbb{R}^n_+$ is a solution of the LCP($\vec{q}$,A), then $\vec{x^*}=\frac{1}{2}\gamma(\vec{z}^*-\Omega^{-1}\vec{w}^*)$ obviously holds the fixed-point equation

$$(M+\Omega)\vec{x}^*=N\vec{x}^*+(\Omega-A)|\vec{x}^*|-\gamma q. \tag{3.1}$$

After subtracting (3.1) from (2.4), we can immediately obtain

$$\vec{x}^{(k+1)}-\vec{x}^*=(M+\Omega)^{-1}(N(\vec{x}^{(k)}-\vec{x}^*)+(\Omega-A)(|\vec{x}^{(k)}|-|\vec{x}^*|)). \tag{3.2}$$

Therefore, to prove $\lim_{k\to+\infty}\vec{z}^{(k)}=\vec{z}^*$, we only need to demonstrate the convergence of the sequence $\{\vec{x}^{(k)}\}_{k=0}^{+\infty}$ generated by the MGSTS method. We will use the error relationship (3.2) to establish our convergence theorem.

3.1 A is an H_+-matrix with non-positive off-diagonal entries

Assume that $A\in\mathbb{R}^{n\times n}$ is an H_+-matrix with non-positive off-diagonal entries. If $K_L>0$, $\omega_2=0$ and $B_c=A_H$, then the splitting matrix of the form (2.9) can be rewritten as

$$M=\frac{1}{\tau}(A_H+\omega_1K_L). \tag{3.3}$$

If $K_L<0$, we will let $\omega_1=0$ and $B_c=A_H$. Then the splitting matrix of the form (2.9) can be rewritten as

$$M=\frac{1}{\tau}(A_H+\omega_2K_U). \tag{3.4}$$

It can be easily obtained that $A_H=\frac{1}{2}(A+A^T)$ is a Z-matrix with the same diagonal entries as A. We can establish the following convergence theorem for the MGSTS method.

Theorem 3.1. *Let $A\in\mathbb{R}^{n\times n}$ be an H_+-matrix with non-positive off-diagonal entries, $A=A_H+K_L+K_U$ with K_L and K_U being the strictly lower-triangular and the strictly upper-triangular parts of A_S, respectively. Assume that $A=M-N$ is a splitting of A with M defined by Eq.* (3.3) *or Eq.* (3.4)*, the positive diagonal matrix Ω satisfies $\Omega\geq\frac{1}{2}diag(M)$ and γ is a positive constant. If one of the following conditions holds:*

- *(1) when $K_L>0$, the positive parameter τ and the accelerated parameter ω_1 make $M=\frac{1}{\tau}(A_H+\omega_1K_L)$ to be a Z-matrix and*

$$(\frac{1}{\tau}-1)A_H+(\frac{\omega_1}{\tau}-1)K_L-K_U\geq 0; \tag{3.5}$$

- *(2) when $K_L<0$, the positive number τ and the accelerated parameter ω_2 make $M=\frac{1}{\tau}(A_H+\omega_1K_U)$ to be a Z-matrix and*

$$(\frac{1}{\tau}-1)A_H+(\frac{\omega_2}{\tau}-1)K_U-K_L\geq 0. \tag{3.6}$$

Then the iteration sequence $\{\vec{z}^{(k)}\}_{k=0}^{+\infty}\subset\mathbb{R}_+^n$ generated by the MGSTS method converges to the unique solution $\vec{z}_*\in\mathbb{R}_+^n$ of the LCP($\vec{q}$,A) (1.1) for any initial vector $\vec{x}^{(0)}\in\mathbb{R}^n$.

Proof. If $K_L>0$, then $K_U=-K_L^T<0$. It follows from (3.5) that $\frac{1}{\tau}-1\geq 0$ and $\frac{\omega_1}{\tau}-1\geq 0$ hold, while yields $0<\tau\leq 1$ and $\omega_1\geq\tau>0$. So $M=\frac{1}{\tau}(A_H+\omega_1K_L)$ is a Z-matrix with positive diagonal entries when $A\in\mathbb{R}^{n\times n}$ is an H_+-matrix with non-positive off-diagonal entries. Then $\langle A\rangle=\langle M\rangle-|N|$. Therefore, $A=M-N$ is an H-compatible splitting of the

matrix A. According to Theorem 4.3 in [5], we can acquire that the iteration sequence $\{\vec{z}^{(k)}\}_{k=0}^{+\infty} \subset \mathbb{R}_+^n$ generated by the MGSTS method converges to the unique solution $\vec{z}_* \in \mathbb{R}_+^n$ of the LCP($\vec{q}$,A).

The convergence property for the case $K_L < 0$ can be obtained by the same way as $K_L > 0$. □

Corollary 1. Let $A \in \mathbb{R}^{n\times n}$ be an M-matrix and $A = M - N$ be a splitting of A described in Theorem 3.1. Assume that the positive diagonal matrix Ω satisfies $\Omega \geq \frac{1}{2}\text{diag}(M)$. Then for a given positive constant γ, the iteration sequence $\{\vec{z}^{(k)}\}_{k=0}^{+\infty} \subset \mathbb{R}_+^n$ generated by the MGSTS method converges to the unique solution $\vec{z}_* \in \mathbb{R}_+^n$ of the LCP($\vec{q}$,A) for any initial vector $\vec{x}^{(0)} \in \mathbb{R}^n$.

Proof. Because $N \geq 0$ and M is a nonsingular matrix. Then $A = M - N$ is an M-splitting of the matrix A. By the result of Theorem 4.5 in [5], the conclusion immediately follows. □

3.2 A is symmetric positive definite

Firstly, we review a directly result about the accelerated MGSTS (AMGSTS) iteration method from [22].

Lemma 3.1. ([22]) *Let $A \in \mathbb{R}^{n\times n}$ be a positive definite matrix, and $A = M_1 - N_1 = M_2 - N_2$ with $M_1 \in \mathbb{R}^{n\times n}$ being positive definite. Assume that $\Omega \in \mathbb{R}^{n\times n}$ is a positive diagonal matrix and γ is a positive constant. Define*

$$\xi(\Omega) = \|(\Omega + M_1)^{-1}N_1\|, \tag{3.7a}$$

$$\eta(\Omega) = \|(\Omega + M_1)^{-1}N_2\|, \tag{3.7b}$$

$$\mu(\Omega) = \|(\Omega + M_1)^{-1}(\Omega - M_1)\|. \tag{3.7c}$$

Then the iteration sequence $\{\vec{z}^{(k)}\}_{k=0}^{+\infty} \subset \mathbb{R}_+^n$ generated by the AMGSTS iteration method converges to the unique solution $\vec{z}^* \in \mathbb{R}^{n\times n}$ of the LCP($\vec{q}$,A) for any initial vector $\vec{x}^{(0)} \in \mathbb{R}^{n\times n}$, provided

$$\delta(\Omega) := \mu(\Omega) + 2\xi(\Omega) + 2\eta(\Omega) < 1. \tag{3.8}$$

When the system matrix A is symmetric positive definite, then K_L and K_U are both zeros matrices. Therefore, from the AMGSTS iteration method in Remark 3, the splitting matrix of the form (2.9) can be rewritten as

$$M_1 = \frac{1}{\tau}B_c, \tag{3.9}$$

where B_c is symmetric positive definite and $\tau > 0$ is a positive number. Obviously, the matrix M_1 is symmetric positive definite. Let $\Omega = pI \in \mathbb{R}^{n\times n}$ be a positive scalar matrix. Then the convergence results can be obtained immediately from Lemma 3.1.

Theorem 3.2. *Let $A\in\mathbb{R}^{n\times n}$ be a symmetric positive definite matrix, and $A=M_1-N_1=M_2-N_2$ with $M_1=\frac{1}{\tau}B_c\in\mathbb{R}^{n\times n}$ being symmetric positive definite and $M_2=A$. Assume that $\Omega=pI\in\mathbb{R}^{n\times n}$, ϕ_1 and ϕ_2 are the smallest and the largest eigenvalues of the matrix B_c, respectively, and $\nu:=\tau\cdot\|B_c^{-1}N_1\|<1$. Then the sequence $\{\vec{z}^{(k)}\}_{k=0}^{+\infty}\subset\mathbb{R}_+^n$ generated by the MGSTS iteration method converges to the unique solution $\vec{z}^*\in\mathbb{R}_+^n$ of the LCP(q,A) for any initial $\vec{x}^{(0)}\in\mathbb{R}^n$, provided the parameter p satisfies either of the following conditions:*

- *(1) (i) when $\nu^2\phi_2<\phi_1<\nu\phi_2$,*

$$\nu\phi_2<p<\frac{(1-\nu)\phi_1\phi_2}{\nu\phi_2-\phi_1}; \tag{3.10}$$

- *(1) (ii) when $\phi_1\geq\nu\phi_2$,*

$$p\tau>\nu\phi_2. \tag{3.11}$$

4 Numerical results

In this section, we examine the feasibility and effectiveness of the MGSTS iteration method for solving LCP($\vec{q}$,A) (1.1) in terms of both iteration steps (denoted by "IT") and the elapsed CPU time (denoted by "CPU") in seconds. We list the IT, CPU times and the norm of absolute residual vectors (denoted by "RES"). Here, "RES" are defined as

$$\mathrm{RES}(\vec{z}^{(k)}):=\|\min\{A\vec{z}^{(k)}+\vec{q},\vec{z}^{(k)}\}\|,$$

where $\vec{z}^{(k)}$ is the kth approximate solution to the LCP($\vec{q}$,A) and the minimum is taken componentwise.

All tests are performed in MATLAB R2013a on Intel(R) Core(TM) i7-3770 CPU 3.40 GHz and 8.00 GB of RAM, with machine precision 10^{-16}. In our computations, all runs of MGSTS method is started from the initial vector $\vec{x}^{(0)}=(1,0,1,0,\cdots,1,0,\cdots)^T\in\mathbb{R}^n$ and terminated if the current iteration satisfies either $\mathrm{RES}(\vec{z}^{(k)})<10^{-5}$ or the number of the prescribed iteration $k_{max}=1000$ is exceeded.

We compare the MGSTS method with the projected Gauss-Seidel (PGS), the projected successive over-relaxation (PSOR) methods([1, 3, 16]), the MJ, the MGS, the AMJ, the AMGS and the AMSOR method. We take $\gamma=2$ and $\Omega=pI$ in all iteration methods, in which the parameter p is chosen to minimize the corresponding iteration steps. The abbreviations of the corresponding terminologies are listed in the cases described in Section 2. Six different choices for the iteration parameters and parameter-matrices with respect to the MGSTS iteration method are used in our test experiments, see Table 1. We note that when the matrix A is symmetric, the MGSTS(2) iteration method is equivalent to MGSTS(3). They are independent of the parameters ω_1 and ω_2. The method MGSTS(4) is only for symmetric matrix A. The methods MGSTS(5) and MGSTS(6) are for nonsymmetric A, where MGSTS(5) is for $c=0$, and MGSTS(6) is for $c=4$, respectively.

Example 4.1. ([5, 22]) Consider the LCP($\vec{q}$,A) (1.1), in which $A \in \mathbb{R}^{n\times n}$ is given by $A = \hat{A}+cI$ $(c \geq 0)$, $\vec{q} = (-1,1,-1,1,\cdots,-1,1)^T \in \mathbb{R}^n$, $n := m^2$.

$$\hat{A} = \text{Tridiag}(-lI,S,-rI) = \begin{pmatrix} S & -rI & 0 & \cdots & 0 & 0 \\ -lI & S & -rI & \cdots & 0 & 0 \\ 0 & -lI & S & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \\ 0 & 0 & 0 & \cdots & S & -rI \\ 0 & 0 & 0 & \cdots & -lI & S \end{pmatrix} \in \mathbb{R}^{n\times n}$$

is a block-tridiagonal matrix,

$$S = \text{tridiag}(-l,4,-r) = \begin{pmatrix} 4 & -r & 0 & \cdots & 0 & 0 \\ -l & 4 & -r & \cdots & 0 & 0 \\ 0 & -l & 4 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \\ 0 & 0 & 0 & \cdots & 4 & -r \\ 0 & 0 & 0 & \cdots & -l & 4 \end{pmatrix} \in \mathbb{R}^{m\times m}$$

is a tridiagonal matrix, and $\vec{z}^* = (1,2,1,2,\cdots,1,2)^T \in \mathbb{R}^n$ is the unique solution of the LCP($\vec{q}$,A) (1.1).

4.1 When the system matrix is symmetric

If we choose $l=1$ and $r=1$ in Example 4.1, then the system matrix A is symmetric. Hence, it follows that $K_L = K_U = 0$ and the matrix M, N in (2.6a) and (2.6b) are independent on the parameters ω_1, ω_2. In this case, we list the experimental optimal parameter ranges in Tables 2 and 3 for the proposed methods with different problem scales with $c = 0$ and $c = 4$, respectively. The corresponding numerical results (iteration steps (IT) and CPU times (CPU)) with $c = 0$ and $c = 4$ are listed in Tables 4 and 5, respectively.

From the experimental results in Tables 4 and 5 it can be seen that the PSOR method requires less iteration steps than the MJ, the MGS and the MSOR methods. However, the projected methods cost much computing times. Those experimental results coincide

Table 2: The experimental optimal parameters for symmetric case with c=0.

m\Method:	MJ	MGS	AMJ	AMGS	AMSOR	MGSTS(1)		MGSTS(2)
						$\omega_{2,exp}$	p_{exp}	p_{exp}
10	3.8	1.7	1.7	1.1	1.8	2.4	1.3	[0.10,0.16]
20	3.9	1.7	1.8	0.6	1.3	2.5	1.1	[0.01,0.05]
30	3.8	1.7	1.8	0.4	1.1	2.7	1	[0.01,0.03]
40	3.8	1.6	1.8	0.3	1.0	2.7	0.9	[0.01,0.02]
50	3.8	1.6	1.8	0.3	1.0	2.8	1	[0.01,0.02]
60	3.8	1.6	1.8	0.3	1.0	2.8	1	0.01

Table 3: The experimental optimal parameters for symmetric case with c=4.

m\Method:	MJ	MGS	AMJ	AMGS	AMSOR	MGSTS(1)		MGSTS(2)
						$\omega_{2,exp}$	p_{exp}	p_{exp}
10	[7.3,7.6]	[6.8,6.9]	[6.8,6.9]	[6.7,7.0]	[8.0,8.3]	[2.0,2.7]	[6.2,6.6]	[1.8,2.1]
20	[7.2,7.8]	6.8	6.8	[6.5,7.0]	[7.8,8.3]	[1.7,2.7]	[6.0,6.5]	1.9
30	[7.2,7.9]	[6.5,7]	[6.5,7.0]	[6.5,6.8]	[7.9,8.1]	[1.8,2.7]	[6.1,6.3]	[1.5,2.2]
40	[7.2,7.6]	[6.6,6.9]	[6.6,6.9]	6.6	8.0	[1.9,2.7]	[6.1,6.3]	[1.5,2.1]
50	[7.3,7.5]	6.7	6.7	[6.4,7.0]	[7.8,8.0]	[1.9,2.7]	[6.1,6.3]	[1.5,2.1]
60	[7.1,7.8]	[6.4,7.1]	6.6	[6.4,6.6]	[7.8,7.9]	[1.9,2.7]	[6.2,6.3]	[1.6,2.1]

Table 4: Numerical results for symmetric case (c=0) .

	m	10	20	30	40	50	60
Method							
PGS	IT	145	502	>1000	>1000	>1000	>1000
	CPU	9.78	550.47	-	-	-	-
PSOR	IT	34	116	251	-	-	-
	CPU	2.26	127.09	1478.55	>5000	>5000	>5000
MJ	IT	280	994	-	-	-	-
	CPU	0.007	0.05	-	-	-	-
MGS	IT	131	427	970	-	-	-
	CPU	0.006	0.13	1.39	-	-	-
AMJ	IT	130	463	-	-	-	-
	CPU	0.21	11.80	-	-	-	-
AMGS	IT	30	59	90	119	150	185
	CPU	0.07	0.90	16.69	70.37	220.01	663.23
AMSOR	IT	31	62	91	120	153	186
	CPU	0.07	0.92	16.84	70.44	220.32	663.01
MGSTS(1)	IT	22	39	71	101	126	162
	CPU	0.001	0.034	0.34	1.46	4.40	11.77
MGSTS(2)	IT	9	10	11	11	11	11
	CPU	0.0005	0.008	0.04	0.15	0.35	0.64
MGSTS(4)	IT	10	10	11	11	11	13
	CPU	0.0008	0.008	0.047	0.16	0.36	0.68

with that in [5]. It can also been find that the AMJ method is not convergent when $c=0$. The AMGS and the AMSOR methods are convergent very slowly when the problem size increases. However, the MGSTS method keeps the most efficient when the parameter p is choose according to Tables 2 and 3. When $c=4$, the AMJ, the AMGS and the AMSOR

Table 5: Numerical results for symmetric case (c=4).

Method	m	10	20	30	40	50	60
PGS	IT	13	15	15	15	16	16
	CPU	0.86	16.46	98.73	312.52	973.34	2188.03
PSOR	IT	12	12	12	13	13	14
	CPU	0.81	13.34	70.50	253.81	784.12	1725.33
MJ	IT	20	22	23	23	23	24
	CPU	0.0004	0.009	0.02	0.13	0.45	0.72
MGS	IT	15	16	17	17	17	18
	CPU	0.0008	0.007	0.06	0.11	0.27	0.45
AMJ	IT	15	16	17	17	17	17
	CPU	0.02	0.39	2.05	6.43	15.86	30.21
AMGS	IT	11	12	12	13	13	13
	CPU	0.03	0.41	2.07	6.46	17.21	32.82
AMSOR	IT	12	12	12	12	13	13
	CPU	0.03	0.413	2.07	6.46	17.21	32.82
MGSTS(1)	IT	8	9	9	9	9	9
	CPU	0.0005	0.007	0.04	0.09	0.17	0.36
MGSTS(2)	IT	6	6	7	7	7	7
	CPU	0.0003	0.004	0.01	0.05	0.10	0.25
MGSTS(4)	IT	10	11	11	11	11	12
	CPU	0.0007	0.008	0.06	0.11	0.21	0.39

Table 6: The experimental optimal parameters for nonsymmetric case (c=0).

m\Method:	MJ	MGS	AMJ	AMGS	AMSOR	MGSTS(1)		MGSTS(2)		MGSTS(3)	
						$\omega_{2,exp}$	p_{exp}	$\omega_{2,exp}$	p_{exp}	$\omega_{1,exp}$	p_{exp}
10	4.1	2.1	2.1	2.2	2.9	2.0	[1.6,1.7]	0.8	0.5	1.8	2.9
20	4.1	1.8	1.8	2.1	2.8	1.8	1.2	0.8	0.5	1.8	3.0
30	4.0	1.7	1.7	2.0	2.7	1.7	[0.9,1.0]	0.7	0.5	1.8	3.0
40	4.0	1.6	1.6	2.0	2.7	1.6	0.8	0.7	0.5	1.8	3.0
50	4.0	1.5	1.5	2.0	2.7	1.6	0.7	0.7	0.5	1.8	3.0
60	4.0	1.4	1.5	2.0	2.7	1.5	0.6	0.6	0.5	1.8	3.0

methods become faster convergent than $c = 0$. Hence, the MGSTS(4) method is a good choice as a common way to solve the LCP($\vec{q}$,A) when the system matrix is symmetric.

Besides, we also test the case when we simply take $p = 0.01$, $\omega_1 = 1.3$ and $\omega_2 = 0.7$ for all problems sizes. From Tables 4 and 5 we see that the iteration steps and the computing times with respect to both the experimental optimal parameter and this simple choices

Table 7: The experimental optimal parameters for nonsymmetric case with c=4.

Method\m	10	20	30	40	50	60
MJ	[7.6,8.0]	[7.5,7.9]	[7.4,7.9]	[7.5,7.7]	[7.3,7.8]	[7.3,7.8]
MGS	[6.3,7.0]	[6.1,6.9]	[6.2,6.9]	[6.3,6.5]	[6.0,6.9]	[6.0,6.9]
AMJ	[6.3,7.0]	[6.1,6.9]	[6.2,6.7]	[6.3,6.5]	[6.1,6.6]	[6.2,6.5]
AMGS	[6.6,7.5]	[6.7,7.3]	[6.8,7.2]	[6.9,7.1]	[6.9,7.1]	[6.9,7.0]
AMSOR	[7.9,8.8]	[8.1,8.6]	[8.1,8.5]	[8.2,8.4]	[8.2,8.4]	[8.2,8.3]
MGSTS(1)						
$\omega_{2,exp}$	[2.1,2.7]	[1.9,3.0]	[2.0,3.0]	[2.1,3.0]	[2.2,3.0]	[2.3,3.0]
p_{exp}	[6.3,6.5]	[6.1,6.3]	[6.1,6.3]	[6.1,6.3]	[6.0,6.2]	[6.0,6.2]
MGSTS(2)						
$\omega_{2,exp}$	[0.6,0.7]	[0.6,0.8]	[0.6,0.8]	[0.6,0.7]	[0.6,0.7]	[0.6,0.7]
p_{exp}	[1.7,1.8]	[1.4,1.7]	[1.4,1.6]	[1.4,1.6]	[1.4,1.6]	[1.4,1.5]
MGSTS(3)						
$\omega_{1,exp}$	[1.1,1.4]	[1.1,1.4]	[1.1,1.4]	[1.2,1.4]	[1.1,1.4]	[1.2,1.4]
p_{exp}	[3.2,3.5]	[3.2,3.5]	[3.2,3.5]	[3.2,3.5]	[3.2,3.4]	[3.3,3.4]

are almost the same. The MGSTS(4) method also performs quite efficiently.

4.2 When the system matrix is nonsymmetric

If we let $l = 1.5$ and $r = 0.5$, then the system matrix A is nonsymmetric and $K_L > 0$. According to Theorem 3.1, we will take $\tau = 0.6 \in (0,1]$ for both the MGSTS(2) and MGSTS(3) methods defined by Table 1.

Tables 6 and 7 list the experimental optimal parameter ranges for the proposed method with respect to different problem scales with $c = 0$ and $c = 4$, respectively. Tables 8 and 9 list the numerical results with respect to IT, CPU and RES for the testing methods for Example 4.1, with respect to varying m.

From the numerical results in Tables 8 and 9 it can be found once again that the projected methods require less iteration steps but much computing times than the MJ, the MGS and the AMJ methods. It can been also seen that the MGSTS method is superior to other methods if the optimal parameters are employed. It needs more iterative steps for the MGSTS(2) method than the MGSTS(3) method.

From these tables, we see that the MGSTS method always outperforms the MJ method, the MGS method and the AMJ method considerably in iteration steps. The MGSTS method almost has the same efficiency as that of the accelerated modulus-based methods considerably in iteration steps and residual errors.

Besides, we also choose some fixing parameters. For example, when $c = 0$ and $c = 4$, we use the MGSTS(5) and MGSTS(6) methods in Table 1, respectively. We see from Table 8 and Table 9 that, they are both very efficient.

Table 8: Numerical results for Example 4.1 for nonsymmetric case with c=0.

Method	m	10	20	30	40	50	60
PGS	IT	39	60	78	94	-	-
	CPU	2.63	66.12	462.28	1833.95	>5000	>5000
PSOR	IT	34	42	53	64	-	-
	CPU	2.26	46.21	312.81	1244.14	>5000	>5000
MJ	IT	92	155	211	263	313	363
	CPU	0.002	0.08	0.16	0.52	1.53	2.71
MGS	IT	43	63	81	96	110	126
	CPU	0.001	0.02	0.12	0.41	1.12	2.57
AMJ	IT	42	63	81	96	110	123
	CPU	0.07	1.58	10.23	38.48	53.84	77.49
AMGS	IT	13	14	15	15	16	16
	CPU	0.03	0.49	2.62	8.28	21.61	44.64
AMSOR	IT	13	15	15	15	16	16
	CPU	0.03	0.50	2.62	8.28	21.60	44.65
MGSTS(1)	IT	17	25	32	37	43	49
	CPU	0.001	0.021	0.15	0.53	1.48	2.52
MGSTS(2)	IT	16	29	39	49	58	69
	CPU	0.001	0.03	0.19	0.70	1.60	3.65
MGSTS(3)	IT	9	10	10	11	11	11
	CPU	0.0004	0.007	0.04	0.14	0.36	0.75
MGSTS(5)	IT	10	13	16	18	20	22
	CPU	0.0005	0.008	0.06	0.16	0.44	0.97

Finally in Table 10, we list some numerical results using the MGSTS(4), MGSTS(5), MGSTS(6) methods for high dimensions LCP($\vec{q}$,A). We can find that they perform efficiently.

It can be drawn a conclusion that, when the system matrix is symmetric for the LCP($\vec{q}$,A), we can use the MGSTS(4) method. When the system matrix is nonsymmetric for the LCP($\vec{q}$,A) and $c=0$, we can use the MGSTS(5) method. When the system matrix is nonsymmetric for the LCP($\vec{q}$,A) and $c=4$, we can use the MGSTS(6) method.

5 Conclusions

In this paper, we have proposed a modulus-based generalized skew-Hermitian triangular splitting (MGSTS) method for solving a class of linear complementarity problems with the system matrix being either an H_+-matrix with non-positive off-diagonal entries or a symmetric positive definite matrix. The convergence conditions are given. Numerical ex-

Table 9: Numerical results for nonsymmetric case (c=4).

Method	m	10	20	30	40	50	60
PGS	IT	10	10	11	11	11	11
	CPU	0.69	11.04	64.91	213.86	634.30	1413.82
PSOR	IT	8	9	9	9	9	9
	CPU	0.61	11.19	58.77	201.13	512.24	1127.08
MJ	IT	20	22	23	23	24	24
	CPU	0.0009	0.006	0.05	0.17	0.48	0.69
MGS	IT	13	14	14	14	15	15
	CPU	0.0005	0.004	0.03	0.10	0.28	0.38
AMJ	IT	13	14	14	14	15	15
	CPU	0.02	0.33	1.67	5.28	13.85	27.49
AMGS	IT	8	8	8	8	8	8
	CPU	0.02	0.26	1.32	4.14	10.11	20.94
AMSOR	IT	8	8	8	8	8	8
	CPU	0.02	0.26	1.32	4.13	10.11	20.93
MGSTS(1)	IT	9	25	32	37	43	49
	CPU	0.001	0.02	0.15	0.53	1.48	3.02
MGSTS(2)	IT	16	29	39	49	58	69
	CPU	0.001	0.02	0.19	0.71	2.01	3.96
MGSTS(3)	IT	9	10	10	11	11	11
	CPU	0.0005	0.004	0.01	0.08	0.21	0.28
MGSTS(6)	IT	12	13	14	14	14	14
	CPU	0.0004	0.004	0.03	0.11	0.27	0.35

Table 10: Numerical results for high dimensions.

(l,r)	Method	m	70	80	90	100
c=0:						
(1,1)	MGSTS(4)	IT	16	20	24	28
		CPU	1.05	1.85	3.16	5.91
(1.5,0.5)	MGSTS(5)	IT	24	25	27	29
		CPU	1.34	2.08	3.58	6.04
c=4:						
(1,1)	MGSTS(4)	IT	12	12	12	12
		CPU	0.85	1.55	1.61	2.48
(1.5,0.5)	MGSTS(6)	IT	14	14	15	15
		CPU	0.65	1.13	1.91	3.18

periments show the effectiveness of the MGSTS method for the test problems. However, the choice of the parameters will be studied in the future work.

Acknowledgments

This work was supported by NSFC (No.11271174).

References

[1] B. H. Ahn. Solution of nonsymmetric linear complementarity problems by iterative methods. Journal of Optimization Theory and Applications, 1981, 33: 175–185.
[2] Z.-Z. Bai. The convergence of parallel iteration algorithms for linear complementarity problems. Computers and Mathematics with Applications, 1996, 32: 1–17.
[3] Z.-Z. Bai. On the monotone convergence of the projected iteration methods for linear complementarity problems. Numerical Mathematics, 1996, 5: 228–233.
[4] Z.-Z. Bai. On the convergence of the multisplitting methods for the linear complementarity problem. SIAM Journal on Matrix Analysis and Applications, 1999, 21: 67–78.
[5] Z.-Z. Bai. Modulus-based matrix splitting iteration methods for linear complementarity problems. Numerical Linear Algebra with Applications, 2010, 17: 917–933.
[6] Z.-Z. Bai, D. J. Evans. Matrix multisplitting relaxation methods for linear complementarity problems. International Journal of Computer Mathematics, 1997, 63: 309–326.
[7] Z.-Z. Bai, D. J. Evans. Matrix multisplitting methods with applications to linear complementarity problems: parallel synchronous and chaotic methods. Calculateurs Paralleles Reseaux et Systems Pepartis, 2001, 13: 125–141.
[8] Z.-Z. Bai, D. J. Evans. Matrix multisplitting methods with applications to linear complementarity problems: parallel asynchronous methods. International Journal of Computer Mathematics, 2002, 79: 205–232.
[9] Z.-Z. Bai, L.-L. Zhang. Modulus-based synchronous multisplitting iteration methods for linear complementarity problems. Numerical Linear Algebra with Applications, 2013, 20: 425–439.
[10] Z.-Z. Bai, L.-L. Zhang. Modulus-based synchronous two-stage multisplitting iteration methods for linear complementarity problems. Numerical Algorithms, 2013, 62: 59–77.
[11] J.-L. Dong, M.-Q. Jiang. A modified modulus method for symmetric positive-definite linear complementarity problems. Numerical Linear Algebra with Applications, 2009, 16: 129–143.
[12] A. Hadjidimos, M. Tzoumas. Nonstationary extrapolated modulus algorithms for the solution of the linear complementarity problem. Linear Algebra and its Applications, 2009, 431: 197–210.
[13] A. Hadjidimos, M. Lapidakis, M. Tzoumas. On iterative solution for linear complementarity problem with an H_+-Matrix. SIAM Journal on Matrix Analysis and Applications, 2012, 33: 97–110.
[14] L. A. Krukier, B. L. Krukier, Z. R. Ren. Generalized skew-Hermitian triangular splitting iteration methods for saddle-point linear systems. Numerical Linear Algebra with Applications, 2014, 21: 152–170.
[15] Y.-T. Li, P.-F. Dai. Generalized AOR methods for linear complementarity problem. Applied Mathematics and Computation, 2007, 188: 7–18.
[16] O. L. Mangasarian. Solution of symmetric linear complementarity problems by iterative methods. Journal of Optimization Theory and Applications, 1977, 22: 465–485.
[17] K. G. Murty, F. T. Yu. Linear Complementarity, Linear and Nonlinear Programming. Berlin: Heldermann, 1988.

[18] J.-S. Pang. Necessary and sufficient conditions for the convergence of iterative methods for the linear complementarity problem. Journal of Optimization Theory and Applications, 1984, 42: 1–17.
[19] U. Sch*ä*fer. A linear complementarity problem with a P-matrix. SIAM Review, 2004, 46: 189–201.
[20] L.-L. Zhang. Two-step modulus-based matrix splitting iteration method for linear complementarity problems. Numerical Algorithms, 2011, 57: 83–99.
[21] L.-L. Zhang. Two-stage multisplitting iteration methods using modulus-based matrix splitting as inner iteration for linear complementarity problems. Journal of Optimization Theory and Applications, 2014, 160: 189–203.
[22] N. Zheng, J.-F. Yin. Accelerated modulus-based matrix splitting iteration methods for linear complementarity problem. Numerical Algorithms, 2013, 64: 245–262.