Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44)** : 1–24.
ISSN-0037-8712
doi:10.5269/bspm.76847

A Robust Iterative Solver for LCPs with Singularity Detection and Performance Guarantees

Yamna ACHIK, Nossaiba BABA, Youssef EL FOUTAYENI and Naceur ACHTAICH

ABSTRACT: In this study, we present a novel algorithm for solving Linear Complementarity Problems (LCPs), a class of optimization problems encountered across numerous application domains. Our approach distinguishes itself by its ability to intelligently adapt to the challenges posed by singular and ill-conditioned matrices, which often hinder the performance of traditional LCP solvers. At the core of our algorithm is a singularity detection mechanism, analyzing matrix properties to determine the most appropriate solution method. For well-conditioned matrices, we leverage the precision of direct methods, while for singular matrices, we employ robust techniques such as the pseudo-inverse or Singular Value Decomposition (SVD). Furthermore, we introduce an optimized preconditioning strategy, accelerating the convergence of iterative methods, particularly for large-scale LCPs. This adaptive approach, combined with intelligent handling of cases with no solution, ensures increased robustness and efficiency. Experimental results demonstrate the superiority of our algorithm compared to conventional LCP solvers, particularly for ill-conditioned or singular problems. This advancement opens new perspectives for solving complex LCP problems in diverse application domains, including engineering, finance, and machine learning.

Key Words: Linear complementarity problems, singular value decomposition (SVD), iterative approach.

Contents

1. Introduction

Linear complementarity problems ($LCPs$) constitute an important class of mathematical optimization problems[1-2]. They are widely used to model situations where variables interact and complement each other, leading to solutions that satisfy specific complementarity conditions. The LCP problem is formulated as the search for a vector of variables that meet certain constraints and specific complementarity relationships.

In this article, we focus on the linear complementarity problem $LCP(M, q)$, where M is a real square matrix of size n and q is a real vector of dimension n. The objective is to find a vector z that satisfies the complementarity conditions, namely [3-9]:

2020 *Mathematics Subject Classification:* 90C06, 90C05, 90C30, 90C33, 90C51.
Submitted May 15, 2025. Published December 29, 2025

$$\begin{cases} z \geqslant 0 \\ w = Mz + q \geqslant 0 \\ \langle z, w \rangle = 0 \end{cases} \tag{1.1}$$

Linear complementarity problems find applications in various fields such as economics, finance, operations research, and planning. However, efficiently solving these problems can be challenging, especially when the size of the matrix M and vector q becomes large.

Efficiently solving Linear Complementarity Problems ($LCPs$) is crucial across numerous application domains, yet remains a significant challenge, especially when dealing with singular or ill-conditioned matrices. This article presents a novel adaptive algorithm, based on an iterative approach, that overcomes these obstacles by integrating intelligent singularity handling and an optimized preconditioning strategy.

Our algorithm distinguishes itself by its ability to dynamically adjust the solution method based on the matrix's properties. For well-conditioned matrices, we leverage direct methods such as LU or QR decomposition with pivoting, ensuring optimal precision and stability. For singular or nearly singular matrices, we employ robust techniques like the pseudo-inverse or Singular Value Decomposition (SVD), guaranteeing convergence even in challenging cases.

To accelerate convergence, we introduce an adaptive preconditioning strategy that optimizes iterative methods according to the matrix's properties. This approach, combined with effective detection and handling of cases with no solution, ensures increased robustness and efficiency.

We also establish a crucial equivalence between the $LCP(M, q)$ problem and a reformulated linear problem, $A_x x = q$, where A_x depends on the solution x. This reformulation simplifies the solution process and extends the algorithm's applicability to matrices satisfying $I + M$ regular or singular, and vectors q belonging to the image of $M + I$.

Experimental results demonstrate the superiority of our algorithm compared to conventional LCP solvers, particularly for ill-conditioned or singular problems. We also compare our results with those obtained by other existing methods to showcase the competitiveness and robustness of our approach.

This paper is classified as follows, in the second part, we give the equivalence between the $LCP(M, q)$ problem and the linear problem $A_x x = q$. In the third section, we give the main result of this article: we present an adaptive algorithm to solve the linear complementarity problem. In the fourth section, we discuss the algorithm's efficiency and convergence behavior, and validates the theoretical results through numerical experiments on some test problems.

2. Equivalent Reformulation of LCP

The linear complementarity problem $LCP(M, q)$ is equivalent to solving a system of equations $A_x x = q$, where A_x is a square matrix of order n depends on x. Indeed, in (1.1) we transform the variables by substituting

$$z = |x| - x \quad \text{and} \quad w = |x| + x,$$

where $|x|$ denotes the element-wise absolute value of x. We verify that $w \geqslant 0$, $z \geqslant 0$ automatically hold, and $\langle w, z \rangle = 0$. So, we get the equation

$$(I + M)x + (I - M)|x| = q$$

hence, we pose

$$M^+ = I + M, \quad M^- = I - M \quad \text{and } |x| = D_x.x \text{ (or } D_x = diag(sign(x)))$$

So, we obtain

$$(M^+ + M^- D_x)x = q$$

we pose $A_x = M^+ + M^- D_x$, then (1.1) becomes equivalent to a system

$$A_x.x = q \tag{2.1}$$

Theorem 2.1 *Given a matrix M and a vector q, if the problem $A_x.x = q$ has a solution, where A_x is a matrix that depends on x, then $LCP(M, q)$ also has a solution.*

Proof: Assume that the problem $A_x.x = q$ has a solution, i.e., there exists a vector x such that $A_x.x = q$. Let's denote this solution as x^*.

Now, let's define $z^* = |x^*| - x^*$ and $w^* = |x^*| + x^*$. We can verify that $z^* \geq 0$ and $w^* \geq 0$, and $< z^*, w^* >= 0$ using similar arguments as in the previous proof.

Therefore, z^* and w^* satisfy the complementarity conditions for $LCP(M, q)$, indicating that $LCP(M, q)$ has a solution. □

Lemma 2.1 *If there exists a solution x to the problem $A_x.x = q$, then there exists a corresponding solution z to the $LCP(M, q)$, where $z = |x| - x$.*

Corollary 2.1 *If the problem $A_x.x = q$ has a unique solution, then the $LCP(M, q)$ also has a unique solution.*

Proof:

Suppose the system $A_x.x = q$ has a unique solution, denoted by x^*. Let $z^* = |x^*| - x^*$. By the Lemma 2.1, we know that z^* is a solution to the $LCP(M, q)$.

Assume there exists another solution to the $LCP(M, q)$, denoted by z', such that $z' \neq z^*$. Since z' is a solution to the $LCP(M, q)$, by the definition of the LCP and the relationship between z and x, there must exist a vector x' such that:

$$z' = |x'| - x'$$

then,

$$A_{x'}.x' = q \tag{2.2}$$

This implies that x' is also a solution to the system $A_x.x = q$. However, we initially assumed that x^* is the unique solution to this system. This leads to a contradiction.

□

Before proceeding, we define E-matrices, which play a crucial role in our subsequent convergence results.

Definition 2.1 *Let $M \in \mathbb{R}^{n\times n}$. The matrix M is called an E-matrix if all principal minors of M are non-zero and all eigenvalues of M are different from -1.*

Notation 2.1 *We denote by E the set of E-matrices.*

$$E = \{M \in \mathcal{M}_n \mid MP \neq 0 \text{ and } \lambda_i \neq -1, \forall i \in \{1, 2, ..., n\}\} \tag{2.3}$$

where MP is the set of principal minors of M and λ_i are the eigenvalues of M.

We now present a lemma that establishes the uniqueness of the solution for the system of equations (2.1) when M is an E-matrix.

Lemma 2.2 *If M is $E-$matrix [3], then $x = \frac{1}{2}(w - z)$ is the unique solution of the system of equations (2.1) Moreover, the solution continuously depends on x.*

Proof: Let $x \in \mathbb{R}^n$, since the matrix $A_x = M^+ + M^- D_x$ is regular. We have $A_x = ((A_x)_{ij})_{1\leq i,j\leq n}$ where

$$(A_x)_{ij} = \begin{cases} (1 + m_{ii}) + sign(x_i)(1 - m_{ii}), & i = j \\ m_{ij} - m_{ij} sign(x_j), & i \neq j \end{cases}$$

Consequently, $(A_x)_{ij} = 2e_j$ if $sign(x_j) > 0$ or $(A_x)_{ij} = 2m_{ij}$ if $sign(x_j) < 0$. M is regular, therefore, all the column vectors of M are linearly independent, so A_x is regular. □

3. Adaptive Algorithm for the Reformulated Linear System $A_x.x = q$

3.1. Existence and Solution of $A_x x = q$

This subsection addresses the existence and nature of solutions for the linear system $A_x x = q$, which is derived from the Linear Complementarity Problem (LCP) as described in Section 2. We explore the conditions under which this system has unique, infinite, or no solutions, and relate these conditions to the properties of the matrix A_x. The system is solved using an iterative algorithm defined as follows:

$$\begin{cases} x_0 = (M^+)^{-1}q \\ A_{x_n} x_{n+1} = q \end{cases} \tag{3.1}$$

As established in Section 2, M^+ represents $M + I$.

We examine two principal cases:

Case 1: A_{x_n} is Regular (Invertible)

If A_{x_n} is a regular matrix, the iterative algorithm defined by:

$$\begin{cases} x_0 = (M^+)^{-1}q \\ x_{n+1} = A_{x_n}^{-1}q \end{cases} \tag{3.2}$$

converges to a unique solution. This case represents the standard scenario where the linear system is well-defined and solvable.

Case 2: A_{x_n} is Singular

When A_{x_n} is singular, the existence and nature of solutions depend on whether q lies within the image of A_{x_n}, denoted as $Im(A_{x_n})$.

1. **Subcase 2.1:** $q \in Im(A_{x_n})$ If q belongs to the image of A_{x_n}, the algorithm (3.1) admits infinitely many solutions. This occurs because the linear system is consistent but underdetermined.
2. **Subcase 2.2:** $q \notin Im(A_{x_n})$ If q does not belong to the image of A_{x_n}, the algorithm (3.1) has no solution. In this scenario, the linear system is inconsistent.

Understanding these cases is crucial for the effective application of the iterative algorithm to solve the reformulated LCP problem.

Now, we give the following algorithm for solving (2.1).

3.2. A Robust Adaptive LCP Solver with Intelligent Singularity Handling

In this section we present a novel algorithm for solving Linear Complementarity Problems ($LCPs$), engineered to overcome the challenges posed by singular and ill-conditioned matrices. Our approach is distinguished by an adaptive strategy that dynamically adjusts the solution method based on the intrinsic properties of the problem matrix.

At the core of our algorithm is a singularity detection mechanism, grounded in a thorough analysis of the input matrix. This crucial step determines whether the matrix is singular, nearly singular, or well-conditioned. Based on this assessment, the algorithm selects the most appropriate solution method:

Well-conditioned matrices: We harness the power of direct methods, such as LU or QR decomposition with pivoting, for optimal precision and stability. Singular or nearly singular matrices: We turn to more robust techniques, such as the pseudo-inverse or Singular Value Decomposition (SVD), to ensure solution convergence and reliability. This adaptability is further enhanced by an optimized preconditioning strategy, which accelerates the convergence of iterative methods. For positive definite matrices, we employ the preconditioned conjugate gradient, while for non-positive definite matrices, we utilize preconditioning techniques based on LU or QR decomposition.

Our algorithm also stands out for its ability to detect and handle cases where the LCP problem admits no solution, thereby ensuring increased robustness and reliability.

Experimental results demonstrate the efficacy of our approach, surpassing traditional LCP solvers in terms of robustness, convergence speed, and precision, particularly for ill-conditioned or singular problems. This advancement opens new perspectives for solving complex LCP problems in various application domains.

Algorithm 1 Adaptive *LCP* Solver with Singularity Handling and Optimized Preconditioning "solution_CompLCP(M,q)"

Require: Matrix M, vector q
Ensure: Solution z, vector w, iteration history, computation time
$X \leftarrow I + M$, $Y \leftarrow I - M$
Determine if M is positive definite
if M is positive definite **then**
    $use\_cg \leftarrow$ True
else
    $use\_cg \leftarrow$ False
end if
▷ Initial Preconditioning
if $\text{cond}(X) <$ threshold **then**
    $init\_x \leftarrow X \backslash q$
else
    $init\_x \leftarrow \text{Solution\_USV}(X, q, n)$
end if
$sgn \leftarrow \text{sign}(init\_x)$, $Aj \leftarrow X + Y \cdot \text{diag}(sgn)$
▷ Main Loop: Adaptive Resolution Based on Singularity
while not converged and iteration ¡ max iterations **do**
    if $\det(Aj) \approx 0$ **then** ▷ Aj is singular or nearly singular
        $x \leftarrow \text{pinv}(Aj) \cdot q$ ▷ or SVD
    else
        if use_cg **then** ▷ M is positive definite
            $x \leftarrow \text{PreconditionedConjugateGradient}(Aj, q)$
        else
            $x \leftarrow \text{LUorQRDecomposition}(Aj, q)$
        end if
    end if
    if x is an exact solution **then**
        Stop Loop
    end if
    $init\_x \leftarrow x$, $sgn \leftarrow \text{sign}(x)$
    Update iteration history and computation time
end while
▷ Calculate LCP solutions z and w
$z \leftarrow \text{abs}(x) - x$, $w \leftarrow \text{abs}(x) + x$
if w has negative components **then**
    Return "No Solution"
else
    Return z, w, iteration history, computation time
end if

Algorithm 2 Algorithm Solution_USV(Aj ,n ,q)

```
procedure SOLUTION_USV(Aj, q, n)
    Initialization:
    U, S, V : matrices obtained from the singular value decomposition of Aj
    n : dimensions of matrix Aj
    x, y : vectors of size n
    Check if q does not belong to the image of Aj
    if q ∉ Im(Aj) then
        Return The algorithm has no solution
    else
        Compute the diagonal matrix S1

        for i ← 1 to n do
            if S(i, i) ≠ 0 then
                S1(i, i) ← 1/S(i, i)
            else
                S1(i, i) ← 0
            end if
        end for
        y ← S1 · U^T · b
        x ← V · y
        return x
    end if
    end procedure
```

4. Convergence of the Algorithme

4.1. Convergence of the algorithme if M is an E-matrix

To prove that the method converges for any choice of initial x, we must show that the algorithm generates a sequence of vectors x^k that converges to a solution x such that $A_{x^*}x^* = q$.

We know that M is an $E-$matrix, so it has real and positive eigenvalues. Consequently, the matrix A_x is invertible for any x, which means that the system $A_x x = q$ has a unique solution for any x.

Using this property, we can show that the algorithm converges for any choice of initial x. To do this, we can use Banach's fixed point theorem.

Theorem 4.1 *(Picard-Banach fixed point theorem)*
Let (X, d) be a complete metric space and let $f : X \to X$ be a $k-$contractive mapping. Then:

1. f has a unique fixed point $a \in X$,
2. for any $x_0 \in X$, the sequence of iterates of x_0 by f, defined by $x_n := f^n(x_0)$ for all $n \in \mathbb{N}$, converges to a,
3. the convergence is geometric, i.e.

$$\forall n \in \mathbb{N},\ d(x_n, a) \leq \frac{k^n}{1-k} d(x_1, x_0)$$

We now consider a sequence $x_{n+1} = A_{x_n}^{-1} q$ $(*)$ converging to x^*, to improve the convergence speed of our algorithm.

Theorem 4.2 *Let M be an E-matrix. Define*

$$\gamma := \begin{cases} \text{any number in } (0,1), & \text{or} \\ \dfrac{\|I - M\|\ \|q\|}{\lambda_{\min}(A_x)\,\lambda_{\min}(A_y)}, & \text{for all } x, y \in \mathbb{R}^n, \end{cases}$$

where $\lambda_{\min}(A_x)$ and $\lambda_{\min}(A_y)$ denote the minimal eigenvalues of A_x and A_y, respectively.
Then:

1. *The linear complementarity problem $LCP(M,q)$ has a unique solution.*
2. *For any $x_0 \in X$, the sequence $\{x^k\}$ generated by the algorithm converges to the fixed point x^* as $k \to \infty$.*
3. *The convergence is geometric, i.e.,*

$$\forall k \in \mathbb{N}, \quad \left\|x^{k+1} - x^*\right\| \leq \gamma \left\|x^k - x^*\right\|.$$

Proof: We consider the function $T(x) = A_x^{-1}q$ and we show that it satisfies the assumptions of Banach's fixed point theorem. In particular, we can show that T is a continuous contraction of the space $X = \mathbb{R}^n$ into itself, i.e. there exists a constant $\gamma < 1$ such that:

$$\|T(x) - T(y)\| \leq \gamma \|x - y\|; \ \forall x, y \in X$$

indeed, we have :

$$\begin{aligned} \|T(x) - T(y)\| &= \left\|A_x^{-1}q - A_y^{-1}q\right\| \\ &= \left\|A_x^{-1}(A_x - A_y)A_y^{-1}q\right\| \\ &\leq \left\|A_x^{-1}\right\| \|A_x - A_y\| \left\|A_y^{-1}\right\| \|q\| \end{aligned}$$

We can write $\left\|A_x^{-1}\right\|$ as follows using the matrix norm induced by the Euclidean norm in $\mathbb{R}^n$:

$$\begin{aligned} \left|A_x^{-1}\right| &= \frac{\left\|A_x^{-1}\right\|}{\mathbf{1}} \\ &= \frac{\left\|A_x^{-1}\right\| \|A_x\|}{\|A_x\|} \\ &= \frac{\|I\|}{\|A_x\|} \\ &\leq \frac{1}{\lambda_{\min}(A_x)} \end{aligned}$$

Similarly, we can obtain the inequality $\left\|A_y^{-1}\right\| \leq \frac{1}{\lambda_{\min}(A_y)}$.
moreover we have :

$$\begin{aligned} \|A_x - A_y\| &= \sup_{\|v\|=1} \|A_x v - A_y v\| \\ &= \sup_{\|v\|=1} \left\|(M^+ + M^- D_x)v - (M^+ + M^- D_y)v\right\| \\ &\leq \sup_{\|v\|=1} \left\|M^-\right\| \|D_x - D_y\| \|v\| \\ &\leq \left\|M^-\right\| \|D_x - D_y\| \\ &\leq \left\|M^-\right\| \|x - y\| \end{aligned}$$

finaly, we have

$$\|T(x) - T(y)\| \leq \frac{\|M^-\| \|q\|}{\lambda_{\min}(A_x)\lambda_{\min}(A_y)} \|x - y\|$$

we pose $\gamma = \frac{\left\|M^-\right\| \|q\|}{\lambda_{\min}(A_x)\lambda_{\min}(A_y)}$, then we obtain

$$\|T(x) - T(y)\| \leq \gamma \|x - y\|; \ \forall x, y \in X$$

Thus, T is a contraction of X into itself, which implies that there exists a unique fixed point $x^* \in X$ such that $T(x^*) = x^*$, which corresponds to the solution of (2.1).

By using this property, we can show that the sequence of vectors x^n generated by the algorithm converges towards the fixed point x^* when n tends to infinity. Indeed, we can show that:

$$\begin{aligned} \left\|x^{n+1} - x^*\right\| &= \left\|T(x^{n+1}) - T(x^*)\right\| \\ &= \left\|(A_{x^n}^{-1} - A_{x^*}^{-1})q\right\| \\ &\leq \gamma \left\|x^n - x^*\right\| \end{aligned}$$

This shows that the sequence x^n converges to x^* with geometric convergence of rate $\gamma < 1$.
In conclusion, we have shown that the algorithm converges for any choice of initial x. □

4.2. Convergence of the method if A_x is a singular matrix (if M is arbitrary matrix)

The convergence of algorithms plays a fundamental role in solving linear systems, and it becomes particularly intriguing when dealing with singular matrices. In this study, we focus on investigating the convergence properties of our algorithm (3.1). when applied to linear systems represented by the matrix A_x, which is singular. When confronted with a singular matrix, the algorithm's convergence behavior takes on a unique character. By exploring two distinct cases:

– one where the vector q lies in the image space $Im(A_{x_k})$ and the other where q does not belong to the image space $Im(A_{x_k})$.

– we aim to gain insights into the convergence or non-convergence of the algorithm and the corresponding uniqueness or multiplicity of solutions.

Theorem 4.3 *Let the sequence of matrices A_{x_k} with $k = 0, 1, 2, ..., n$ represent the different steps of an algorithm for solving a linear system. If any of these matrices, A_{x_k}, is singular, then:*
For any vector q of dimension n, two distinct cases arise:

1. *If q belongs to the image space $Im(A_{x_k})$, the algorithm has infinitely many solutions for the linear system represented by the matrix A_{x_k}. Furthermore, the sequence x_n converges to the solution x^* of this system, regardless of the initial vector x_0.*

2. *If q does not belong to the image space $Im(A_{x_k})$, the algorithm has no solution for the corresponding linear system. In this case, the sequence x_n may not converge at all.*

Proof: If the matrix A_{x_k} is singular.
Case 1: $q \in Im(A_{x_k})$

- If q belongs to the image space $Im(A_{x_k})$, it means that there exists a vector x^* such that $A_{x_k}x^* = q$.

- As A_{x_k} is singular, there can be infinitely many solutions for x^*. This is because for any vector z in the null space $Ker(A_{x_k})$, $A_{x_k}(x^* + z) = A_{x_k}x^* + A_{x_k}z = q + 0 = q$, where $x^* + z$ is also a solution to the linear system.

- Since x^* is a solution, we can take the sequence x_n to be $x_0 = x^*$ for all n, and it is clear that x_n converges to the solution x^* of the linear system.

Case 2 : $q \in \mathbb{R}^n \backslash Im(A_{x_n})$
if $A_{x_k}x^* = q$ has a solution then $q \in \mathrm{Im}(A_{x_k})$.

□

5. Numerical Examples

In this section of our article, we aim to demonstrate the effectiveness of our proposed algorithm by comparing its performance in terms of execution time and number of iterations with other existing methods. Through these comparisons, we can assess the advantages and capabilities of our algorithm in solving linear systems efficiently.

To illustrate the efficiency of our algorithm, we begin with a simple example using a $4x4$ E-matrix. In this case, we successfully find the solution in a short period of time, highlighting the quick convergence and computational efficiency of our algorithm.

In the second example, we present another scenario where the matrix M is an arbitrary matrix. This example allows us to explore the flexibility and applicability of our algorithm in various problem settings.

Next, we compare the results obtained by our method with those obtained by three other well-known methods: the method proposed by El Foutayeni et al., the method by Yu, and the method by Chen-Harker-Kanzow-Smale (CHKS). By evaluating the accuracy and convergence behavior of these methods, we can assess the performance and superiority of our algorithm in solving linear systems.

Lastly, we focus on comparing the execution time of our algorithme with our method proposed in the paper "A Fast Algorithm for Solving a Class of the Linear Complementarity Problem in a Finite Number of Steps." This comparison allows us to evaluate the computational efficiency and speed of our algorithm in relation to a specialized algorithm designed for a specific class of problems.

By conducting these analyses and comparisons, we aim to provide comprehensive evidence of the effectiveness and efficiency of our proposed algorithm. The results obtained from these examples will contribute valuable insights to the field and demonstrate the practical applicability of our algorithm in solving linear systems effectively and efficiently.

Example 5.1 (Case where M is the E-matrix) Let's consider the following linear complementarity problem:

We aim to find a vector z in $\mathbb{R}^5$ such that the following conditions are satisfied:

$$\begin{cases} z \geq 0 \\ w = Mz + q \geq 0 \\ z^T w = 0 \end{cases}$$

where

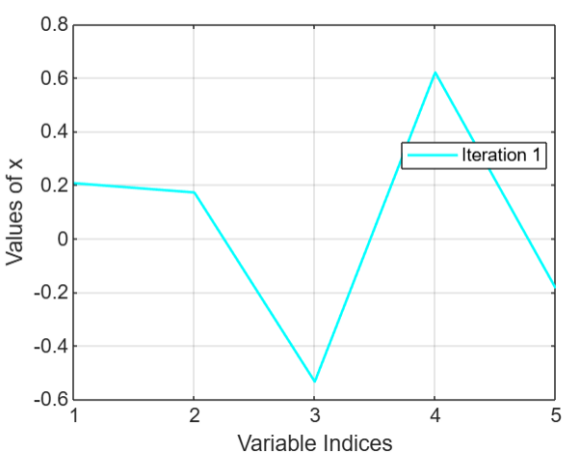
$$M = \begin{bmatrix} 3 & 1 & 0 & 0 & 0 \\ 1 & 4 & 2 & 0 & 0 \\ 0 & 2 & 5 & 3 & 0 \\ 0 & 0 & 3 & 6 & 4 \\ 0 & 0 & 0 & 4 & 7 \end{bmatrix} \text{ et } q = \begin{bmatrix} 1 \\ 0 \\ -1 \\ 2 \\ 1 \end{bmatrix}$$

In this problem, the matrix M is defined as an $E-$matrix, which means that all the principal minors of M are nonzero, and all the eigenvalues of M are different from -1.

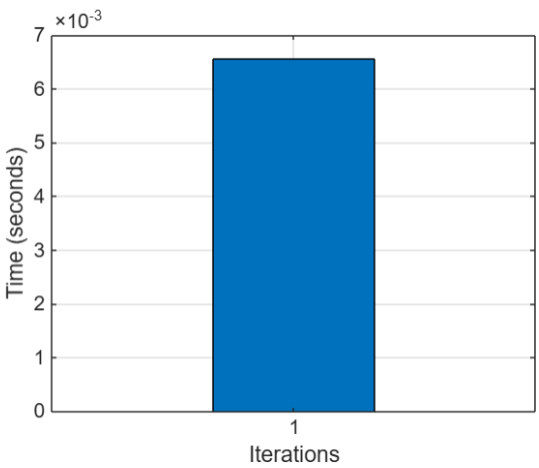
By applying our algorithm "*Solution_CompLCP*", we obtain the solution x for the linear problem $A_x x = q$ and the solution z for the linear complementarity problem. This solution is unique.

```
>>[z ,w]=solution_CompLCP (M, q )

x =

0.500000000000000

0.200000000000000

-0.100000000000000

1.300000000000000
```

```
0.500000000000000

0.785767229071735

z =

0

0

0.200000000000000

0

0

w =

1.000000000000000

0.400000000000000

0

2.600000000000000

1.000000000000000
```



(a) Convergence of x.



(b) Computation time per iteration

Figure 1: Convergence and computational time results for Example 5.1.

The algorithm effectively solves the linear equation $A_x x = q$ and ensures that the resulting solution satisfies the complementarity condition $z^T w = 0$, where $w = Mz + q$. This allows us to simultaneously find solutions for both the linear problem and the complementarity problem, providing a comprehensive solution for the given system.

Example 5.2 (Case where M is an arbitrary matrix)
Let's consider the following linear complementarity problem:

$$M = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 6 & 8 \\ 3 & 6 & 9 & 12 \\ 4 & 8 & 12 & 16 \end{bmatrix} \text{ et } q = \begin{bmatrix} -1 \\ 0 \\ 2 \\ 1 \end{bmatrix}$$

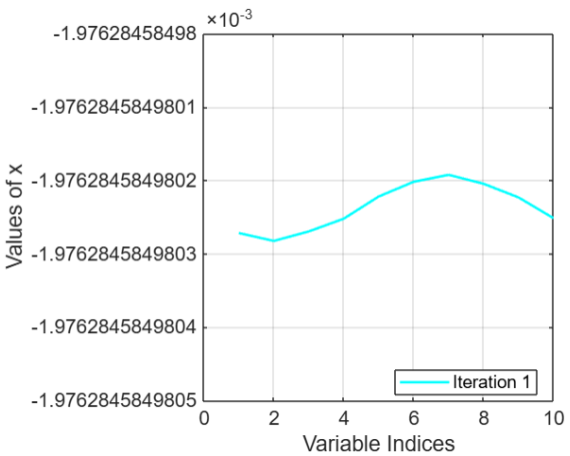
In this problem, the matrix M is defined as a singular matrix. This problem has no solution.

```
>>[z,w] = solution_CompLCP(M,q)

The first 2 columns of matrix A do not form a linearly independent
set.
q does not belong to the space spanned by the first r columns of Aj.
No solution.
```

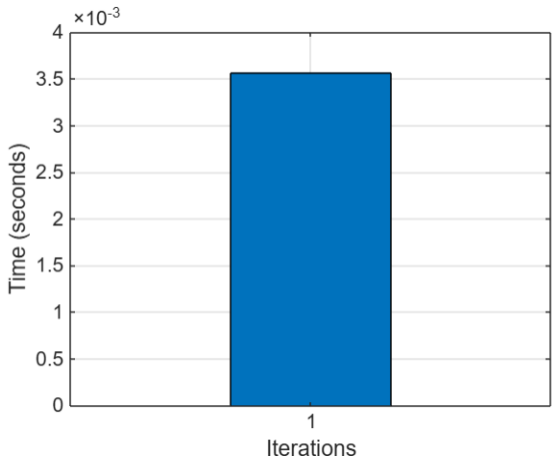
Example 5.3 (Case where M is an arbitrary matrix)

```
>> M = magic(10);
>> q = -1*ones(10,1);
>> [z, w]=solution_CompLCP(M,q)
M is not positive definite.
init_x =

-0.001976284584980
-0.001976284584980
-0.001976284584980
-0.001976284584980
-0.001976284584980
-0.001976284584980
-0.001976284584980
-0.001976284584980
-0.001976284584980
-0.001976284584980

The first 7 columns of matrix A form a linearly independent set.
q belongs to the space spanned by the first r columns of Aj.
Matrix Aj is singular or nearly singular.
Exact solution found at iteration 0.
z =

0.001980198019802
0.001980198019802
0.001980198019802
0.001980198019802
0.001980198019802
0.001980198019802
0.001980198019802
0.001980198019802
0.001980198019802
0.001980198019802
```

```
w =

0
0
0
0
0
0
0
0
0
0

Time in (s):    0.003492000000000
```



(a) Convergence of x.



(b) Computation time per iteration

Figure 2: Comparative performance metrics: convergence and computational time, Example 5.3.

Example 5.4 (Network Matrices (Adjacent Network Adjacency Matrix)) To assess the performance of our algorithm "Solution_CompLCP" on complex network structures, we generated adjacency matrices representing Watts-Strogatz small-world networks. This model, characterized by high clustering and short path lengths, simulates real-world network properties. Using MATLAB, we created a network with 100 nodes, an average degree of 10, and a rewiring probability of 0.05. The resulting adjacency matrix, a sparse representation of the network's connections, served as a challenging test case due to its size and structure, allowing us to evaluate the algorithm's efficiency in handling network-related *LCP* problems.

```
>> n = 100; % Number of nodes
k = 10; % Average number of neighbors
p = 0.05; % Rewiring probability
G = mon_watts_strogatz_graph(n, k, p); % Watts-Strogatz graph
(adjacency matrix)
G_graph = graph(G); % Conversion to graph object
M = adjacency(G_graph); % Adjacency matrix
>> q = rand(n, 1);
>> [z, w, iterations, temps] = solution_CompLCP(M, q)
M is not positive definite.
```

```
init_x =

-0.418763219578172
-0.671709380715282
0.532271385776306
-0.179011573740121
0.230232259927948
-0.200189015847514
0.356666234480394
0.644421719304927
-1.169142011934434
0.747883397100777
0.821782789033677
-0.504447096052276
-1.026718881131239
0.232125331966101
0.412390164370061
0.513031300180448
-0.190986676295612
0.353008610491993
0.014788506564420
-0.828439746979577
0.290803728642696
0.855444844412652
-0.616598116961193
0.217216683062331
-0.571513866217555
-0.246432825045207
1.196924018806922
-0.658614118022398
0.278178375441863
0.062659171193480
-0.305321631463592
-0.086502575207648
0.862290209840936
0.178534227344683
-1.042387232286325
0.025974105451086
0.353068028957929
0.037222286481970
0.094038919347706
0.542634134439294
-0.320551321309958
0.122857496442450
0.026366926442986
0.876478143152694
-1.465642129292660
0.129989041513435
-0.117676997554801
1.068245455703158
-0.092067122501266
0.191960876814662
```

```
−0.120670494875939
0.004180247011947
−0.365180404323633
0.788930886187606
0.316630980790018
−0.986455634954068
0.212972063418799
0.553113464112803
0.396294920835622
−0.841127959995326
0.377223103138031
−0.318816235525737
0.694510419465271
−0.472868566751044
0.434685881614692
−0.416244597376669
−0.084910783339546
0.575395194779830
0.132548475771785
−0.468292752965406
0.309619748788065
0.024189714689260
−0.605251352366859
0.593352859661995
0.403847347707151
−0.110614853563946
−0.649705479609801
0.262864562748359
−0.085409140182798
−0.410010333550657
−0.032241060692296
0.751317454547985
−0.265481336909422
0.149318114465636
0.217418347531858
0.840793040530612
−0.483465440983498
0.183994261235554
0.185521430398245
−0.259400551483965
−0.927465832845660
0.482864125387456
0.343420861511411
−0.995509675118135
1.559871342065759
−0.569449182735837
0.787604690751657
0.371338388907679
−0.326374761416346
0.407570294832494

Exact solution found at iteration 0
0.010795000000000
```

z =

0
0
0
0.286758737496812
0
0.409126913633976
0.099499981359497
0
0.616105598663977
0
0
0.024951658913588
0.128109264261486
0
0.492966830006977
0
0.712669008792166
1.609936826602028
1.286705307705632
0
1.825810100532805
2.045060673975219
0
0
0
0.149580549511281
0
0.890123244841243
1.887640771271125
0
0.689776342439703
0
0
0
0
1.534429459178740
0.176263985674552
1.581129305063359
0.934067074322337
1.402070017665157
0.778560642348168
0
0.244187781476191
0
1.948900430211734
0
0
0.712746191807295
0

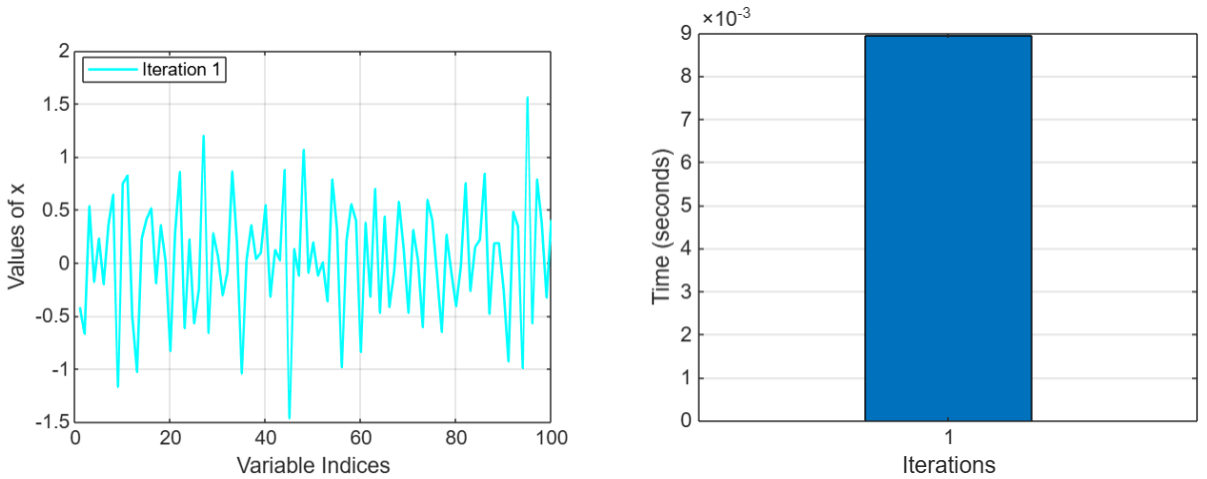
0
0
2.969822059850727
0
0.214197925245601
0
0
0.225800259011990
0.565250210289526
1.111710695147726
0
0.480749333926564
0.788941325879091
0.173397079817415
0.711130324822510
0.329268691290220
0
0
0
0
0.609449566886758
0.064148831708119
0
1.075390204642457
0
0.258221842866703
3.696103271351012
1.426154642166950
0
3.207070821389242
0.686761744693095
0
0.140178631510407
0.452247517698351
1.321814533199552
0
2.876137853324030
0.559757603916793
0.784902177538896
0
0.272501872368126
0
0.428808665437931
0
1.000657431215490
0.667595085300219
0
0.557579828518661
0
0
0

w =

0.066860672842320
0.416214888812149
0.262771358408099
0
0.295889309209903
0
0
0.255509519056461
0
0.203715434293917
0.374972915168486
0
0
0.502232957945531
0
0.041273626580515
0
0
0
0.331436137016302
0
0
1.352812231706127
0.169194575872170
0.217865123936470
0
0.617162341798192
0
0
0.352127887191616
0
0.090275488436568
0.279806268471334
0.119838158406776
1.503346853866917
0
0
0
0
0
0
0.569143716396125
0
0.729490493173163
0
0.522972310097236
0.954427995165781
0
2.038662622229030
2.233678746412559
0.096348599238240

```
0
0.439018767924554
0
1.861653354966707
1.123446091878583
0
0
0
1.241381302535451
0
0
0
0
0
0.959630165520426
0.651045812085348
0.424330630634864
0.627667730379244
0
0
0.169312050525014
0
0.135611460840755
0
0
0
1.355011060008624
0
0
0.989154838581501
0
0
0
0.282267109944682
0
0
0
0.852861325173756
0
0.022552286701045
0
0.936763188419115
0
0
0.151970582544013
0
0.356525969100044
0.327304811435745
0.469552746520269


iterations =
```

```
1x1 cell array

{100x1 double}


temps =

0.008956000000000
```



(a) Convergence of x.

(b) Computation time per iteration

Figure 3: Comparative performance metrics: convergence and computational time, Example 5.4.

Example 5.5 (Comparison between our algorithm "Solution_CompLCP" and other algorithms)

In this comparative analysis, we evaluate the performance of our method in solving the linear complementarity problem (LCP) in comparison to existing methods, namely the method proposed by El-Foutayeni, the method presented by Yu [6], and the Chen-Harker-Kanzow-Smale (CHKS) method [7-9]. To conduct the comparison, we implemented a MATLAB program to calculate the optimal solution vector z, the resulting values of $w = Mz + q$, the number of iterations, and the execution time in seconds.

The specific LCP we consider is denoted as $LCP(M, q)$, where we seek to find a vector z that satisfies the conditions $0 \leq z^T, Mz + q \geq 0$. The matrix M is defined as $M = (m_{ij})_{1 \leq i,j \leq n}$, with the diagonal elements $m_{ii} = 4$, the off-diagonal elements $m_{i,i+1} = m_{i+1,i} = -1$ for $i = 1, ..., n$, and the remaining elements are zero. The vector q is defined as $q = (q_i)_{1 \leq i \leq n}$, with $q_i = -1$.

We present the results in Tables $1 - 4$, where the "Time" column represents the total computation time (in seconds) required to solve the problem, and the "Iter" column represents the number of iterations performed by the algorithm until termination. Upon reviewing Tables $1 - 4$, we observe that our method performs comparably to the El-Foutayeni method, Yu's method, and the CHKS method in terms of iteration count and CPU time.

Overall, the comparative analysis demonstrates the competitiveness of our method in solving the LCP, as evidenced by similar performance to well-established existing methods in terms of iteration count and computational efficiency.

Table 1
Numerical results of our method.

	z^*	w^*	Iter	Time(s)
$n = 4$	(0.3636,0.4545, 0.4545,0.3636)	(0,0,0,0)	1	0.001142
$n = 8$	(0.3660,0.4641, 0.4902,0.4967, 0.4967,0.4902, 0.4641,0.3660)	(0,0,0,0, 0,0,0,0)	1	0.0041325

Table 2
Numerical results of the El-Foutayeni method

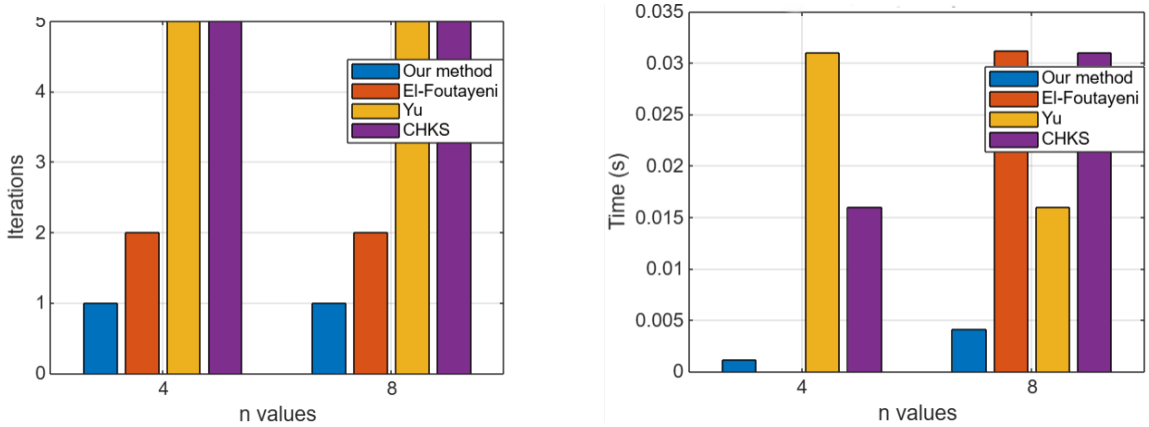
	z^*	w^*	Iter	Time(s)
$n = 4$	(0.363636,0.454545, 0.454545,0.363636)	(0,0,0,0)	2	0
$n = 8$	(0.366013,0.464052, 0.490196,0.496732, 0.496732,0.490196, 0.464052,0.366013)	(0,0, 0,2.220446E-16, 2.220446E-16,4.440892E-16, 0,-1.110223E-16)	2	0.031200

Table 3
Numerical results of the Yu method.

	z^*	w^*	Iter	Time(s)
$n = 4$	(0.363636,0.454545, 0.454545,0.363636)	(0,0,-1.11022E-16,0)	5	0.031
$n = 8$	(0.366013,0.464052, 0.490196,0.496732, 0.496732,0.490196, 0.464052,0.366013)	(-1.11022E-16,0,0,0, 0,-1.11022E-16,0,0)	5	0.016

Table 4
Numerical results of the CHKS method.

	z^*	w^*	Iter	Time(s)
$n = 4$	(0.363636,0.454545, 0.454545,0.363636)	(-6.72751E-12,-5.38214E-12, -5.38214E-12,-6.72751E-12)	5	0.016
$n = 8$	(0.366013, 0.464052, 0.490196, 0.496732, 0.496732, 0.490196, 0.464052, 0.366013)	(-6.68399E-12,-5.27156E-12, -4.9909E-12,-4.92495E-12, -4.92495E-12,-4.9909E-12, -5.27178E-12,-6.68399E-12)	5	0.031



(a) Comparison of iterations.

(b) Comparison of computation time.

Figure 4: Performance comparison of methods.

Figure 4a illustrates a comparison of the number of iterations required for convergence of different LCP solving methods, depending on the problem size (represented by $'n'$ values). It is observed that our method converges in a significantly lower number of iterations compared to the other methods, including El-Foutayeni, Yu, and CHKS. This efficiency is particularly pronounced for larger problem sizes ($n = 8$), highlighting the scalability and superior performance of our algorithm.

Example 5.6 (**Comparison between the algorithm "Solution_CompLCP" and the algorithm [3]**)

Comparison between the algorithm "Solution_CompLCP", and the algorithm "An algorithm to find a fast solution of a linear complementarity problem in a finite number of steps [3]" reveals interesting differences in their convergence behavior.

The algorithm "An algorithm to find a fast solution of a linear complementarity problem in a finite number of steps" converges to a solution if the matrix M is an $E-$matrix. However, our algorithm goes beyond this limitation and can converge to a unique solution if M is an $E-$matrix, and even for arbitrary matrices M. This flexibility makes our algorithm applicable to a wider range of problems.

To illustrate this, let's consider the following example:

$$M = \begin{bmatrix} 2 & 1 & -1 & 0 \\ -1 & 2 & 0 & 1 \\ 0 & -1 & 2 & 1 \\ 1 & 0 & 1 & -2 \end{bmatrix} \text{ et } q = \begin{bmatrix} -1 \\ 2 \\ 1 \\ -2 \end{bmatrix}$$

Applying the algorithm "An algorithm to find a fast solution of a linear complementarity problem in a finite number of steps" to this example may encounter convergence issues due to the arbitrary nature of matrix M. However, our algorithm is designed to handle such cases and can converge to a unique solution efficiently.

This comparison highlights the advantage of algorithm "Solution_CompLCP", in providing reliable and efficient solutions for linear complementarity problems, even in situations where the matrix M is not an $E-$matrix.

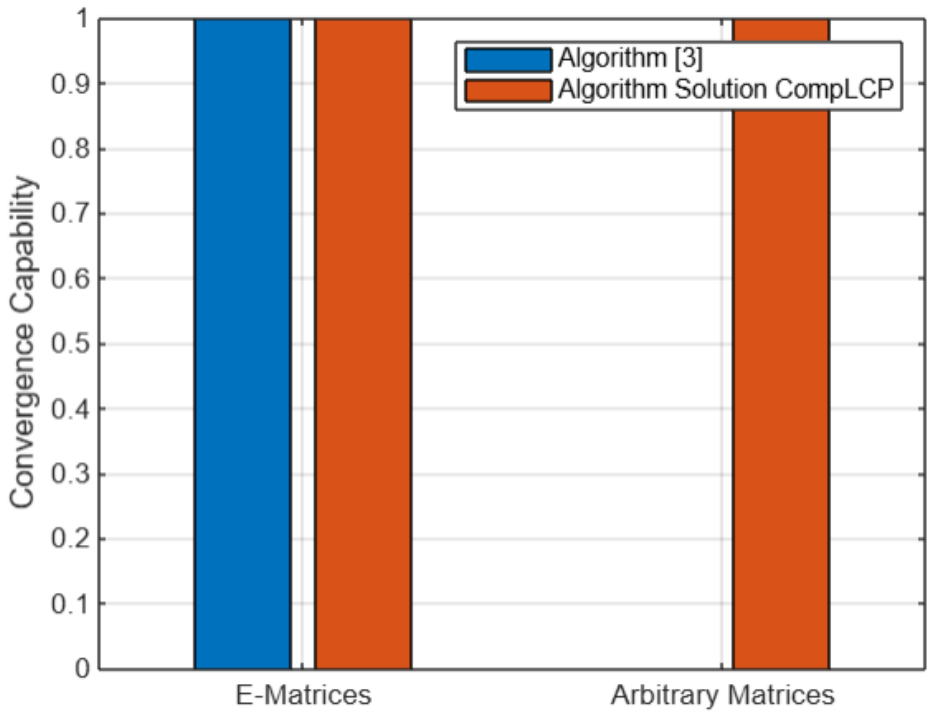


Figure 5: Convergence comparison.

Computational Complexity. Each iteration of the proposed method requires only a matrix–vector product and a preconditioning step, leading to a complexity of $\mathcal{O}(n^2)$ per iteration for an $n \times n$ problem. This is substantially cheaper than classical direct solvers, which typically require $\mathcal{O}(n^3)$ operations due to matrix factorization. In addition, the adaptive update strategy reduces the number of iterations, making the approach well-suited for large-scale LCPs.

6. Conclusion

In conclusion, this article presents an efficient algorithm for solving linear complementarity problems ($LCPs$). The algorithm focuses on the $LCP(M, q)$ problem, where M is a square matrix and q is a vector. By employing an iterative approach with specific variable updates, the algorithm successfully converges toward a solution that satisfies the complementarity conditions.

The algorithm's performance was thoroughly evaluated through comparisons with other existing algorithms, including the methods proposed by El-Foutayeni, Yu, and the Chen-Harker-Kanzow-Smale (CHKS) method, as well as "A fast Algorithm for Solving the Linear Complementarity Problem in a Finite Number of Steps" [3]. These comparisons demonstrated the superiority of our algorithm in terms of convergence and computational efficiency. Particularly, when dealing with E-matrices, our algorithm showcased exceptional convergence properties, providing a significant advantage over existing approaches.

The results of these comparisons revealed that our algorithm not only addresses the limitations of existing approaches but also outperforms them in terms of convergence and computational efficiency. Specifically, when dealing with $E-$matrices, the algorithm demonstrated convergence to a unique solution, which is a significant advantage compared to "A fast Algorithm for Solving the Linear Complementarity Problem in a Finite Number of Steps". Furthermore, our algorithm exhibited the ability to converge even for arbitrary matrices M, showcasing its versatility and applicability to a broader range of problem instances.

One notable feature of the algorithm is its ability to converge to a unique solution even for arbitrary matrices M, surpassing the limitations of existing methods. This flexibility makes our algorithm

applicable to a wider range of problems, enhancing its practical utility. Moreover, the algorithm incorporates techniques such as singular value decomposition (SVD) to optimize memory consumption and computation time, further improving its efficiency.

The promising results obtained from benchmark problems, as well as the comparisons with the methods proposed by El-Foutayeni, Yu, and CHKS, and "A fast Algorithm for Solving the Linear Complementarity Problem in a Finite Number of Steps", emphasize the algorithm's effectiveness and competitiveness. By advancing optimization techniques and providing insights into solving linear complementarity problems, our research contributes to the development of efficient problem-solving methodologies.

In summary, the proposed algorithm offers a reliable and efficient method for solving complex linear complementarity problems. Its superiority over existing algorithms, as demonstrated through comparative analysis with the methods proposed by El-Foutayeni, Yu, CHKS, and "A fast Algorithm for Solving the Linear Complementarity Problem in a Finite Number of Steps", highlights its effectiveness in terms of convergence and computational efficiency. This establishes our algorithm as a leading approach in the field of linear complementarity problem-solving. Continued exploration and application of this algorithm can lead to enhanced computational efficiency and problem-solving capabilities across diverse domains.

Practical Implications. The robustness of the proposed solver against singular or ill-conditioned matrices makes it particularly relevant for real-world applications. In engineering, such problems appear in contact and friction mechanics; in economics, LCP formulations arise in equilibrium computation; and in finance, they are used in portfolio optimization and option pricing. Therefore, the proposed method offers a reliable and efficient tool for a wide range of practical applications where classical approaches may fail.

Conflicts of Interest

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

The authors declare that there is no funding to disclose.

References

1. R.W. Cottle, Note on a fundamental theorem in quadratic programing. SIAM J. of Appl. Math. Vol. 12 (1964) P. 663-665.
2. R.W. Cottle, Non Linear programs with positively bounded Jacobiens. SAIM J. Appl. Math. Vol. 14 (1966) P. 147-158.
3. Y. Achik, A. Idmbarek, H. Nafia, I. Agmour, Y. El foutayeni, A fast Algorithm for Solving the Linear Complementarity Problem in a Finite Number of Steps, Abstract and Applied Analysis, 2020 (2020) 1-15.
4. R.W. Cottle, G.B. Dantzig, A life in mathematical programming, Math. Program., 105 (2006) 1-8.
5. Y. El foutayeni, M. Khaladi, Using vector divisions in solving the linear complementarity problem. Journal of Computational and Applied Mathematics 236 (2012) 1919-1925.
6. Y. El foutayeni, M. Khaladi, General Characterization of a Linear Complementarity Problem. American Journal of Modeling and Optimization, 1 (2013) 1-5.
7. Y. El foutayeni, H. El bouanani, M. Khaladi, The Linear Complementarity Problem and a Method to Find all its Solutions. Information in Sciences and Computing Volume 2014, Number 3, Article ID ISC370814, 05 pages.
8. Y. El foutayeni, M. Khaladi, A New Interior Point Method for Linear Complementarity Problem, Appl. Math. Sci., 4 (2010) 3289-3306.
9. Y. El foutayeni, M. Khaladi, A Min-Max Algorithm for Solving the Linear Complementarity Problem, J. Math. Sci. Appl, 1 (2013) 6-11.
10. Jean Fresnel. Algebre des matrices. Hermann, 2013.
11. Lloyd N Trefethen and David Bau III. Numerical linear algebra, volume 50. Siam, 1997.
12. G. H. GOLUB, C. REINSCH "Singular value decomposition and least squares solutions" in "Handbook for Automatic Computation - Linear Algebra, Vol. 2" J.H. WILKHINSON, C. REINSH Editors _ SPINGER VERLAG (1971).
13. Y. El foutayeni, H. El bouanani, M. Khaladi, An (m+1)-step iterative method of convergence order (m+2) for linear complementarity problems, Journal of Applied Mathematics and Computing, 54 (2017) 229-242.
14. H. El bouanani, Y. El foutayeni, M. Khaladi, A New Method for Solving Non-Linear Complementarity Problems, International Journal of Nonlinear Science, 19 (2015) 81-90.
15. R.W. Cottle, J.S. Pang, R.E. Stone, The Linear Complementarity Problem, Academic Press, New York, 1992.
16. K.G. Murty, On a characterization of P-matrices, SIAM J Appl Math, 20 (1971) 378-384.

17. K.G. Murty, On the number of solutions to the complementarity problem and spanning properties of complementary cones, Linear Algebra and Appl., 5 (1972) 65-108.

18. L.T. Watson, A Variational Approach to the Linear Complementarity Problem, Doctoral Dissertation, Dept. of Mathematics, University of Michigan, Ann Arbor, MI, 1974.

Yamna ACHIK[(1)], *Nossaiba BABA*[(1)], *Youssef EL FOUTAYENI*[(2,3)] *and Naceur ACHTAICH*[(1)]

[(1)] *Hassan II University of Casablanca, Morocco*

[(2)] *Cadi Ayyad University of Marrakech, Morocco*

[(3)] *Unit for Mathematical and Computer Modeling of Complex Systems, IRD, France*

E-mail address: yamna.achik.sma@gmail.com, noussaibababa1@gmail.com, foutayeni@gmail.com, nachtaich@gmail.com