Summer 2015

Improved Full-Newton-Step Infeasible Interior-Point Method for Linear Complementarity Problems

Mustafa Ozen

IMPROVED FULL-NEWTON-STEP INFEASIBLE INTERIOR-POINT METHOD FOR LINEAR COMPLEMENTARITY PROBLEMS

by

MUSTAFA OZEN

(Under the Direction of Goran Lesaja)

ABSTRACT

In this thesis, we present an improved version of Infeasible Interior-Point Method (IIPM) for monotone Linear Complementarity Problem (LCP). One of the most important advantages of this version in compare to old version is that it only requires feasibility steps. In the earlier version, each iteration consisted of one feasibility step and some centering steps (at most three in practice). The improved version guarantees that after one feasibility step, the new iterated point is feasible and close enough to central path. Thus, the centering steps are eliminated. This improvement is based on the lemma proved in [1]. Thanks to this lemma, proximity of the new point after the feasibility step is guaranteed with a more strict upper bound. Another advantage of this method is that it uses full-Newton steps, which means that no calculation of the step size is required at each iteration and that the cost is decreased. The implementation and numerical results demonstrate the reliability of the method.

Key Words: linear complementarity problem, interior-point method, infeasible interior-point method, full Newton-step

2009 *Mathematics Subject Classification*: 90C33, 90C51

IMPROVED FULL-NEWTON-STEP INFEASIBLE INTERIOR-POINT METHOD FOR LINEAR COMPLEMENTARITY PROBLEMS

by

MUSTAFA OZEN

B.S. in Mathematics and Computer Science at Cankaya University, 2014, Turkey

B.S. in Electronics and Communication Engineering (Double Major), 2014

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial Fulfillment

of the Requirement for the Degree

MASTER OF SCIENCE

STATESBORO, GEORGIA

2015

IMPROVED FULL-NEWTON-STEP INFEASIBLE INTERIOR-POINT METHOD FOR LINEAR COMPLEMENTARITY PROBLEMS

by

MUSTAFA OZEN

Major Professor: Goran Lesaja

Committee: Scott Kersey

Yan Wu

Electronic Version Approved:

July, 2015

DEDICATION

This thesis is dedicated to my all family and to all who believed and supported me during my whole education life.

ACKNOWLEDGEMENTS

In the first place, I would like to express my special appreciation and thanks to my advisor Dr. Goran Lesaja for enlightening me. Without his assistance and dedicated involvement in every step throughout the process, this paper would have never been accomplished.

I would also like to show my appreciation to my colleague Demet Yalman for her help, patience and encouragement in every steps of my study. I also take this opportunity to express gratitude to all of the Department faculty members especially to my committee for their encouragement, insightful comments, help and support. Finally, I would like to thank to my family for their encouragement and support during my whole education life.

TABLE OF CONTENTS

Page

DEDICATION v

ACKNOWLEDGEMENTS vi

LIST OF TABLES ix

LIST OF FIGURES x

LIST OF SYMBOLS xi

CHAPTER

1 Introduction 1

1.1 Problem 1

1.2 Solution Methods 2

1.3 Historical Overview 3

2 Linear Complementarity Problem 6

2.1 Linear Complementarity Problem 6

2.2 Classes of LCP 7

2.3 Examples 9

3 Lemke's Method 14

3.1 Definitions 14

3.2 Algorithm 14

3.3 Example 18

4 Improved Infeasible Full Newton-step Interior-Point Method 21

4.1 Idea of the Method 21
4.1.1 One main iteration of the algorithm 25
4.2 Feasibility Step 26
4.2.1 Pseudo-code of the Algorithm 29
5 Analysis of full Newton-step IIPM for LCP 30
5.1 Feasibility Step 30
6 Numerical Results 49
6.1 Comparison with old version 58
7 Conclusion 60
REFERENCES 63
A MATLAB Codes 66
A.1 LCPmain.m 66
A.2 FNSIIPM.m 69
A.3 Proximity.m 72

LIST OF TABLES

Table Page

3.1 Initial tableau for Phase I. 15

3.2 General form of initial tableau 16

5.1 Proximity of new iterates to μ-center for certain choice of τ and θ 44

5.2 Required number of iterations for different τ and θ values 48

6.1 Changes in $x^T s, \mu, \nu$ and $\delta(v^f)$ for the example 50

6.2 Required number of iterations for different θ, τ and ϵ values 51

6.3 $\theta = \frac{1}{39+n}, \tau = \frac{1}{5}, \epsilon = 10^{-4}$ and (x^0, s^0) is feasible pair 54

6.4 $\theta = \frac{1}{39+n}, \tau = \frac{1}{5}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair 54

6.5 $\theta = \frac{1}{40+n}, \tau = \frac{1}{4}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair 55

6.6 $\theta = \frac{1}{53+n}, \tau = \frac{1}{3}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair 55

6.7 $\theta = \frac{1}{170+n}, \tau = \frac{1}{2}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair 56

6.8 $\theta = 0.2, \epsilon = 10^{-4}$ 56

6.9 $\theta = 0.5, \epsilon = 10^{-4}$ 57

6.10 $\theta = 0.9, \epsilon = 10^{-4}$ 57

6.11 $\theta_{new} = \frac{1}{40+n}$, $\theta_{old} = \frac{1}{12n}$, $\tau = \frac{1}{4}, \epsilon = 10^{-4}$ 58

LIST OF FIGURES

Figure Page

2.1 Relations and examples of the classes of matrix M. 9

4.1 Graphical representation of $IIPM$ for LCP_ν 26

5.1 Graph of $\xi(t)$ for different $\theta \in [0, 1)$ 36

LIST OF SYMBOLS

A symbol table can be created in various ways. Here are a few:
Tabular environment:

$\mathbb{R}$	Real Numbers
$\mathbb{C}$	Real Numbers
$\mathbb{Z}$	Integers
$\mathbb{N}$	Natural Numbers
$\mathbb{N}_0$	Natural Numbers including 0
$L_p(\mathbb{R})$	p-integrable functions over $\mathbb{R}$
$L(X, Y)$	Linear maps from X to Y
$\text{rank}(T)$	Rank of a linear map

Multicols environment:

$\mathbb{R}$ Real Numbers
$\mathbb{C}$ Real Numbers
$\mathbb{Z}$ Integers
$\mathbb{N}$ Natural Numbers
$\mathbb{N}_0$ Natural Numbers including 0
$L_p(\mathbb{R})$ p-integrable functions over $\mathbb{R}$
$L(X, Y)$ Linear maps from X to Y
$\text{rank}(T)$ Rank of a linear map

Itemize environment:

- $\mathbb{R}$ Real Numbers
- $\mathbb{C}$ Real Numbers
- $\mathbb{Z}$ Integers
- $\mathbb{N}$ Natural Numbers
- $\mathbb{N}_0$ Natural Numbers including 0
- $L_p(\mathbb{R})$ p-integrable functions over $\mathbb{R}$
- $L(X, Y)$ Linear maps from X to Y
- $\text{rank}(T)$ Rank of a linear map

CHAPTER 1

INTRODUCTION

In this chapter, we present Linear Complementarity Problem (LCP) and solution methods with brief historical background.

1.1 Problem

Linear Complementarity Problem actually is not an optimization problem. Instead of optimizing an objective function, our aim is to find a vector satisfying given set of relationships which are linear equality constraints, nonnegativity conditions and complementarity conditions. The problem basically has the following standard form:

$$\begin{aligned} s = Mx + q \geq 0 \\ x^T s = 0, \\ x \geq 0, s \geq 0 \end{aligned} \tag{1.1}$$

where $x, s \in \mathbb{R}^n$, $M \in \mathbb{R}^{n \times n}$ and $q \in \mathbb{R}^n$. We denote (1.1) as $LCP(M, q)$. More specifically, as seen in the system (1.1), we seek a nonnegative vector pair (x, s) satisfying the linear equation $s = Mx + q$ and complementarity condition which is also orthogonality condition $x^T s = 0$.

As we mentioned, LCP is not an optimization problem, but it has robust relationship with both linear programming(LP) and quadratic programming(QP) problems. This strong relationship is based on the fact that Karush-Kuhn-Tucker (KKT) optimality conditions for LP and QP can be converted into LCP. Many optimization problems from engineering, finance, transportation, etc. can be directly written as LCP. Therefore, solving $LCPs$ has been very important topic for many years.

To illustrate, LP can be converted into LCP thanks to KKT conditions:

$$\begin{aligned} Ax = b, x \geq 0, \\ A^T y + s = c, s \geq 0, \\ s.t\ x^T s = 0 \end{aligned} \tag{1.2}$$

with variables $x, s \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, and input parameters $c \in \mathbb{R}^n, b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$. The conversion into LCP is given as follows:

$$LCP(M, q) : \bar{s} = M\bar{x} + q$$

$$\bar{x}^T \bar{s} = 0$$

$$where\ \bar{s} = \begin{bmatrix} s \\ 0 \end{bmatrix}, M = \begin{bmatrix} 0 & -A^T \\ A & 0 \end{bmatrix}, \bar{x} = \begin{bmatrix} x \\ y \end{bmatrix},\ and\ q = \begin{bmatrix} c \\ -b \end{bmatrix}$$

in which M is skew symmetric matrix. Also, Quadratic Programming (QP) problems can be converted into LCP in a similar way [2]. This means that if we can solve LCP, then we can solve both LP and QP problems.

1.2 Solution Methods

In this thesis, we mention two different solution methods. The first method is Lemke's Method connected to Simplex method and the second method is Interior Point Method (IPM) connected to Newton's method.

Lemke's Method is pivot based algorithm. Unfortunately, pivot based algorithms are not polynomial algorithms. They are successful in practice for low dimension problems, but when the dimension of problem increases, efficiency and numerical stability decrease. On the other hand, IPM is an iterative method. It is polynomial algorithm and very successful in practice. We will introduce an improved version of IPM which is very effective both theoretically and in practice and compare it with

the old version. These two methods will be explained in detail in Chapter 3 and Chapter 4 respectively.

1.3 Historical Overview

The existence of optimization methods can be traced back to the days of Newton, Lagrange and Cauchy. Studies on calculus of variations, which deals with the minimization of functions, were first considered by Bernoulli Euler, Lagrange and Weierstrass. Lagrange invented the method of optimization for constraint problems involving the addition of unknown multipliers. The first application of steepest descent method to solve unconstrained optimization problems was made by Cauchy. Following these contributions, optimization problems have been very popular for many years [4]. In the remaining part of this section, some major developments with a few milestones are given.

In 1947, Simplex Method (SM) which is one of the most famous optimization methods was developed by George Dantzig for LP [3]. This advancement initiated a strong research activity in the area of optimization. The main idea of this method is to travel from vertex to vertex along the boundary of a feasible region on which the minimization or the maximization process is achieved if the objective function is decreasing or increasing respectively [5]. SM is very popular because of its efficiency in solving large scale practical problems. According to some computational experiences, the number of iterations to solve the problem is $O(n)$ or sometimes $O(logn)$, with n being the number of variables in the problem.

SM is a pivot based algorithm. However, pivoting algorithms, unfortunately, are not polynomial algorithms. If the worst case complexity theory is considered, they are good in practice when the size of the problem is low. However, when the size of the problems increases, complexity and number of iterations increase as well. In

1971, an LP example for which some pivot based algorithms require an exponential number of pivots was shown by Klee and Minty [6]. A similar example was shown by Murty [7] for LCP, in 1978 (For more detail see [5]). However these examples have never appeared in practice.

More than 30 years after the appearance of the SM, Khachiyan [8] developed the first polynomial algorithm for LP which is called the Ellipsoid Algorithm, by applying Shor's original method [9], in 1979. Although it is theoretically better than SM, computational experiments showed that Ellipsoid Algorithm is not practical for solving LP problems. Nevertheless, Ellipsoid Algorithm still maintains its importance because of being a tool for developing polynomial time algorithms for a large class of convex optimization problems which are more general than linear programming [10].

One of the most important milestones in optimization is development of Interior Point Methods (IPM) for LP. The first IPM was proposed by Karmarkar [11], in 1984. The main idea of the algorithm differs from SM. It uses projective transformations and Karmarkar's potential function. IPM is an iterative method and iterates are calculated in the interior of feasible region. After development of Karmarkar's algorithm, huge number of papers were published about $IPMs$ for LP and related areas initiating the field of interior-point methods.

In 1986, it was proved that the Karmarkar's algorithm is connected to barrier and Newton-type methods [13]. Then, in 1988, Renegar [14] developed a first path-following Newton-type algorithm. These developments motivated the development of Newton method based $IPMs$ and different versions were proposed by many researchers. These various $IPMs$ can be classified into two main groups. The first group is potential reduction algorithms [15] based on the constant reduction of some potential function at each iteration. The second group is path-following algorithms [16] based on following central path which was studied first by Megiddo [17]. Both

groups contain algorithms based on primal, dual, or primal-dual formulation of LP.

After first appearance of IPM, its different types have been generalized to solve other optimization problems. The most important is the work of Nesterov and Nemirowski [20] that provided a general theory of polynomial IPM for convex programming problems. Also, $IPMs$ have been used for Nonlinear Complementarity Problems (see [21, 22, 23]). Nowadays, $IPMs$ have been used to solve many different types of optimization problems such as Conic optimization problems, $LCPs$, etc.

Concentrated study of LCP has begun in the mid 1960's [18]. To solve $LCPs$, in 1965, Lemke [19] proposed an algorithm so-called Lemke's Algorithm similar to SM. For many years, researchers have been searching for other methods to solve LCP. Eventually, many different and efficient $IPMs$ have been developed to solve LCP. In this thesis we will focus on improved version of one of these $IPMs$ which is called Full-Newton step Infeasible Interior Point Method.

CHAPTER 2

LINEAR COMPLEMENTARITY PROBLEM

In this chapter, we introduce the Linear Complementarity Problem (LCP) with its different classes and several applications. The organization of this chapter is as follows: in Section 2.1, we define the LCP in detail. Then, we present classes of LCP in Section 2.2. Finally, we finish this chapter by demonstrating LCP examples.

2.1 Linear Complementarity Problem

LCP is not an optimization problem, but it is closely related to optimization problems. It was shown that Karush-Kuhn-Tucker (KKT) optimality conditions for LO and convex QP can be written as $LCPs$. Besides, some classes of variational inequalities can be written as an LCP. Moreover, many important practical problems from engineering, finance, transportation, economics theory etc. can be directly written as LCP [24].

As we mentioned in Chapter 1, LCP in standard form is shown as follows: for a given matrix $M \in \mathbb{R}^{n\times n}$ and a vector $q \in \mathbb{R}^n$, we seek a pair of vectors $x, s \in \mathbb{R}^n$ satisfying the following system:

$$\begin{aligned} s &= Mx + q \geq 0 \\ xs &= 0, \\ x &\geq 0, s \geq 0 \end{aligned} \tag{2.1}$$

or equivalently

$$s = Mx + q, \quad 0 \leq x \perp s \geq 0.$$

where xs denotes the componentwise (Hadamard) product of vectors x and s as follows:

$$xs = (x_1 s_1, x_2 s_2, ..., x_n s_n)^T.$$

The unique solution pair (x^*, s^*) of the system (2.1) exist if the matrix M is symmetric positive definite. The set of points satisfying the system (2.1) except complementarity condition is called feasible region and is denoted as:

$$F = \{(x, s) \in \mathbb{R}^{2n} : s = Mx + q,\ x \geq 0,\ s \geq 0\} \tag{2.2}$$

and the solution set of the system (2.1) can be written as:

$$F^* = \{(x^*, s^*) \in F : {x^*}^T s^* = 0\}. \tag{2.3}$$

2.2 Classes of LCP

In this section, we present classes of matrices from the point of view of the LCP. For different type of matrix M, LCP type and solution method can change. Therefore, we need to define some types of matrix M for which IPM perform well. Here, we consider P_0 matrices and its subclasses. They can be defined as follows:

Let M be an $n \times n$ matrix and I be an index set. Then,

- **P_0-matrix:** If all the principal minors of M are non-negative, then M is called $P_0 - matrix$ [25]. Equivalently,

$$x \in \mathbb{R}^n \ and \ x \neq 0, \exists\, i \in I \ s.t. \ x_i(Mx)_i \geq 0. \tag{2.4}$$

- **P-matrix:** If all the principle minors of M are strictly positive, then M is called $P - matrix$ [26]. Equivalently,

$$x \in \mathbb{R}^n \ and \ x \neq 0, \exists\, i \in I \ s.t. \ x_i(Mx)_i > 0. \tag{2.5}$$

- **Positive semi-definite matrix (PSD):** The matrix M is called positive semi-definite matrix if it satisfies:

$$\forall x \in \mathbb{R}^n, x^T Mx \geq 0. \tag{2.6}$$

Additionally, if $x^T Mx > 0$, M is called positive definite.

- **Skew-symmetric matrix (SS):** The matrix M is called skew-symmetric if it satisfies:

$$\forall x \in \mathbb{R}^n, x^T Mx = 0. \tag{2.7}$$

- $P_*(\kappa)$**:** is the another class of matrices M such that

$$(1+4\kappa) \sum_{i \in I_+(x)} x_i [Mx]_i + \sum_{i \in I_-(x)} x_i [Mx]_i \geq 0 \tag{2.8}$$

where $x \in \mathbb{R}^n$, $\kappa \geq 0$, $[Mx]_i$ is the i-th component of Mx and $I_+ = \{i \in \mathbb{N} : x_i[Mx]_i > 0\}$, $I_- = \{i \in \mathbb{N} : x_i[Mx]_i < 0\}$. Moreover, if $\kappa \geq 0$, then the union of $P_*(\kappa)$ is called $P_* - matrices$.

- **Sufficient matrices (SM):** There are two types of sufficient matrices which are row sufficient (RS) and column sufficient (CS) matrices.

  The matrix M is called column sufficient if

$$\forall x \in \mathbb{R}^n, \forall i \in I \ s.t. \ x_i(Mx)_i \leq 0 \ \Rightarrow \ x_i(Mx)_i = 0 \tag{2.9}$$

and M is called row sufficient if M^T is column sufficient. Moreover, a matrix which is both row sufficient and column sufficient is called sufficient matrix [27].

These sub-classes of P_0 matrices are related to each other. Some relationship between classes are shown below.

$$SS \subset PSD, \ P \cap SS = \emptyset, \ (PSD \cup P) \subset P_* \subset CS \subset P_0 \tag{2.10}$$

Figure 5.1 presents the the relationship between types of M graphically. For different types of M matrix, LCP has different outcomes. For instance, if M is a column sufficient matrix, then the solution set of LCP is convex for every $q \in \mathbb{R}^n$ [28]. In addition, if M belongs to class of $P - matrices$, then the LCP has a unique

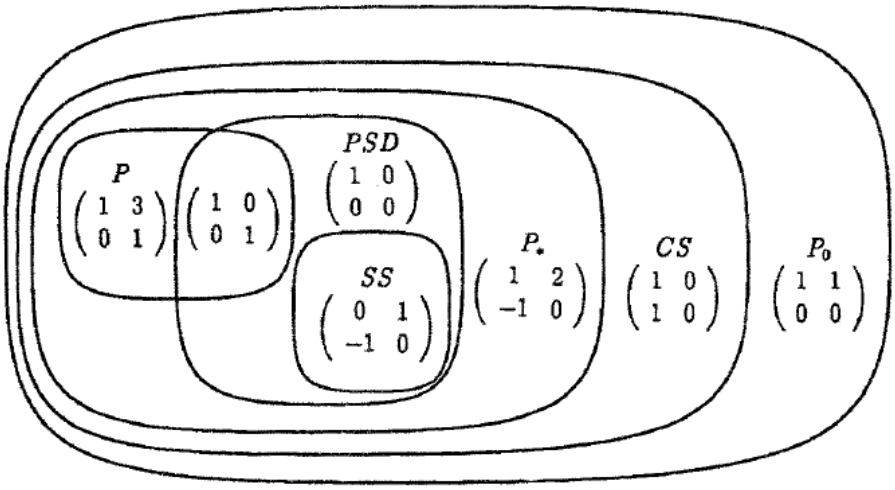

Figure 2.1: Relations and examples of the classes of matrix M.

solution for every $q \in \mathbb{R}^n$ [29]. For more detail of classes of LCP see [30]. In practice, most commonly used LCP is monotone LCP with a positive semi-definite matrix M because the LCP with positive semi-definite matrix M has a unique solution. That's why, we consider PSD matrix M in this thesis.

2.3 Examples

In this section we present some examples for the LCP. It has many applications in many areas. As we mentioned before, many important problems from engineering, finance-economy, transportation etc. can be directly written as LCP. First of all, we will give Quadratic Programming example. Also, we will show Bimatrix games as an application of LCP.

Example 1: Quadratic Programming

Quadratic Programming (QP) problems are one of the most common application

area of LCP. Basically, QPs optimize an quadratic objective function of several variables with respect to some set of constraint inequalities. The general QP can be written as:

$$\begin{aligned} min\ \ f(x) &= \frac{1}{2}x^T Q x + p^T x \\ subject\ to\ \ Ax &\geq b, \\ x &\geq 0 \end{aligned} \tag{2.11}$$

where $A \in \mathbb{R}^{m \times n}$ is constraint matrix, $Q \in \mathbb{R}^{n \times n}$ is symmetric coefficient matrix of quadratic terms, $x \in \mathbb{R}^n$ is vector of decision variables, $p \in \mathbb{R}^n$ is coefficient vector of linear terms in objective function and $b \in \mathbb{R}^m$ is vector of right-hand-side of constraint inequalities. When Q is a positive semi-definite matrix, then QP is called as *convex quadratic program* [2]. Also note that when $Q = 0$, then QP is reduced to LP.

In 1956, Frank and Wolfe proved the following theorem giving conditions that ensure the existence of a global solution to the QP in [31].

Theorem 2.1. *Let $S := \{x : Ax \geq b, x \geq 0\}$ be be a nonempty polyhedral set and f given by (2.11). Then either (2.11) has a global solution, or there is a feasible half line in S along which f approaches $-\infty$(that is, there are vectors x and $d \neq 0$ such that $x(\lambda) = x + \lambda d \in S$ for all $\lambda \geq 0$ and $f(x(\lambda)) \to -\infty as \lambda \to \infty$). Hence, if f is bounded below on S, it attains its minimum value on S.*

The KKT conditions are necessary conditions for any QP problem. Any local optimal solution will satisfy the KKT conditions. These conditions are also sufficient conditions for convex quadratic programs. More specifically, let Q be positive semi-definite matrix and x^* be a local solution for (2.11). Then, there exist a vector $u \in \mathbb{R}^m$ such that (x^*, u) pair satisfies the KKT optimality conditions:

$$\begin{aligned} 0 \leq x^* \perp Qx^* - A^T u + p \geq 0 \\ 0 \leq u \perp Ax^* - b \geq 0 \end{aligned} \tag{2.12}$$

where " $\perp$ " states the complementarity condition. Conversely, x^* is global solution of (2.11) if a pair $(x^*, u) \in \mathbb{R}^n \times \mathbb{R}^m$ satisfies (2.12) and Q is a positive semi-definite [2].

As we mentioned, QP problem shown in the system (2.11) can be written as LCP thanks to KKT optimality conditions. This conversion is presented as follows:

$$LCP(\bar{M}, \bar{q}): \ \ \bar{s} = \bar{M}\bar{x} + \bar{q}, \ \ \bar{x}, \bar{s} \geq 0 \ \ and \ \ \bar{x}^T \bar{s} = 0$$

where

$$\bar{M} = \begin{bmatrix} Q & -A^T \\ A & 0 \end{bmatrix}, \bar{x} = \begin{bmatrix} x \\ u \end{bmatrix} \ \ and \ \ \bar{q} = \begin{bmatrix} c \\ -b \end{bmatrix}.$$

QP problem is only one of many problems that can be written as LCP. Note that if $Q = 0$ then QP reduces to LP. Therefore, solving LCP is very important.

Example 2: Bimatrix Games

Bimatrix games are one of the direct applications of LCP. In the real world, some problems can be modelled with two objectives which are opposed to each other. For instance, we consider a game with two players called as Player 1 and Player 2. There are large number of plays and in each play, there are many choices so-called *pure strategies*. Player 1 picks one of m choices and Player 2 picks one of n choices. In a certain play, if Player 1 selects $i^{th} pure strategy$ and Player 2 selects $j^{th} pure strategy$, then Player 1 loses A_{ij} dollars and Player 2 loses B_{ij} dollars where A_{ij} and B_{ij} are an element in i^{th}-row and j^th-column of matrices A and B respectively. If $A_{ij} > 0$, then it represents a loss for Player 1 and if $A_{ij} < 0$, then it represents a gain for Player 1. This situation is the same for Player 2 and B_{ij}. The game is determined by the matrix pair (A, B) where A and B are called loss matrices.

The game is called *zero-sum game*, if $A + B = 0$. If $A + B \neq 0$, then the game

is known as *bimatrix games* or *two-person nonzero-sum game* with a finite number of *pure strategies* [2]. In this game, it is not efficient for players to choose a pure strategy which results the same play on every moves. Following a mixed strategy is generally better, in which the opponent cannot guess the next move easily. With this strategy the player chooses randomly moves available to him/her. Let Player 1 chooses play i with probability x_i and Player 2 chooses play j with probability y_j where the vectors x and y define the mixed strategies for each player and $\sum_{i=1}^{m} x_i = \sum_{j=1}^{n} y_i = 1$. Then, summation of all possible combinations of strategies for both players gives us the *expected loss* of Player 1 and Player 2 which are $x^T Ay$ and $x^T By$ respectively.

The $(\bar{x}, \bar{y})$ pair is called *Nash equilibrium pair* of strategies and aim is to minimize loss by changing his/her own strategy while the opponent's strategy is fixed, i.e,

$$\begin{aligned} \bar{x}^T A\bar{y} \leq x^T A\bar{y} \quad \forall x \geq 0,\ e_m^T x = 1, \\ \bar{x}^T A\bar{y} \leq \bar{x}^T Ay \quad \forall y \geq 0,\ e_n^T y = 1. \end{aligned} \tag{2.13}$$

where e_m and e_n are vectors of length m and n respectively whose elements are all 1.

The following lemma shows that the Nash equilibrium pairs for bimatrix games can be found by using LCP.

Lemma 2.2. *Suppose $A, B \in \mathbb{R}^{m\times n}$ are positive loss matrices representing a game (A, B), and suppose that $(s, t) \in \mathbb{R}^m \times \mathbb{R}^n$ solves $LCP(M, q)$, where*

$$M = \begin{bmatrix} 0 & A \\ B^T & 0 \end{bmatrix}, \qquad q = -e_{m+n} \in \mathbb{R}^{m+n}.$$

Then, $(\bar{x}, \bar{y})$ is an equilibrium pair of (A, B) if

$$\bar{x} = \frac{s}{e_m^T s} \text{ and } \bar{y} = \frac{t}{e_n^T t}.$$

Proof. Let write out the LCP conditions explicitly:

$$\begin{aligned} 0 \leq At - e_m \perp s \geq 0, \\ 0 \leq B^T s - e_n \perp t \geq 0. \end{aligned} \tag{2.14}$$

It is obvious that since $At - e_m \geq 0$, t must be non-zero and similarly s must be non-zero. Hence, $\bar{x}$ and $\bar{y}$ are well defined. From definitions of $\bar{x}, \bar{y}, e_m$ and e_n we have that $\bar{x} \geq 0, \bar{y} \geq 0, e_m^T \bar{x} = 1$ and $e_n^T \bar{y} = 1$. Thus, $\bar{x}$ and $\bar{y}$ are mixed strategies.

By complementarity condition, we have

$$\bar{x}^T(At - e_m) = \frac{s^t}{e_m^T s}(At - e_m) = 0.$$

Thus $\bar{x}^T At = \bar{x}^T e_m = 1$. By using this relation, we obtain that

$$A\bar{y} - (\bar{x}^T A\bar{y})e_m = \frac{1}{e_n^T t}(At - (\bar{x}^T At)e_m) = \frac{1}{e_n^T}(At - e_m) \geq 0.$$

So, given any strategy x, we have from $x \geq 0$ and from the expression above that

$$0 \leq x^T(A\bar{y} - e_m(\bar{x}^T A\bar{y})) \rightarrow x^T A\bar{y} \geq (e_m^T x)\bar{x}^T A\bar{y} = \bar{x}^T A\bar{y}.$$

Hence the first equation in (2.13) is satisfied. The second equation of (2.13) can be shown in a similar way. Thus, as claimed, the pair $(\bar{x}, \bar{y})$ is a Nash equilibrium pair. For detail see [2]. □

Thanks to this result, we can solve the bimatrix games by the following procedure:

1. To obtain A and B having positive entries, increase all the entries in each loss matrix by a constant amount.

2. Set
$$M = \begin{bmatrix} 0 & A \\ B^T & 0 \end{bmatrix}, \ and \ q = \begin{bmatrix} -e_m \\ -e_n \end{bmatrix},$$
and solve $LCP(M, q)$.

3. Set
$$\bar{x} = \frac{s}{e_m^T s} \ and \ \bar{y} = \frac{t}{e_n^T t}.$$

CHAPTER 3

LEMKE'S METHOD

In this chapter, we present a well known pivot based algorithm that is called Lemke's Method which is the first algorithm proposed to solve $LCPs$. It was proposed in 1965 by Lemke [19]. Some related definitions, algorithm of Lemke's Method and related examples are given in this chapter.

3.1 Definitions

Before giving the algorithm, we need to make the following definitions:

Let consider the pair (x, s) where $x, s \in \mathbb{R}^n$.

(a) A pair (x, s) is called *feasible* for $LCP(M, q)$ if it satisfies the following:

$$s = Mx + q, \quad x \geq 0, s \geq 0.$$

(b) A component s_i of vector s is called the *complement* of component x_i of vector x, and vice versa, for $i = 1, 2, ..., n$.

(c) A pair (x, s) is called *complementary* if $x \geq 0, s \geq 0$, and $x^T s = 0$. (Note that the Hadamard product of vectors x and s is 0, i.e., $x_i s_i = 0$ for $i = 1, 2..., n$.)

(d) A pair (x, s) is called *almost complementary* if $x \geq 0, s \geq 0$ and $x_i s_i = 0$ for $i = 1, 2, ..., n$ except for a single index j, where $1 \leq j \leq n$.

3.2 Algorithm

First recall the standard form of LCP denoted as $LCP(M, q)$ described in (2.1):

$$\begin{aligned} s = Mx + q \geq 0 \\ xs = 0, \\ x \geq 0, s \geq 0 \end{aligned} \tag{3.1}$$

If matrix M in the system (3.1) is positive semi-definite, then Lemke's method generates a finite sequence of feasible almost complementary pairs that terminates at a complementary pair or an unbounded ray.

This algorithm has two phases: Phase I and Phase II. As in the SM, we must find an initial pair to run the algorithm. This initial pair is usually obtained via Phase I. Depending on particular structures of $LCPs$, there are different Phase I schemes. Here, we included the most widely applicable scheme requiring only one pivot. On the other hand, unlike Phase I, Phase II performs several pivots. It generates a sequence of almost complementary vector pairs. At each iteration, the pivot row is selected by means of ratio test as we do in SM. Purpose of using ratio test is to guarantee the nonnegativity of components of vectors x and s. The algorithms for Phase I and Phase II can be summarized as follows:

Phase I: Generates a feasible almost complementary tableau

1. If $q \geq 0$, STOP: the feasible complementary pair $(x, s) = (0, q)$ is a solution pair of $LCP(M, q)$.

2. Otherwise, add the artificial variables x_0 and s_0 satisfying the following relationships:

$$s = Mx + ex_0 + q, \quad s_0 = x_0$$

where $e \in \mathbb{R}^n$ is the vector of ones. Then the initial tableau is as follows:

		x	x_0	1
s	$=$	M	e	q
s_0	$=$	0	1	0

Table 3.1: Initial tableau for Phase I.

3. Make this tableau feasible by carrying a Jordan exchange (see Chapter 2 in [2]) on the x_0 column and the corresponding row having the most negative q_i.(This is the only pivoting step in Phase I and corresponds to the special pivot in Phase I of SM for LP.)

4. Continue to Phase II without removing the artificial variables from the tableau.

Phase II: Generates a feasible complementary or unbounded tableau

It starts with a feasible almost complementary pair (x, s) and corresponding tableau in Jordan exchange form. The tableau obtained via Phase I is used as an initial tableau here. The general form of initial tableau is as follows:

		s_{I_1}	x_{J_2}	1
x_{J_1}	$=$	$H_{I_1J_1}$	$H_{I_1J_2}$	h_{I_1}
s_{I_2}	$=$	$H_{I_2J_1}$	$H_{I_2J_2}$	h_{I_2}

Table 3.2: General form of initial tableau

Record the variable that become a column label(become nonbasic) at the previous iteration.

1. Pivot column selection: Choose the column s which is the complement of the variable that became nonbasic at the previous pivot.

2. Pivot row selection: Choose the row r satisfying

$$-h_r/H_{rs} = \min_i\{-h_i/H_{is} | H_{is} < 0\}.$$

   An unbounded ray has been found if all $H_{is} \geq 0$, STOP.

3. Perform a Jordan exchange on element H_{rs}. If the pair (x, s) is complementary then (x, s) is a solution, STOP: else, go to Step 1.

More specifically, in this algorithm, Step 1 ensures the almost complementarity by moving a component into the basis as soon as its complement is moved out. By doing this, we guarantee that exactly one of x_i and s_i is basic while the others are nonbasic. This fact ensures the almost complementarity property, i.e., $x_i s_i = 0$ for all except one since nonbasic variables are assigned the value 0. In Step 2, similar to SM, we use ratio test to maintain nonnegativity of all the components in the last column. Thus, the nonbasic variable in the column s increases away from zero until it causes one of the basic variables to become zero. This basic variable is determined by using ratio test. In addition, in practice, we don't need to insert s_0 row into tableau because s_0 is always equal to x_0 and it remains in the basis throughout.

In theory, Lemke's method depends on whether or not the algorithm terminates at an unbounded ray. More specifically, termination of the algorithm depends on some conditions on the matrix M. The following theorem presents two fundamental results under some conditions on the matrix M.

Theorem 3.1. *(i) If $M \in \mathbb{R}^{n\times n}$ is positive semi-definite matrix, then Lemke's algorithm terminates at a solution of $LCP(M,q)$ or at an unbounded ray. Additionally, if the set $\{x|Mx+q \geq 0, x \geq 0\}$ is empty, then there is no feasible pair (x,s).*

(ii) If $M \in \mathbb{R}^{n\times n}$ is positive definite matrix, then Lemke's method terminates at the unique solution of $LCP(M,q)$ for any $q \in \mathbb{R}^n$.

Proof. See [32]. □

Note: When the LCP is obtained from QP problem, then the matrix M is positive semi-definite if and only if the matrix Q is positive semi-definite.

Proof. First recall the QP example from the Section 2.3 in which the matrix M is

defined by

$$M = \begin{bmatrix} Q & -A^T \\ A & 0 \end{bmatrix}. \tag{3.2}$$

Then,

$$\begin{bmatrix} x^T & u^T \end{bmatrix} M \begin{bmatrix} x \\ u \end{bmatrix} = x^T Q x - x^T A^T u + u^T A x = x^T Q x.$$

Therefore, If Q is positive semi-definite, then $x^T Q x \geq 0$ which concludes the positive semi-definiteness of M. □

3.3 Example

In this section we solve a simple LCP example using Lemke's method.

Consider $LCP(M, q)$ with

$$M = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \quad q = \begin{bmatrix} -2 \\ -1 \end{bmatrix}$$

Then, we can write $LCP(M, q) : s = Mx + q$ as follows:

$$\begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -2 \\ -1 \end{bmatrix}$$

with initial tableau

		x_1	x_2	1
s_1	=	1	0	-2
s_2	=	-1	1	-1

Apply Lemke's method to find a feasible, complementary solution pair (x, s).

1. Since $q < 0$, we need to apply Phase I. Then $LCP(M, q)$ becomes $s = Mx + ex_0 + q$ with corresponding tableau

		x_1	x_2	x_0	1
s_1	=	1	0	1	-2
s_2	=	-1	1	1	-1

The pivot is x_0 with the row 1 that has been chosen corresponding to the most negative q_i in the last column. Then, the following tableau represents an almost complementary solution.

		x_1	x_2	s_1	1
x_0	=	-1	0	1	2
s_2	=	-2	1	1	1

2. Now, we have a feasible almost complementary pair (x, s). We can continue with Phase II of Lemke's method.

   Since at the last pivot step, s_1 became nonbasic, we choose its complement to become nonbasic which is x_1. Therefore, the pivot column is $s = 1$.

   Apply ratio test to identify the pivot row.

$$\min_i \{\frac{-2}{-1}, \frac{-1}{-2}\}$$

   The ratio test chooses row $r = 2$ as the pivot row. With these current selections, after performing Jordan exchange, we have the following tableau:

		s_2	x_2	s_1	1
x_0	=	0.5	-0.5	0.5	1.5
x_1	=	-0.5	0.5	0.5	0.5

   Since the current pair is still almost complementary, we carry out another pivot. At the last pivot, s_2 left the basis, that's why we need to choose its complement x_2 to enter at this pivot $(s = 2)$.

By applying ratio test

$$\min_i \{\frac{-1.5}{0.5}, \frac{-0.5}{-0.5}\}$$

we choose row $r = 1$ as the pivot row. Hence, the pivot is performed on the element in position $(1, 2)$. Then, we have the following tableau:

		s_2	x_0	s_1	1
x_2	=	1	-2	1	3
x_1	=	0	-1	1	2

The resulting pair (x, s) is both feasible and complementary. Therefore, the solution to the $LCP(M, q)$ is $x_1 = 2, x_2 = 3$.

As seen from the example above, Lemke's method finds a feasible and complementary solution to the LCP under some conditions on matrix M. Since a lot of problem can be written as LCP and Lemke's method solves LCP, we can solve many problem via Lemke's method. For more examples see [2]. As we mentioned in Section 1.2, since Lemke's method is a pivot based algorithm, it is not a polynomial algorithm. Although it is successful for low dimension problems, in practice, efficiency and numerical stability decreases when the dimension of the problem increases. Therefore, we prefer more stable, efficient and polynomial algorithm which we introduce in the next Chapter.

CHAPTER 4

IMPROVED INFEASIBLE FULL NEWTON-STEP INTERIOR-POINT METHOD

In this chapter, we discuss the main part of this thesis which is infeasible full Newton-step interior point method ($IIPM$) to solve monotone LCP. We take step size $\alpha = 1$, thus we call this method as full Newton-step. Also, this algorithm finds a solution for all feasible or infeasible initial starting point. Hence, this method is an infeasible method. First, we will explain idea of this method, then we will introduce feasibility step of this method. In the old version of this method, each iteration requires a feasibility step and few centring steps. Here, we improved the algorithm thanks to the lemma proved in [1]. After the improvement, this algorithm ensures that the new iterated point is close enough to the central path immediately after feasibility step so centering steps are no longer needed. In the following section, we introduce the concept of this new version.

4.1 Idea of the Method

First, recall the monotone LCP:

$$\begin{aligned} s &= Mx + q, \\ xs &= 0, \\ x \geq 0, s &\geq 0 \end{aligned} \tag{4.1}$$

where $M \in \mathbb{R}^{n \times n}$ is a positive semi-definite matrix, $q \in \mathbb{R}^n$ is a constant vector. Our aim is to find a pair of vectors $x, s \in \mathbb{R}^n$ satisfying the system (4.1). The pair (x, s) is called *ϵ-solution* if the following inequalities are satisfied:

$$||s - Mx - q|| \leq \epsilon \ \ and \ \ x^T s \leq \epsilon$$

As we introduced in Chapter 2, $x^T s = 0$ is equivalent to $xs = 0$ which represents the Hadamard product of vectors x and s.

It is known that $IPMs$ are Newton's method(NM) based algorithms. Therefore, they are iterative algorithms. The NM is essential part of the $IPMs$. They use NM to find approximated solutions to the systems. In our case $IIPM$ uses NM to solve system (4.1). However, the complementarity condition in the system (4.1) can be a problem for NM. If any component of vectors x and s become 0, it stays 0 forever and the iteration sequence never converges to the solution. Hence, to prevent this problem we replace the complementarity condition by so-called centring condition $xs = \mu e$, where μ can be any positive number. Then the system (4.1) becomes

$$\begin{aligned} s &= Mx + q, \\ xs &= \mu e, \\ x \geq 0, s &\geq 0 \end{aligned} \tag{4.2}$$

It is known that if matrix M is positive definite, then the system (4.2) has a unique solution. Additionally, if (4.2) has a solution for some $\mu > 0$, then there exist a solution for every $\mu > 0$. This happens if and only if the system (4.2) satisfies the *Interior point condition*(IPC), i.e., there exist a pair (x^0, s^0) such that $s^0 = Mx^0 + q$ and $x^0 > 0, s^0 > 0$. In this case, the IPM is called feasible IPM. The only drawback of feasible IPM is that it is not easy to find an initial feasible pair. Therefore, we consider infeasible IPM which works for every initial starting pairs. Before explaining one iteration of this algorithm, we give the following definition to make the concept more precise.

Definition 4.1. *(i) The parametric solution $(x(\mu), s(\mu))$ to the system (4.2) is called μ-center and the set of all μ-centers is called central path.*

(ii) For some $\mu > 0$, let (x^0, s^0) be a random starting pair such that $x^0 s^0 = \mu^0 e$.

Then, we denote the residual r^0 *as:*

$$s^0 - Mx^0 - q = r^0 \tag{4.3}$$

The main idea of $IIPM$ is to follow central path by reducing μ to 0. However, it is not efficient to find exact μ-centers because to find exact μ-center, NM performs infinitely many iteration which is not possible in practice. Thus, we follow the central path approximately. More specifically, we measure the proximity of new iterated points to the central path and restrict it to be less than a certain threshold τ. Formulations for proximity will be given later.

Let's consider a positive random starting pair (x^0, s^0) such that $x^0 s^0 = \mu^0 e$ where the duality gap $\mu^0 = \frac{{x^0}^T s^0}{n}$. Then, for any ν with $0 < \nu \leq 1$ we consider a small perturbation which we call *perturbed* LCP, denoted by LCP_ν. The LCP_ν is defined by

$$\begin{aligned} s - Mx - q &= \nu r^0, \\ xs &= 0, \\ x \geq 0, s &\geq 0 \end{aligned} \tag{4.4}$$

where r^0 is defined as in (4.3). It is easy to show that when $\nu = 1$, the initial pair (x^0, s^0) is a μ^0-center. Namely, the pair (x^0, s^0) is strictly feasible solution for $LCP_{\nu=1}$. Therefore, the IPC is satisfied when $\nu = 1$.

Lemma 4.2. *The original problem (4.1) is feasible if and only if the* LCP_ν *in (4.4) is feasible for* $0 < \nu \leq 1$.

Proof. ($\Rightarrow$): Let the original problem (4.1) be feasible. Let (x^*, s^*) be a feasible solution to (4.1). Then,

$$s^* = Mx^* - q \;\; such\ that \;\; x^* \geq 0, s^* \geq 0.$$

For $0 < \nu \leq 1$, we consider convex combinations of x^*, x^0 and s^*, s^0, i.e.,

$$x = (1-\nu)x^* + \nu x^0, \;\; s = (1-\nu)s^* + \nu s^0.$$

Also, since $x^*, s^* > 0$ and $x^0, s^0 > 0$, then obviously $x, s > 0$. Then we have

$$\begin{aligned}
s - Mx - q &= (1-\nu)s^* + \nu s^0 - M[(1-\nu)x^* + \nu x^0] - q \\
&= (1-\nu)s^* + \nu s^0 - (1-\nu)Mx^* - \nu Mx^0 - q \\
&= (1-\nu)(s^* - Mx^*) + \nu(s^0 - Mx^0) - q \\
&= (1-\nu)(s^* - Mx^* - q + q) + \nu(s^0 - Mx^0 - q + q) - q \\
&(Since\ (x^*, s^*)\ and\ (x^0, s^0)\ are\ feasible, \\
&then\ s^* - Mx^* - q = 0)\ and\ s^0 - Mx^0 - q = r^0) \\
&= (1-\nu)(q) + \nu(r^0 + q) - q \\
&= (1-\nu)(q) + \nu r^0 + q\nu - q \\
&= q(1 - \nu + \nu) + \nu r^0 - q \\
&= q + \nu r^0 - q \\
&= \nu r^0.
\end{aligned}$$

This shows that (x, s) is feasible for LCP_ν (4.4).

($\Leftarrow$) : To prove the inverse implication, suppose LCP_ν is feasible for $0 < \nu \leq 1$. Then, letting ν go to zero it follows that LCP is feasible. □

Since we consider to solve LCP_ν (4.4) via IPM, complementarity condition $xs = 0$ can be a problem for NM because of the same reason as in (4.2). Hence, we consider central path for LCP_ν. Lemma 4.2 implies that the LCP_ν satisfies the IPC. Thus, the central path exists, i.e., for some positive μ, we modify the system (4.4) as follows:

$$\begin{aligned}
s - Mx - q &= \nu r^0, \\
xs &= \mu e, \\
x \geq 0, s &\geq 0
\end{aligned} \tag{4.5}$$

This system has a unique solution for every $\mu > 0$ if M is positive semi-definite matrix

and this solution is denoted by $(x(\mu,\nu), s(\mu,\nu))$. These pairs are (μ,ν)-centers of LCP_ν.

Instead of finding exact solution, here we consider to find iterates that are in the τ-neighbourhood of (μ,ν)-centers. Then, we simultaneously reduce μ and ν with a positive so-called *barrier parameter* $\theta \in [0,1)$, i.e.,

$$\mu^+ = (1-\theta)\mu$$
$$\nu^+ = (1-\theta)\nu$$

Note that when μ and ν approach zero, we will obtain the approximate solution for LCP (4.1). Note that, when $\mu = \mu^0$ and $\nu = 1$, $(x(\mu^0,1), s(\mu^0,1)) = (x^0, s^0)$. In what follows the parameters μ and ν are connected as $\nu = \frac{\mu}{\mu^0}$.

We also need to define the proximity of (x,s) to the μ-centers. The proximity which is denoted by $\delta(x,s;\mu)$ is defined as:

$$\delta(x,s;\mu) = \delta(v) = \frac{1}{2}||v - v^{-1}||, \quad where \quad v = \sqrt{\frac{xs}{\mu}}. \tag{4.6}$$

Note that $\delta(x,s;\mu) = 0$ means (x,s) is μ-center. One main iteration of $IIPM$ is explained in the following Section.

4.1.1 One main iteration of the algorithm

We assume that at the start of each iteration, $\delta(x,s;\mu)$, defined in (4.6), is less than or equal to a threshold $\tau > 0$. As we established before, when $\mu = \mu^0$ and $\nu = 1$, (x^0, s^0) is μ-center of LCP_ν. Thus, initially we have $\delta(x^0, s^0; \mu^0) = 0 < \tau$ which satisfies our assumption at the first iteration.

Suppose $\mu \in (0, \mu^0]$, we have (x,s) pair feasible to the system (4.5) for $\nu = \mu/\mu^0$ and $\delta(x,s;\mu) \leq \tau$. Then, we reduce μ to $\mu^+ = (1-\theta)\mu$ and $\nu = \mu^+/\mu^0 = (1-\theta)\nu$ with $\theta \in [0,1)$ and find new iterates (x^+, s^+) satisfying (4.5) with μ^+ and ν^+.

To be more precise, at each iteration, there exists only one feasibility step. This step serves to get strictly feasible iterates to the LCP_{ν^+} denoted by (x^f, s^f). In the old version, after one feasibility step, we must perform a few centering steps to satisfy $\delta(x^+, s^+; \mu^+) \leq \tau$. However, thanks to the lemma proved in [1] our new points (x^f, s^f) after one feasibility step ensure that $\delta(x^f, s^f; \mu^+) \leq \tau$. We will elaborate the improvement in Chapter 5. The following figure presents one iteration of the algorithm graphically.

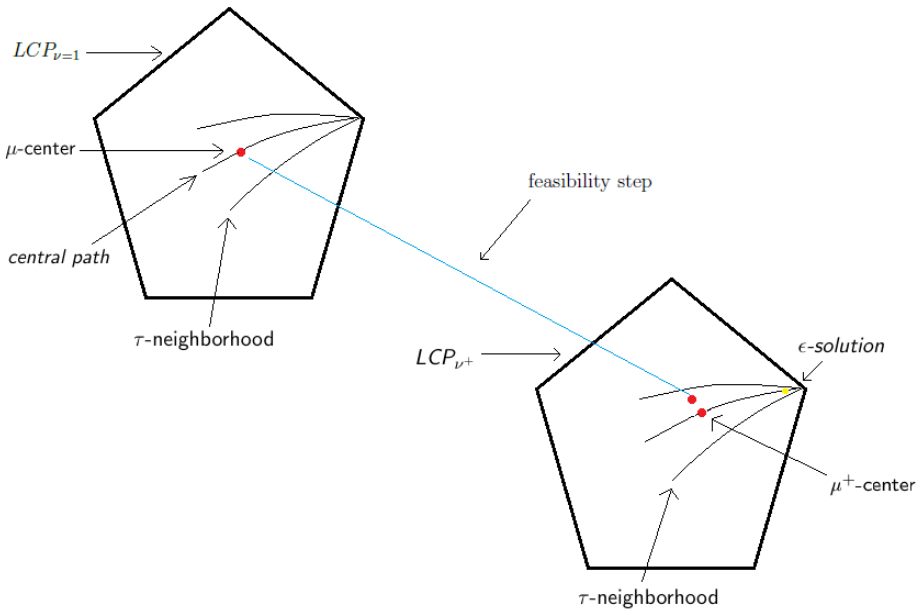

Figure 4.1: Graphical representation of $IIPM$ for LCP_ν

Now, we describe the feasibility step which is the only required step of this algorithm in the next Section.

4.2 Feasibility Step

Let (x, s) be an approximate solution to LCP_{ν^+}. Our aim is to find a strictly feasible solution to LCP_{ν^+}. First of all, we need to find search directions $\triangle^f x$ and $\triangle^f s$. In

line with this purpose, we can rewrite the system (4.5) as follows:

$$F(x,s) = \begin{bmatrix} s - Mx - q - \nu^+ r^0, \\ xs - \mu^+ e \end{bmatrix} = 0 \tag{4.7}$$

To solve (4.7), we use NM. When we apply NM, we have the following equation which gives the search directions:

$$\bigtriangledown F \begin{bmatrix} \triangle^f x \\ \triangle^f s \end{bmatrix} = -F(x,s) \tag{4.8}$$

where $\bigtriangledown F$ is Jacobian of $F(x,s)$. Equivalently, to find search directions we can solve the system

$$\begin{aligned} M \triangle^f x - \triangle^f s &= \theta\nu r^0, \\ s \triangle^f x + x \triangle^f s &= (1-\theta)\mu e - xs. \end{aligned} \tag{4.9}$$

This reduction can be shown as follows:

$$(i)\ \ (s + \triangle^f s) - M(x + \triangle^f x) - q = \nu^+ r^0 \ \ s.t.$$
$$(x + \triangle^f x)(s + \triangle^f s) = \mu^+ e$$

$$(ii)\ \ s + \triangle^f s - Mx - M \triangle^f x - q = \nu^+ r^0 \ \ s.t.$$
$$xs + x \triangle^f s + s \triangle^f x + \triangle^f x \triangle^f s = \mu^+ e$$

$$(iii)\ \ M \triangle^f x - \triangle^f s = s - Mx - q - \nu^+ r^0 \ \ s.t.$$
$$x \triangle^f s + s \triangle^f x + \triangle^f x \triangle^f s = \mu^+ e - xs$$

Since (x,s) is feasible for LCP_ν, $s - Mx - q = \nu r^0$, also we can neglect $\triangle^f x \triangle^f s$ term. Thus, we have

$$(iv)\ \ M \triangle^f x - \triangle^f s = (\nu - \nu^+) r^0 \ \ s.t.$$
$$x \triangle^f s + s \triangle^f x = \mu^+ e - xs$$

Since $\nu - \nu^+ = \nu - (1-\theta)\nu = \theta\nu$ and $\mu^+ = (1-\theta)\mu$, we have the result (4.9):

$$(v)\ M \triangle^f x - \triangle^f s = \theta\nu r^0 \ \ s.t.$$
$$x \triangle^f s + s \triangle^f x = (1-\theta)\mu e - xs$$

After finding search directions by solving (4.9), we update the new iterates (x^f, s^f) by:

$$x^f = x + \alpha \triangle^f x$$
$$s^f = s + \alpha \triangle^f s$$

where α is called step size. Since we consider full Newton-step, in our case $\alpha = 1$. Thus,

$$\begin{aligned} x^f &= x + \triangle^f x \\ s^f &= s + \triangle^f s \end{aligned} \tag{4.10}$$

Thanks to improvement that we have done, (x^f, s^f) are both feasible and close enough to central path, i.e., $\delta(x^f, s^f; \mu^+) \leq \tau$. This will be shown in the next Chapter together with the analysis of feasibility step and entire algorithm.

4.2.1 Pseudo-code of the Algorithm

The following pseudo-code gives a strictly feasible ϵ-solution to the LCP. In this version, we eliminate the centering steps because after one feasibility step, new iterates (x^f, s^f) are close enough to central path thanks to the improvement. Namely, after each iteration $\delta(x^f, s^f; \mu^+) \leq \tau$ condition is satisfied.

Data: **Input**

Accuracy parameter $\epsilon > 0$,

Barrier update parameter $\theta,\ 0 < \theta < 1$,

Initial points:

$x^0 > 0, s^0 > 0$,

$\mu = \mu^0$ with $x^0 s^0 = \mu^0 e$,

$\nu = 1$.

begin

while $max(x^T s, ||s - Mx - q||) \geq \epsilon$ **do**

begin

Update $\mu = (1-\theta)\mu$, $\nu = (1-\theta)\nu$;

Feasibility step

Calculate $\triangle_x^f, \triangle_s^f$ by solving (4.9);

Update $(x, s) = (x, s) + (\triangle_x^f, \triangle_s^f)$ as in (4.10);

end

Algorithm 1: Full Newton-step Infeasible IPM Algorithm for LCP

CHAPTER 5

ANALYSIS OF FULL NEWTON-STEP IIPM FOR LCP

In this Chapter, we analyse the feasibility step of the algorithm that is given in Section 4.2. More specifically, we analyse convergence of the algorithm and calculate iteration number that is required to find an ϵ-solution.

5.1 Feasibility Step

Analysis of the feasibility step is based on finding an upper bound for the proximity of iterates. Old version of the algorithm requires two steps. At the first step, we find a pair (x^f, s^f) that is strictly feasible but possibly not close enough to μ-center. Then, we perform a few centering steps to make this new iterate closer to μ-center under control of a positive τ, i.e., $\delta(x^f, s^f; \mu^+) \leq \tau$. Since we want to eliminate centering steps, we find an upper bound for $\delta(x, s, \mu)$ such that after each feasibility step, new iterated pair (x^f, s^f) is strictly feasible and satisfy $\delta(x^f, s^f; \mu^+) \leq \tau$. In this section, we show how to find such upper bound, and required number of iterations to have an ϵ-solution.

Our assumption is that we choose $x^0 = \gamma_p e$ and $s^0 = \gamma_d e$ so that $\mu^0 = \frac{(x^0)^T s^0}{n}$. Then, we start the analysis of feasibility step by presenting the feasibility of new iterates (x^f, s^f), however, before that let us recall the system (4.9),

$$M \triangle^f x - \triangle^f s = \theta \nu r^0, \tag{5.1a}$$

$$s \triangle^f x + x \triangle^f s = (1-\theta)\mu e - xs. \tag{5.1b}$$

The analysis will require the following transformations

$$v = \sqrt{\frac{xs}{\mu}}, \quad d_x = \frac{v \triangle x}{x}, \quad d_s = \frac{v \triangle s}{s}. \tag{5.2}$$

where $x, s \in \mathbb{R}^n$. Our aim is to transform (5.1) into d_x, d_s form by using (5.2). Now,

substitute (5.2) into (5.1). Then we have:

$$\begin{aligned} M\frac{xd_x}{v} - \frac{sd_s}{v} &= \theta\nu r^0, \\ x\frac{sd_s}{v} + s\frac{xd_x}{v} &= (1-\theta)\mu e - xs. \end{aligned} \tag{5.3}$$

Let $X = diag(x)$, $S = diag(s)$, $V = diag(v)$ and $V^{-1} = diag(v^{-1})$. Then, we write the system (5.3) in its matrix form as follows:

$$MV^{-1}Xd_x - SV^{-1}d_s = \theta\nu r^0, \tag{5.4a}$$

$$XV^{-1}Sd_s + SV^{-1}Xd_x = (1-\theta)\mu e - XSe. \tag{5.4b}$$

After multiplying both sides of (5.4a) by $S^{-1}V$ and both sides of (5.4b) by $X^{-1}VS^{-1}$, we have:

$$S^{-1}VMV^{-1}Xd_x - S^{-1}VSV^{-1}d_s = S^{-1}V\theta\nu r^0,$$

$$X^{-1}VS^{-1}XV^{-1}Sd_s + X^{-1}VS^{-1}SV^{-1}Xd_x = X^{-1}VS^{-1}[(1-\theta)\mu e - XSe].$$

which is equal to:

$$S^{-1}MXd_x - d_s = S^{-1}V\theta\nu r^0, \tag{5.5a}$$

$$d_s + d_x = X^{-1}VS^{-1}[(1-\theta)\mu e - XSe]. \tag{5.5b}$$

Then, substitute $V = \sqrt{\frac{XS}{\mu}}$ into (5.5):

$$S^{-1}MXd_x - d_s = S^{-1}X^{1/2}S^{1/2}\mu^{-1/2}\theta\nu r^0,$$

$$d_s + d_x = X^{-1}VS^{-1}(1-\theta)\mu e - X^{-1}VS^{-1}XSe.$$

which can be written as:

$$S^{-1/2}X^{1/2}MS^{-1/2}X^{1/2}d_x - d_s = S^{-1/2}X^{1/2}\mu^{-1/2}\theta\nu r^0, \tag{5.6a}$$

$$d_s + d_x = \frac{\mu}{XS}\sqrt{\frac{XS}{\mu}}(1-\theta)e - Ve. \tag{5.6b}$$

Now, let $D = S^{-1/2}X^{1/2}$, then (5.6) becomes

$$DMDd_x - d_s = D\mu^{-1/2}\theta\nu r^0, \tag{5.7a}$$

$$d_s + d_x = \sqrt{\frac{\mu}{XS}}(1-\theta)e - Ve. \tag{5.7b}$$

Finally, let $\widetilde{M} = DMD$ and rewrite X, S, V as x, s, v. Then we have the d_x, d_s form of the system (5.1) which is as follows:

$$\widetilde{M}d_x - d_s = D\mu^{-1/2}\theta\nu r^0, \tag{5.8a}$$

$$d_x + d_s = v^{-1}(1-\theta) - v. \tag{5.8b}$$

Let, (x, s) be a feasible pair for LCP_ν. Then, from (4.10), after one feasibility step we get (x^f, s^f) such that:

$$x^f = x + \triangle^f x$$

$$s^f = s + \triangle^f s$$

where $\triangle^f x$ and $\triangle^f s$ are search directions calculated by solving (5.1). Our aim is to show that (x^f, s^f) is strictly feasible and satisfies $\delta(x^f, s^f; \mu^+) \leq \tau$ where $\mu^+ = (1-\theta)\mu$ for certain choice of θ and τ. We use transformations in (5.2) and equation (5.1b) to show the following:

$$\begin{aligned} x^f s^f &= (x + \triangle^f x)(s + \triangle^f s) \\ &= xs + x\triangle^f s + s\triangle^f x + \triangle^f x \triangle^f s \\ (from\ (5.1b)) \rightarrow &= (1-\theta)\mu e + \triangle^f x \triangle^f s \\ &= (1-\theta)\mu e + \frac{xs}{v^2} d_x^f d_s^f \\ (v^2 = \frac{xs}{\mu} \rightarrow) &= (1-\theta)\mu e + \mu d_x^f d_s^f \\ &= \mu[(1-\theta)e + d_x^f d_s^f]. \end{aligned} \tag{5.9}$$

The following lemma shows the feasibility of the new iterates.

Lemma 5.1. *The iterates* (x^f, s^f) *are strictly feasible if and only if* $(1-\theta)e+d_x^f d_s^f > 0$.

Proof. $(\Rightarrow)$: Let (x^f, s^f) be strictly feasible. Then, $x^f > 0$ and $s^f > 0$. Therefore,

$$\mu[(1-\theta)e + d_x^f d_s^f] > 0.$$

Thus,

$$(1-\theta)e + d_x^f d_s^f > 0.$$

$(\Leftarrow)$: Let introduce a step length $\alpha \in [0,1]$, and define

$$x^\alpha = x + \alpha \triangle^f x, \quad s^\alpha = s + \alpha \triangle^f s.$$

Then, when $\alpha = 0$, we have $x^0 = x$, $s^0 = s$ and when $\alpha = 1$, we have $x^1 = x^f$, $s^1 = s^f$. Thus, $x^0 s^0 > 0$. Then,

$$\begin{aligned}
x^\alpha s^\alpha &= (x + \alpha \triangle^f x)(s + \alpha \triangle^f s) \\
&= xs + x\alpha \triangle^f s + s\alpha \triangle^f x + \alpha^2 \triangle^f x \triangle^f s \\
&= xs + \alpha(x \triangle^f s + s \triangle^f x) + \alpha^2 \triangle^f x \triangle^f s \\
(by\ (5.1b)) &= xs + \alpha((1-\theta)\mu e - xs) + \alpha^2 \triangle^f x \triangle^f s \\
&= xs + \alpha(1-\theta)\mu e - \alpha xs + \alpha^2 \triangle^f x \triangle^f s \\
&= xs(1-\alpha) + \alpha(1-\theta)\mu e + \alpha^2 \triangle^f x \triangle^f s \\
(Since\ \triangle^f x \triangle^f s = \mu d_x^f d_s^f) &= xs(1-\theta) + \alpha(1-\theta)\mu e + \alpha^2 \mu d_x^f d_s^f \\
(Since\, xs = v^2\mu) &= v^2\mu(1-\alpha) + \alpha(1-\theta)\mu e + \alpha^2 \mu d_x^f d_s^f \\
&= \mu[v^2(1-\alpha) + \alpha(1-\theta)e + \alpha^2 d_x^f d_s^f]
\end{aligned}$$

Now, let $(1-\theta)e + d_x^f d_s^f > 0$. Then $d_x^f d_s^f > -(1-\theta)e$. So, we have

$$\begin{aligned} x^\alpha s^\alpha &> \mu[v^2(1-\alpha) + \alpha(1-\theta)e - \alpha^2(1-\theta)e] \\ &= \mu[v^2(1-\alpha) + \alpha e - \alpha\theta e - \alpha^2 e + \alpha^2\theta e] \\ &= \mu[v^2(1-\alpha) + \alpha e(1-\alpha) - \alpha\theta e(1-\alpha)] \\ &= \mu(1-\alpha)[v^2 + \alpha e - \alpha\theta e] \\ &= \mu(1-\alpha)[v^2 + \alpha e(1-\theta)] \end{aligned}$$

Since $\mu > 0$, $\alpha \in [0,1]$, $v^2 > 0$ and $e > 0$, then $\mu(1-\alpha)[v^2 + \alpha e(1-\theta)] \geq 0$ implying that $x^\alpha s^\alpha > 0$ for $\alpha \in [0,1]$. Thus, none of the entries of x^α and s^α vanishes for $0 \leq \alpha \leq 1$. Since x^0, s^0 are positive, then x^α, s^α which depend on α linearly are also positive for $0 \leq \alpha \leq 1$. Therefore, $x^1 = x^f$ and $s^1 = s^f$ must be positive. □

We have the following corollary from the Lemma 5.1.

Corollary 5.2. *The iterates (x^f, s^f) are strictly feasible if $||d_x^f d_s^f||_\infty < 1-\theta$.*

Proof. By Lemma 5.1, the iterates x^f, s^f are strictly feasible if and only if $(1-\theta)e + d_x^f d_s^f > 0$. Then,

$$(1-\theta) + d_{x_i}^f d_{s_i}^f > 0 \ \ for \ \ i = 1, 2, ..., n$$

$$d_{x_i}^f d_{s_i}^f > -(1-\theta)$$

By the definition of ∞-norm i.e., $||d_x^f d_s^f||_\infty = \max\{|d_{x_i}^f d_{s_i}^f| : i = 1, 2, ..., n\}$, and from the assumption, we have that $|d_{x_i}^f d_{s_i}^f| < 1-\theta$. Equivalently, we have

$$-(1-\theta) < d_{x_i}^f d_{s_i}^f < (1-\theta)$$

which implies that

$$d_{x_i}^f d_{s_i}^f + (1-\theta) > 0 \quad or \quad d_x^f d_s^f + (1-\theta)e > 0.$$

Thus, by Lemma 5.1, (x^f, s^f) are strictly feasible. □

Our aim is to find an upper bound for $\delta(x^f, s^f; \mu^+)$ such that proximity of the new iterates (x^f, s^f) is always less than the threshold τ, for each iteration. In line with this purpose, we use the notation $\omega(v) = \frac{1}{2}(||d_x^f||^2 + ||d_s^f||^2)$. Then, it is a norm fact that

$$||d_x^f d_s^f||_\infty \leq ||d_x^f d_s^f|| \leq ||d_x^f||\,||d_s^f|| \leq \frac{1}{2}(||d_x^f||^2 + ||d_s^f||^2) = \omega(v) \tag{5.10}$$

Corollary 5.3. *If $\omega(v) < (1-\theta)$, then (x^f, s^f) are strictly feasible.*

Proof. Due to (5.10), $\omega(v) < (1-\theta)$ implies that $||d_x^f d_s^f||_\infty < (1-\theta)$. Then, by Corollary 5.2, (x^f, s^f) are strictly feasible. □

Let us recall that the proximity of new iterates (x^f, s^f) is given by

$$\delta(x^f, s^f; \mu^+) = \delta(v^f) = \frac{1}{2}||v^f - (v^f)^{-1}|| \quad where \quad v^f = \sqrt{\frac{x^f s^f}{\mu^+}}. \tag{5.11}$$

Also, we use the function ξ defined by

$$\xi(t) = \frac{1+t}{1-\theta} + \frac{1-\theta}{1+t} - 2 = \frac{(\theta+t)^2}{(1-\theta)(1+t)} \geq 0, \;\; t > -1. \tag{5.12}$$

The following Lemma proved by C.Roos in 2015 [1] leads us to find an upper bound for $\delta(x^f, s^f; \mu^+) = \delta(v^f)$. Our improvement is based on this lemma.

Lemma 5.4. *Let $a, b \in \mathbb{R}^n$, $r \in [0, 1)$ and $f(a, b) = \sum_{i=1}^n \xi(a_i b_i)$. If $||a||^2 + ||b||^2 \leq 2r^2$, then*

$$f(a, b) \leq (n-1)\xi(0) + \max\{\xi(r^2), \xi(-r^2)\}.$$

Proof. For proof of the Lemma, see Appendix A in [1]. □

We use the Lemma 5.4 to prove the following Lemma which denotes the upper bound for $\delta(v^f)$.

Lemma 5.5. *If $\omega(v) < 1-\theta$, then*

$$4\delta(v^f)^2 \leq (n-1)\xi(0) + \max\{\xi(\omega(v)), \xi(-\omega(v))\}.$$

Proof. By using (5.9) and (5.11), $(v^f)^2$ can be written as:

$$(v^f)^2 = \frac{x^f s^f}{\mu^+} = \frac{(1-\theta)e + d_x^f d_s^f}{1-\theta}\frac{\mu}{\mu} = e + \frac{d_x^f d_s^f}{1-\theta}.$$

Since $4\delta(v^f)^2 = ||v^f - (v^f)^{-1}||^2$, then

$$\begin{aligned}
4\delta(v^f)^2 &= \sum_{i=1}^{n}\left(v_i^f - \frac{1}{v_i^f}\right)^2 \\
&= \sum_{i=1}^{n}\left[(v_i^f)^2 - 2 + \left(\frac{1}{v_i^f}\right)^2\right] \\
&= \sum_{i=1}^{n}(v_i^f)^2 + \sum_{i=1}^{n}\left(\frac{1}{v_i^f}\right)^2 - \sum_{i=1}^{n}2 \\
&= -2n + \sum_{i=1}^{n}\left(\frac{(1-\theta) + d_{x_i}^f d_{s_i}^f}{1-\theta}\right) + \sum_{i=1}^{n}\left(\frac{1-\theta}{(1-\theta) + d_{x_i}^f d_{s_i}^f}\right) \\
&= \sum_{i=1}^{n}\xi(d_{x_i}^f d_{s_i}^f - \theta).
\end{aligned}$$

As it can be seen from the Figure 5.1, $\xi(t)$ function is an increasing function when $t > -1$. Since, $\theta \in [0, 1)$, we have

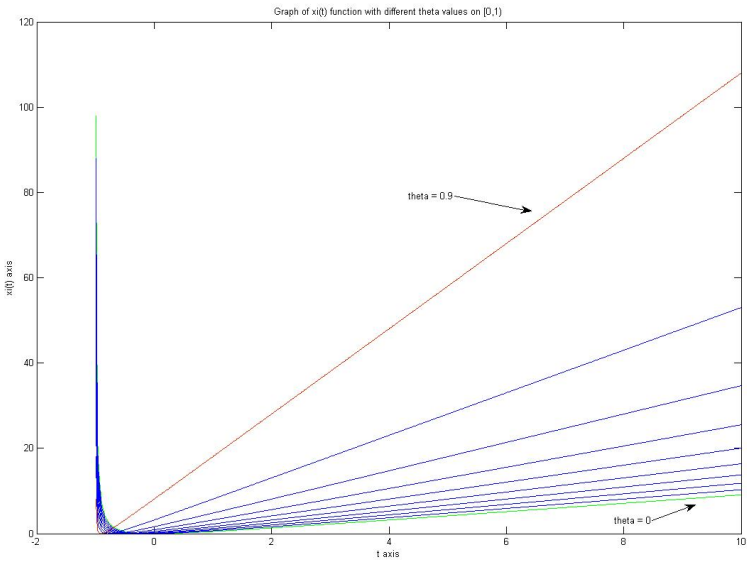

Figure 5.1: Graph of $\xi(t)$ for different $\theta \in [0, 1)$

$$\sum_{i=1}^{n} \xi(d_{x_i}^f d_{s_i}^f - \theta) \leq \sum_{i=1}^{n} \xi(d_{x_i}^f d_{s_i}^f).$$

Then, by Lemma 5.4, we have the result which is

$$4\delta(v^f)^2 \leq (n-1)\xi(0) + \max\{\xi(\omega(v)), \xi(-\omega(v))\}.$$

□

Upper bound for $\omega(v)$:

To bound $\delta(v^f)$, we search for an upper bound for $\omega(v)$. In order to obtain an upper bound for $\omega(v)$, first let us recall that

$$\omega(v) = \frac{1}{2}(||d_x^f||^2 + ||d_s^f||^2).$$

To bound $\omega(v)$, we need to find an upper bound for $||d_x^f||^2 + ||d_s^f||^2$. Thus, we need to use following lemma which was proved in [21].

Lemma 5.6. *We are given the following system*

$$\widetilde{M}u - z = \tilde{a} \tag{5.13a}$$

$$u + z = \tilde{b} \tag{5.13b}$$

Then, the following hold

(1) $Du = (1 + DMD)^{-1}(a + b), \;\; Dz = (b - Du)$

(2) $||Du|| \leq ||a + b||$

(3) $||Du||^2 + ||Dz||^2 \leq ||b||^2 + 2||a + b||\,||a||.$

where $D = S^{-1/2}X^{1/2}, \;\; b = D\tilde{b}, \;\; a = D\tilde{a}$ *and* $\widetilde{M} = DMD.$

Proof. First, we start with multiplying (5.13a) and (5.13b) from left with D. Then we have

$$DMDDu - Dz = a \tag{5.14a}$$

$$Du + Dz = b \tag{5.14b}$$

Then, by adding (5.14a) and (5.14b), we obtain equation (1). Since the matrix $I + DMD$ is positive definite, then inequality (2) follows. Finally, from (5.14) and by using Cauchy-Schwartz inequality, inequality, (2), and positive semi-definiteness of DMD, we have

$$\begin{aligned} ||Du||^2 + ||Dz||^2 &= ||Du + Dz||^2 - 2(Du)^T Dz \\ &= ||b||^2 - 2(Du)^T (DMDDu - a) \\ &= ||b||^2 - 2(Du)^T DMDDu + 2(Du)^T a \\ &\leq ||b||^2 + 2||Du||\,||a|| \\ &\leq ||b||^2 + 2||a + b||\,||a||. \end{aligned}$$

□

We apply Lemma 5.6 to the system (5.8) which is

$$\begin{aligned} \widetilde{M} d_x^f - d_s^f &= D\mu^{-1/2}\theta\nu r^0, \\ d_x^f + d_s^f &= v^{-1}(1-\theta) - v. \end{aligned} \tag{5.15}$$

Let us call

$$\begin{aligned} a &= D(\theta\nu D r^0 \mu^{-1/2}) = D^2(\theta\nu r^0 \mu^{-1/2}) \\ b &= D((1-\theta)v^{-1} - v) \\ u &= d_x^f \\ z &= d_s^f. \end{aligned}$$

Then by Lemma 5.6 part (3), we have

$$\begin{aligned} ||Dd_x^f||^2 + ||Dd_s^f||^2 \leq\ & ||D((1-\theta)v^{-1} - v)||^2 + 2||D^2(\theta\nu r^0 \mu^{-1/2}) \\ & + D((1-\theta)v^{-1} - v)||\,||D^2(\theta\nu r^0 \mu^{-1/2})|| \end{aligned} \tag{5.16}$$

Then, we apply the following norm facts to (5.16).

$$\begin{aligned}
&(i) \quad ||Dd_x^f|| \leq ||D||\,||d_x^f|| \ , \ ||Dd_s^f|| \leq ||D||\,||d_s^f|| \\
&(ii) \quad ||D^2(\theta\nu r^0\mu^{-1/2})|| \leq ||D||^2||\theta\nu r^0||\mu^{-1/2} \\
&(iii) \quad ||D((1-\theta)v^{-1}-v)|| \leq ||D||\,||(1-\theta)v^{-1}-v||
\end{aligned}$$

where $||D||$ represents a matrix norm. By using $(i), (ii)$ and (iii), we have that

$$\begin{aligned}
||D||^2\left(||d_x^f||^2+||d_s^f||^2\right) \leq ||D||^2||((1-\theta)v^{-1}-v)||^2 + 2||D||\,||D(\theta\nu r^0\mu^{-1/2}) \\
+ ((1-\theta)v^{-1}-v)||\,||D||\,||D(\theta\nu r^0\mu^{-1/2})||
\end{aligned} \tag{5.17}$$

After cancelling $||D||^2$ from both sides of (5.17), we have

$$\begin{aligned}
||d_x^f||^2+||d_s^f||^2 \leq ||(1-\theta)v^{-1}-v||^2 \\
+ 2\left(||\theta\nu r^0\mu^{-1/2}|| + ||(1-\theta)v^{-1}-v||\right)||D(\theta\nu r^0\mu^{-1/2})||
\end{aligned} \tag{5.18}$$

As we can see in (5.18), to find an upper bound for $||d_x^f||^2+||d_s^f||^2$ we need to find upper bounds for $||D(\theta\nu r^0\mu^{-1/2})||$ and $||(1-\theta)v^{-1}-v||$ respectively.

a) Upper bound for $||D(\theta\nu r^0\mu^{-1/2})||$:

To find an upper bound for $||D(\theta\nu r^0\mu^{-1/2})||$, we reduce it as follows:

$$\begin{aligned}
||D(\theta\nu r^0\mu^{-1/2})|| &= \frac{\theta\nu}{\sqrt{\mu}}||Dr^0|| \\
&= \frac{\theta\nu}{\sqrt{\mu}}||X^{1/2}S^{-1/2}r^0|| \\
&= \frac{\theta\nu}{\sqrt{\mu}}||\sqrt{\frac{x}{s}}r^0|| \\
(Since\ \nu=\mu/\mu^0) &\leq \frac{\theta}{\sqrt{\mu}}\frac{\mu}{\mu^0}||\sqrt{\frac{x}{s}}r^0||_1 \\
&= \frac{\theta}{\mu^0}||\sqrt{\frac{\mu x}{s}}r^0||_1 \\
&= \frac{\theta}{\mu^0}||\sqrt{\frac{\mu}{xs}}xr^0||_1 \rightarrow \left|\frac{1}{v_i}x_ir_i^0\right| \leq \frac{1}{v_{\min}}|x_ir_i^0| \leq \frac{1}{v_{\min}}|x_i|\,|r_i^0|
\end{aligned}$$

$$
\begin{aligned}
(Since\ \sqrt{\frac{\mu}{xs}} = \frac{1}{v})\ \ ||D(\theta\nu r^0\mu^{-1/2})|| &\leq \frac{\theta}{\mu^0}\frac{1}{v_{\min}}||xr^0||_1 \\
&\leq \frac{\theta}{\mu^0}\frac{1}{v_{\min}}||x||_1||r^0||_\infty.
\end{aligned} \tag{5.19}
$$

Since our initial assumptions are that $x^0 = \gamma_p e$ and $s^0 = \gamma_d e$ so that $\mu^0 = \gamma_p\gamma_d$, then we can choose γ_p and γ_d such that $||x^0||_\infty \leq \gamma_p$ and $||s^0||_\infty \leq \gamma_d$. Then, we have that

$$
\begin{aligned}
r^0 &= s^0 - Mx^0 - q \\
&= \gamma_d e - \gamma_p Me - q \\
&= \gamma_d\left(e - \frac{\gamma_p}{\gamma_d}Me - \frac{1}{\gamma_d}q\right).
\end{aligned}
$$

Thus, we can bound $||r^0||_\infty$ as follows

$$
\begin{aligned}
||r^0||_\infty &= \gamma_d||e - \frac{\gamma_p}{\gamma_d}Me - \frac{1}{\gamma_d}q||_\infty \\
&\leq \gamma_d\left(1 - \frac{\gamma_p}{\gamma_d}||Me||_\infty - \frac{1}{\gamma_d}||q||_\infty\right).
\end{aligned}
$$

By assuming $\max\{||Me||_\infty,\ ||q||_\infty\} \leq \gamma_d$, we have that

$$
\begin{aligned}
||r^0||_\infty &\leq \gamma_d(1+1+1) \\
&= 3\gamma_d
\end{aligned}
$$

Then, finally, by substituting upper bound for $||r^0||$ into (5.19), we have

$$
\begin{aligned}
||D(\theta\nu r^0\mu^{-1/2})|| &\leq \frac{\theta}{\mu^0}\frac{1}{v_{\min}}||x||_1 3\gamma_d \\
&= \frac{\theta}{\gamma_d\gamma_p}\frac{1}{v_{\min}}3\gamma_d||x||_1 \\
&= \frac{3\theta}{\gamma_p}\frac{||x||_1}{v_{\min}}.
\end{aligned} \tag{5.20}
$$

b) Upper bound for $||(1-\theta)v^{-1}-v||$:

We find an upper bound for $||(1-\theta)v^{-1}-v||$ by reducing it as follows:

$$\begin{aligned}
||(1-\theta)v^{-1}-v||^2 &= ||(1-\theta)v^{-1}||^2 - 2(1-\theta)(v^{-1})^T v + ||v||^2 \\
&= (1-\theta)^2||v^{-1}||^2 - 2(1-\theta)n + ||v||^2 \\
&= (1-\theta)^2||v^{-1}||^2 - 2n + 2\theta n + ||v||^2 \\
&\leq ||v^{-1}||^2 - 2n + ||v||^2 + 2\theta n \\
&= ||v^{-1}-v||^2 + 2\theta n \\
&= 4\delta(v)^2 + 2\theta n.
\end{aligned}$$

Hence, we have

$$||(1-\theta)v^{-1}-v|| \leq \sqrt{4\delta(v)^2 + 2\theta n} \tag{5.21}$$

Now, we substitute (5.20) and (5.21) into (5.18). Then, we have

$$||d_x^f||^2 + ||d_s^f||^2 \leq \left(4\delta(v)^2 + 2\theta n\right) + 2\left(\frac{3\theta}{\gamma_p}\frac{||x||_1}{v_{\min}} + \sqrt{4\delta(v)^2 + 2\theta n}\right)\frac{3\theta}{\gamma_p}\frac{||x||_1}{v_{\min}} \tag{5.22}$$

To find a more strict upper bound for $||d_x^f||^2 + ||d_s^f||^2$, we need to find an upper bound and a lower bound for $||x||_1$ and $v_{\min}$ respectively. We achieve this goal in the following lemma.

Lemma 5.7. *Let*

$$q(\delta) = \delta + \sqrt{\delta^2 + 1}.$$

Then, the following hold,

$$\begin{aligned}
&(i) \quad q^{-1}(\delta) \leq v_i \leq q(\delta) \\
&(ii) \quad ||x||_1 \leq (2 + q(\delta))n\gamma_p, \quad ||s||_1 \leq (2 + q(\delta))n\gamma_d
\end{aligned}$$

Proof. Since, for each i, v_i is positive, from (4.6) we have

$$-2\delta v_i \leq 1 - v_i^2 \leq 2\delta v_i$$

implying that

$$v_i^2 - 2\delta v_i - 1 \leq 0 \leq v_i^2 + 2\delta v_i - 1.$$

By adding and subtracting δ^2, we can rewrite the above inequality as:

$$(v_i - \delta)^2 - 1 - \delta^2 \leq 0 \leq (v_i + \delta)^2 - 1 + \delta^2$$

Then, we obtain

$$(v_i - \delta)^2 \leq 1 + \delta^2 \leq (v_i + \delta)^2$$

which implies that

$$v_i - \delta \leq |v_i - \delta| \leq \sqrt{1 + \delta^2} \leq v_i + \delta.$$

Hence, we get

$$-\delta + \sqrt{1 + \delta^2} \leq v_i \leq \delta + \sqrt{1 + \delta^2}$$

Equivalently, since $\delta + \sqrt{1 + \delta^2} = q(\delta)$ we have

$$q^{-1}(\delta) \leq v_i \leq q(\delta)$$

proving (i). To prove (ii), we know that x, s, x^* and s^* are positive. Thus, $s^T x^* + x^T s^*$ is positive. Hence,

$$(s^0)^T x + (x^0)^T s \leq v(x^0)^T s^0 + \frac{x^T s}{v} + (1 - v)((s^0)^T x^* + (x^0)^T s^*)$$

By substituting $x^0 = \gamma_p e, s^0 = \gamma_d e, ||x^*||_\infty \leq \gamma_p$ and $||s^*||_\infty \leq \gamma_d$, we have

$$(s^0)^T x^* + (x^0)^T s^* \leq \gamma_p(e^T s^0) + \gamma_d(e^T x^0) = 2n\gamma_p\gamma_d.$$

Also, $(x^0)^T s^0 = n\gamma_p\gamma_d$. Thus, we have

$$\begin{aligned}
(s^0)^T x + (x^0)^T s &\leq \frac{x^T s}{v} + 2n\gamma_p\gamma_d - vn\gamma_p\gamma_d \\
&\leq \frac{x^T s}{v} + 2n\gamma_p\gamma_d \\
&= \frac{\mu(e^T v^2)}{v} + 2n\gamma_p\gamma_d \\
&= \gamma_p\gamma_d(e^T v^2) + 2n\gamma_p\gamma_d,
\end{aligned}$$

Since $v = \frac{\mu}{\mu^0}$ and $\mu^0 = \gamma_p\gamma_d$, the last inequality follows. Moreover,

$$\begin{aligned}
\gamma_p\gamma_d(e^T v^2) + 2n\gamma_p\gamma_d &= \gamma_p\gamma_d(e^T v^2 + 2n) \\
&= \gamma_p\gamma_d(\sum_{i=1}^{n} v_i^2 + 2n) \\
&\leq \gamma_p\gamma_d(\sum_{i=1}^{n} q^2(\delta) + 2n) \\
&= \gamma_p\gamma_d(q^2(\delta)\sum_{i=1}^{n} 1 + 2n) \\
&= \gamma_p\gamma_d(q^2(\delta)n + 2n) \\
&= \gamma_p\gamma_d n(q^2(\delta) + 2).
\end{aligned}$$

Therefore $(s^0)^T x + (x^0)^T s \leq \gamma_p\gamma_d n(q^2(\delta) + 2)$ from which we obtain that

$$\begin{aligned}
(s^0)^T x &\leq \gamma_p\gamma_d n(q^2(\delta) + 2) \\
(x^0)^T s &\leq \gamma_p\gamma_d n(q^2(\delta) + 2).
\end{aligned}$$

By substituting $x^0 = \gamma_p e$ and $s^0 = \gamma_d e$, we have

$$\begin{aligned}
||x||_1 &\leq \gamma_p n(q^2(\delta) + 2) \\
||s||_1 &\leq \gamma_d n(q^2(\delta) + 2)
\end{aligned}$$

which proves part (ii). □

By applying Lemma 5.7, (5.22) becomes

$$\begin{aligned}
||d_x^f||^2 + ||d_s^f||^2 &\leq (4\delta^2 + 2\theta n) + \frac{18\theta^2}{\gamma_p^2}\frac{||x||_1^2}{v_{\min}^2} + \frac{6\theta}{\gamma_p}\sqrt{4\delta^2 + 2\theta n}(2 + q(\delta))q(\delta) \\
&= (4\delta^2 + 2\theta n) + 18\theta^2 n^2(2 + q(\delta))^2 q^2(\delta) \\
&\quad + 6\theta n\sqrt{4\delta^2 + 2\theta n}(2 + q(\delta))q(\delta).
\end{aligned} \tag{5.23}$$

Finally, we have the upper bound for $\omega(v)$ by substituting (5.23) into

$$\omega(v) = \frac{1}{2}\left(||d_x^f||^2 + ||d_s^f||^2\right)$$

which is:

$$\begin{aligned}\omega(v) &\leq \frac{1}{2}\left[2(2\delta^2+\theta n)+18\theta^2 n^2(2+q(\delta))^2q^2(\delta)+6\theta n\sqrt{4\delta^2+2\theta n}(2+q(\delta))q(\delta)\right]\\ &= (2\delta^2+\theta n)+9\theta^2 n^2(2+q(\delta))^2q^2(\delta)+3\theta n\sqrt{4\delta^2+2\theta n}(2+q(\delta))q(\delta)\end{aligned} \tag{5.24}$$

where

$$q(\delta)=\delta+\sqrt{\delta^2+1}.$$

From Lemma 5.5, we have

$$4\delta(v^f)^2 \leq (n-1)\xi(0)+\max\{\xi(\omega(v)),\xi(-\omega(v))\}.$$

Thus, the upper bound for $\delta(v^f)$ is:

$$\delta(v^f) \leq \frac{1}{2}\sqrt{(n-1)\xi(0)+\max\{\xi(\omega(v)),\xi(-\omega(v))\}} \leq \tau \tag{5.25}$$

where

$$\omega(v) \leq (2\delta^2+\theta n)+9\theta^2 n^2(2+q(\delta))^2q^2(\delta)+3\theta n\sqrt{4\delta^2+2\theta n}(2+q(\delta))q(\delta)$$

and

$$q(\delta)=\delta+\sqrt{\delta^2+1}.$$

Now, we need to find a specific τ and θ such that (5.25) is satisfied. For this purpose, we have written a MATLAB code to show that the specific τ and θ that we have found satisfies the inequality (5.25). The result is given in the following table.

θ	τ	$\delta(v^f)\,:\,n=2$	$n=100$	$n=1000$
$\frac{1}{39+n}$	$\frac{1}{5}$	0.1806	0.0751	0.0468
$\frac{1}{40+n}$	$\frac{1}{4}$	0.2483	0.1060	0.0721
$\frac{1}{53+n}$	$\frac{1}{3}$	0.3272	0.1787	0.1329
$\frac{1}{170+n}$	$\frac{1}{2}$	0.4981	0.4371	0.3708

Table 5.1: Proximity of new iterates to μ-center for certain choice of τ and θ

As it can be seen from Table 5.1, when τ increases corresponding θ decreases. Bigger τ means that we increase the range of our τ-neighbourhood of central path. This helps to decrease number of iterations because for bigger τ, we may perform less Newton iteration to satisfy proximity condition that is we can take bigger step. On the other hand, for bigger τ, we have smaller θ. Since $\mu^+ = (1-\theta)\mu$ and $\nu^+ = (1-\theta)\nu$, if θ is small, then we reduce μ and ν slowly. Thus, we reach the ϵ-solution by performing more iterations. Another observation is that, as seen in the table, when dimension of the problem is increased, the new iterates becomes more closer to the central path. As a result, we have the following theorem.

Theorem 5.8. *Let δ and τ be one of the pairs in the Table 5.1 and (x, s) be a current iterate with $\delta(x, s; \mu) \leq \tau$. Then, after the feasibility step the new iterate (x^f, s^f) is strictly feasible and satisfies (5.25), that is $\delta(x^f, s^f; \mu^+) \leq \tau$.*

Calculation of the number of iterations

In this part, we calculate required number of iterations of full-Newton step $IIPM$ algorithm to obtain ϵ-solution. Before that we need to prove following useful results and the lemma which we use in the calculation of the number of iterations.

(i) : The first result is that

$$\triangle x^T \triangle s \leq \mu\delta^2.$$

Proof.

$$\begin{aligned}
\triangle x^T \triangle s &= (xv^{-1}d_x)^T(sv^{-1}d_s) \\
&= \left(\sqrt{\frac{x\mu}{s}}d_x\right)^T \left(\sqrt{\frac{s\mu}{x}}d_s\right) \\
&= \mu \left(\sqrt{\frac{x}{s}}d_x\right)^T \left(\sqrt{\frac{s}{x}}d_s\right) \\
&= \mu d_x^T d_s \\
&\leq \mu\delta^2.
\end{aligned} \tag{5.26}$$

□

(ii) : The second result is that

$$x^f s^f = \mu e + \triangle x \triangle s.$$

Proof.

$$\begin{aligned} x^f s^f &= (x + \triangle x)(s + \triangle s) \\ &= xs + x \triangle s + s \triangle x + \triangle x \triangle s \\ &= xs + (\mu e - xs) + \triangle x \triangle s \\ &= \mu e + \triangle x \triangle s. \end{aligned} \tag{5.27}$$

□

Lemma 5.9. *Let x^f and s^f be new iterates after a feasibility step. Then*

$$(x^f)^T s^f \leq \mu(n + \delta^2).$$

Proof.

$$\begin{aligned} (x^f)^T s^f &= e^T (x^f s^f) \\ (by\ (5.27)) &= e^T (\mu e + \triangle x \triangle s) \\ &= \mu e^T e + e^T \triangle x \triangle s \\ &= \mu n + \triangle x^T \triangle s \\ (by\ (5.26)) &\leq \mu n + \mu \delta^2 \\ &= \mu(n + \delta^2). \end{aligned} \tag{5.28}$$

□

Now, we use the Lemma 5.9 in the proof of the following theorem that shows the required iteration number for full Newton step Infeasible IPM algorithm to find ϵ-solution.

Theorem 5.10. *Let* $\theta = \frac{1}{39+n}$, $\tau = \frac{1}{5}$ *and* $\mu^0 = \frac{(x^0)^T s^0}{n}$. *Then, full Newton step Infeasible IPM algorithm requires at most* $(39+n)\log\frac{51(x^0)^T s^0}{50\epsilon}$ *iterations to reach the* ϵ*-solution of* $LCP(M,q)$ *or equivalently* $O(n\log\frac{n}{\epsilon})$.

Proof. Let x_k and s_k be the k-th iterates of the algorithm. Then,

$$\begin{aligned}
x_k^T s_k &\leq \mu_k(n+\delta^2) \\
&\leq \mu_k(n+\frac{1}{25}) \\
&= (1-\theta)^k\mu^0(n+\frac{1}{25}) \\
&= (1-\theta)^k\frac{(x^0)^T s^0}{n}(n+\frac{1}{25}) \\
&\leq (1-\theta)^k\frac{(x^0)^T s^0}{n}(n+\frac{1}{50}n)\ (Since\ n\geq 2) \\
&= (1-\theta)^k\frac{(x^0)^T s^0}{n}\frac{51}{50}n \\
&= (1-\theta)^k(x^0)^T s^0\frac{51}{50} \\
&\leq \epsilon.
\end{aligned}$$

Then by taking logarithm of both sides, we have

$$\begin{aligned}
\log\left[(1-\theta)^k(x^0)^T s^0\frac{51}{50}\right] &\leq \log\epsilon \\
\log(1-\theta)^k + \log(x^0)^T s^0 + \log\frac{51}{50} &\leq \log\epsilon \\
\log(1-\theta)^k &\leq \log\epsilon - \log(x^0)^T s^0 - \log\frac{51}{50} \\
k\log(1-\theta) &\leq \log\frac{50\epsilon}{51(x^0)^T s^0} \\
-k\log(1-\theta) &\geq -\log\frac{50\epsilon}{51(x^0)^T s^0}
\end{aligned}$$

Since $-\log(1-\theta) \geq \theta$, then

$$k\theta \geq \log \frac{51(x^0)^T s^0}{50\epsilon}$$
$$k \geq \frac{1}{\theta} \log \frac{51(x^0)^T s^0}{50\epsilon}$$
$$k \geq (39+n) \log \frac{51(x^0)^T s^0}{50\epsilon}$$

which concludes the number of iteration that is required to get ϵ-solution. □

Similarly, we can calculate the required number of iterations for other θ and τ values in the Table 5.1. The results for these values are shown in the following table.

θ	τ	number of iterations
$\frac{1}{40+n}$	$\frac{1}{4}$	$(40+n) \log \frac{33(x^0)^T s^0}{32\epsilon}$
$\frac{1}{53+n}$	$\frac{1}{3}$	$(53+n) \log \frac{19(x^0)^T s^0}{18\epsilon}$
$\frac{1}{170+n}$	$\frac{1}{2}$	$(170+n) \log \frac{9(x^0)^T s^0}{8\epsilon}$

Table 5.2: Required number of iterations for different τ and θ values

As a result, after each feasibility step, improved algorithm guarantees that the new iterates (x^f, s^f) satisfy (5.25). Moreover, the new iterates are close enough to μ-center after an iteration. Thus, we eliminate the centering steps. Furthermore, upper bound on the number of iterations to obtain ϵ-solution is $O(n \log \frac{n}{\epsilon})$. In the next Chapter, we show our numerical examples and results.

CHAPTER 6

NUMERICAL RESULTS

In this Chapter, we show our numerical results of full Newton step infeasible IPM algorithm for random generated $LCPs$. Then, we compare improved algorithm with the old version.

We have implemented the pseudocode in Section 4.2.1 via MATLAB. The codes are included in Appendix A. Then, we generated random positive semi-definite matrix M with various dimensions. Also, our initial starting vector x and constant vector q are generated randomly. In the rest of this section, we give a complete example for 2×2 matrix M and some tables showing results for other dimensions.

Example 1:

For this example, our randomly generated inputs are:

$$M = \begin{bmatrix} 0.4512 & 0.6328 \\ 0.6328 & 0.9995 \end{bmatrix}, x = \begin{bmatrix} 0.0791 \\ 0.5094 \end{bmatrix} \ and \ q = \begin{bmatrix} 0.5441 \\ 0.6990 \end{bmatrix}.$$

As it can be seen that all the eigenvalues $\{0.0357, 1.4149\}$ of matrix M are positive. Thus M is positive definite matrix. So, the $LCP(M, q)$ generated with these inputs have a unique solution. Also, we choose $s^0 = Mx^0 + q$. Thus, for this example our initial pair (x^0, s^0) is feasible.

After running the code with these inputs and with $\theta = \frac{1}{39+n} = \frac{1}{39+2} = 0.0244, \tau = \frac{1}{5} = 0.2$, we reach the ϵ-solution with 360 iterations where $\epsilon = 10^{-4}$. Theoretically, the expected number of iteration by Theorem 5.10 is $(39+2)\log\frac{51(x^0)^T s^0}{50\epsilon} = 364.5256$. Table 6.1 includes some iteration steps. In this table, it can be seen that the complementarity condition $x^T s$ is converging to zero at each iteration. Furthermore, it can be observed that at each iteration, the new iterates are close enough to central path. More specifically, at each iteration, the new iterates (x^f, s^f) satisfy $\delta(x^f, s^f; \mu^+) \leq \tau$.

iteration	$x^T s$	μ	ν	$\delta(v^f)$
1 :	0.70044013	0.34745664	0.97560976	0.00891923
2 :	0.67811788	0.33898208	0.95181440	0.00017605
3 :	0.66151292	0.33071423	0.92859941	0.00009034
⋮	⋮	⋮	⋮	⋮
358 :	0.00010316	0.00005158	0.00014483	0.00000008
359 :	0.00010064	0.00005032	0.00014130	0.00000008
360 :	0.00009819	0.00004909	0.00013785	0.00000007

Table 6.1: Changes in $x^T s, \mu, \nu$ and $\delta(v^f)$ for the example

The solution vectors of this specific $LCP(M, q)$ are:

$$x = 1.0e-004 \times \begin{bmatrix} 0.9022 \\ 0.7022 \end{bmatrix} \ \ and \ \ s = \begin{bmatrix} 0.5442 \\ 0.6991 \end{bmatrix}.$$

We also solved this specific example for different θ and τ values given in the Table 5.2. Table 6.2 shows the required number of iterations for those different values. As it is seen in Table 6.1, different θ values requires different number of iterations. As seen in Theorem 5.10, in theory expected number of iterations depend on θ. Thus, for smaller θ, the algorithm performs more iterations. Moreover, since we reduce μ and ν by using θ i.e., $\mu^+ = (1-\theta)\mu$ and $\nu^+ = (1-\theta)\nu$, hence smaller θ increases the number of iterations that is required both in practice and in theory. As a result, the selection of θ is basically depends on the dimension of the problem. If you consider the specific θ values in Table 5.1, it can be observed that for large dimensional problems, this algorithm is much more efficient then the old version. We show how this improved version of the algorithm is efficient in the next Section.

θ	τ	ϵ	expected num. of iter.	computed. num. of iter.
$\frac{1}{39+n}$	$\frac{1}{5}$	10^{-4}	364	360
$\frac{1}{39+n}$	$\frac{1}{5}$	10^{-6}	553	546
$\frac{1}{40+n}$	$\frac{1}{4}$	10^{-4}	374	369
$\frac{1}{40+n}$	$\frac{1}{4}$	10^{-6}	567	560
$\frac{1}{53+n}$	$\frac{1}{3}$	10^{-4}	490	484
$\frac{1}{53+n}$	$\frac{1}{3}$	10^{-6}	744	735
$\frac{1}{170+n}$	$\frac{1}{2}$	10^{-4}	1546	1522
$\frac{1}{170+n}$	$\frac{1}{2}$	10^{-6}	2338	2312

Table 6.2: Required number of iterations for different θ, τ and ϵ values

Example 2: In this example, we solve the same example as in Section 3.3 where it was solved by Lemke's Method and we compare the results. Let us recall the inputs from Section 3.3:

$$M = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \quad q = \begin{bmatrix} -2 \\ -1 \end{bmatrix}.$$

For initial starting pair (x^0, s^0) we take

$$x^0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad s^0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{6.1}$$

We can show that (x^0, s^0) is an infeasible starting pair by plugging them into $LCP(M, q)$ in (2.1) which is:

$$s = Mx + q$$

$$xs = 0, x \geq 0, s \geq 0$$

$$Mx^0 + q = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} -2 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \neq s^0$$

and also

$$(x^0)^T s^0 = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 2 \neq 0$$

Therefore, (x^0, s^0) is infeasible. Lemke's Method solves this question very efficiently. It is well known that Lemke's Method is very efficient for low dimensional problems. However, its efficiency reduces as dimension of the problem increases. For this example, we obtain the result $x = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ with 3 steps. On the other hand, we solve this example with improved full Newton step $IIPM$ with $\theta = \frac{1}{39+n}$, $\tau = \frac{1}{5}$, $\epsilon = 10^{-4}$ and the initial starting pair (x^0, s^0) as in (6.1). In theory, the expected number of iterations is 407 which is calculated by using Theorem 5.10. We have found the result $x = \begin{bmatrix} 1.9999 \\ 2.9999 \end{bmatrix}$ in 416 iterations. Obviously, this result converges to the exact solution $x = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$.

Example 3: In this example we apply full Newton step infeasible IPM to the following QP problem and compare our result with Lemke's Method.

$$\min \ \frac{1}{2}x_1^2 - x_1x_2 + \frac{1}{2}x_2^2 + 4x_1 - x_2$$
$$\textit{such that } \ x_1 + x_2 \geq 0, x_1, x_2 \geq 0.$$

Since the general form of the QP is

$$\min \ \frac{1}{2}x^T Qx + p^T x \ \textit{ subject to } \ Ax \geq b, x \geq 0$$

, then the problem above has the form

$$Q = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \ A = \begin{bmatrix} 1 & 1 \end{bmatrix}, \ p = \begin{bmatrix} 4 \\ -1 \end{bmatrix}, \ b = \begin{bmatrix} 2 \end{bmatrix}.$$

Then, by using (3.2), we define the following LCP by setting

$$M = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & 1 & 0 \end{bmatrix}, \quad q = \begin{bmatrix} 4 \\ -1 \\ -2 \end{bmatrix}.$$

Since Q is positive semi-definite matrix; hence, M is positive demi-definite and there exist a unique solution. Lemke's Method solves this problem in 5 steps. The global solution is $x_1 = 0$ and $x_2 = 2$. Our algorithm with $\theta = \frac{1}{40+n}, \tau = \frac{1}{4}$ and $\epsilon = 10^{-4}$ solves this question in 448 iterations. We obtain the same global minimizer $x^* = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$.

In the following tables, we present some results obtained from MATLAB experiments. We randomly generated matrix M, vectors x^0 and q with different sizes. We analyse these inputs in two different ways. The first way is that we started to solve these problems with a feasible (x^0, s^0) pair. The second way is that we generate both x^0 and s^0 randomly. This means that our initial starting pair may be (x^0, s^0) infeasible. Thus, for these 2 ways, expected iteration numbers and computed iteration numbers differ from each other. Moreover, since numbers of iterations are different according to feasibility of initial pair, elapsed CPU time also differs. Our observations are given in the Table 6.3 and 6.4.

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	340	336	1.0721×10^{-2}
3×3	350	345	1.4080×10^{-2}
4×4	436	431	1.8200×10^{-2}
5×5	489	483	2.0961×10^{-2}
10×10	636	630	3.3836×10^{-2}
20×20	922	914	7.6644×10^{-2}
50×50	1619	1609	40.3594×10^{-2}
100×100	2842	2830	1.993561
200×200	5315	5299	19.068259

Table 6.3: $\theta = \frac{1}{39+n}, \tau = \frac{1}{5}, \epsilon = 10^{-4}$ and (x^0, s^0) is feasible pair

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	354	358	1.1410×10^{-2}
3×3	391	409	1.8885×10^{-2}
4×4	417	477	1.9430×10^{-2}
5×5	429	481	2.0452×10^{-2}
10×10	489	628	3.4805×10^{-2}
20×20	657	856	9.3725×10^{-2}
50×50	1032	1496	34.9231×10^{-2}
100×100	1723	2582	1.928713
200×200	3156	4873	17.310426

Table 6.4: $\theta = \frac{1}{39+n}, \tau = \frac{1}{5}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair

It can be observed from Table 6.3 and Table 6.4 that if we start with an infeasi-

ble pair (x^0, s^0), expected number of iterations decreases, and performed number of iterations is greater than expected number of iterations. Although performed number of iterations is bigger than expected, full Newton step $IIPM$ with infeasible starting pair in most cases requires smaller number of iterations than full Newton step $IIPM$ with feasible initial pair for $\theta = \frac{1}{39+n}, \tau = \frac{1}{5}$ and $\epsilon = 10^{-4}$.

We performed the same experiment for $\{\theta = \frac{1}{40+n}, \tau = \frac{1}{4}\}$, $\{\theta = \frac{1}{53+n}, \tau = \frac{1}{3}\}$ and $\{\theta = \frac{1}{170+n}, \tau = \frac{1}{2}\}$ values with infeasible starting pair. The results are given in Tables 6.5, 6.6, and 6.7.

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	356	355	1.2205×10^{-2}
5×5	421	464	2.0286×10^{-2}
10×10	497	645	3.5290×10^{-2}
100×100	1743	2590	1.852735

Table 6.5: $\theta = \frac{1}{40+n}, \tau = \frac{1}{4}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	477	496	1.6991×10^{-2}
5×5	543	627	2.6566×10^{-2}
10×10	638	804	4.3438×10^{-2}
100×100	1893	2845	2.004844

Table 6.6: $\theta = \frac{1}{53+n}, \tau = \frac{1}{3}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	1149	1463	4.4998×10^{-2}
5×5	1632	1934	7.9164×10^{-2}
10×10	1863	2357	12.3812×10^{-2}
100×100	3352	5020	3.776065

Table 6.7: $\theta = \frac{1}{170+n}, \tau = \frac{1}{2}, \epsilon = 10^{-4}$ and (x^0, s^0) is infeasible pair

Although, in theory, the convergence is not guaranteed for bigger θ values that are not in Table 5.1, we performed a MATLAB experiment for $\theta = 0.2$. Results are shown in Table 6.8.

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	417	45	0.0035
5×5	488	54	0.0040
10×10	577	61	0.0052
100×100	1938	87	0.0667
1000×1000	16795	113	41.864310
Example 3	444	48	0.0035

Table 6.8: $\theta = 0.2, \epsilon = 10^{-4}$

Surprisingly, we have seen that for $\theta = 0.2$, improved algorithm is much more effective. For $\theta = \frac{1}{40+n}$, the code solves LCP with 1000×1000 matrix M in 3 hours and 8 minutes with around 20000 iterations. When $\theta = 0.2$, it solves the same LCP in approximately 42 seconds within 113 iterations. As it can be seen from Table 6.8, for this θ value, algorithm works effectively for almost every size of the matrix M

even for negative q. We also tried $\theta = 0.5$ and $\theta = 0.9$ values. Like $\theta = 0.2$, these θ values works very well except $\theta = 0.9$ for the size of M is 1000×1000. X denotes that the algorithm doesn't converge to a positive (x, s) pair. Results can be seen from the Tables 6.9 and 6.10.

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	417	15	0.00084
5×5	488	17	0.00279
10×10	577	20	0.00325
100×100	1938	28	0.03270
1000×1000	16795	37	13.56987
Example 3	444	16	0.003193

Table 6.9: $\theta = 0.5, \epsilon = 10^{-4}$

size	expected iteration number	computed iteration number	CPU time (sec)
2×2	417	5	0.00049
5×5	488	6	0.00067
10×10	577	7	0.00269
100×100	1938	9	0.01194
1000×1000	16795	X	X
Example 3	444	5	0.00212

Table 6.10: $\theta = 0.9, \epsilon = 10^{-4}$

So far, we solved examples that have a unique solution. More specifically, in our examples the matrix M is positive definite and constant vector q is positive in most cases. It is obvious that when $q > 0$, $x = \bar{0}$ vector is trivial solution. Although

the matrix M is positive definite, when constant vector q is completely negative or have negative components, then the algorithm may not converge to the result. In the following Section, improved version of the algorithm is compared with old version of the algorithm.

6.1 Comparison with old version

In this Section, we compare new and old version of full Newton step $IIPM$ algorithm. Our improvement is based on Lemma 5.4 proved in [1]. The most important difference between old and new version is that we eliminate the centering steps in new version thanks to Lemma 5.4. In the old version, we have to perform one feasibility step and a few centering steps, at each iteration because after one feasibility step new pair (x^f, s^f) may not close enough to central path. In other words, $\delta(x^f, s^f\mu^+) > \tau$.

The improved version of the algorithm guarantees that after one feasibility step new iterates x^f and s^f are close enough to central path i.e., $\delta(x^f, s^f\mu^+) \leq \tau$. Hence, we eliminate centering steps. Eliminating centering steps reduces the required number of iterations and CPU time that is required to find solution. Comparison between old version of the algorithm and improved version of the algorithm can be seen in the Table 6.11.

size	old version(iter. num., time)	improved version(iter. num., time)
3×3	$385 - 1.3239 \times 10^{-2}$	$439 - 1.9803 \times 10^{-2}$
5×5	$716 - 3.2589 \times 10^{-2}$	$506 - 2.5870 \times 10^{-2}$
10×10	$1626 - 7.7880 \times 10^{-2}$	$676 - 3.7156 \times 10^{-2}$
100×100	$23917 - 16.955375$	$2697 - 1.894107$

Table 6.11: $\theta_{new} = \frac{1}{40+n}$, $\theta_{old} = \frac{1}{12n}$, $\tau = \frac{1}{4}, \epsilon = 10^{-4}$

The values for old version in the Table 6.11 were generated by the MATLAB code given in [33]. It can be seen from this table that for large dimensional $LCPs$, improved version of full Newton step $IIPM$ is much more efficient than the old version. It requires smaller number of iteration and consumes less CPU time; thus reduces the cost. This difference comes from the θ values and eliminating centering steps at each iteration. In this thesis we use $\theta = \frac{1}{40+n}$ for $\tau = \frac{1}{4}$. On the other hand, for old version of the algorithm $\theta = \frac{1}{12n}$ for the same τ. Hence, when $n \geq 4$, improved algorithm is much more efficient than the old version.

Note that for both algorithms, we have performed our computational experiments with the computer that has Intel(R) core(TM) processor and 4 Gb of RAM running Windows 7 operating system. MATLAB codes for new version of the algorithm are given in Appendix A.

CHAPTER 7

CONCLUSION

In this thesis, we consider to solve monotone Linear Complementarity Problems ($LCPs$) (2.1) with positive semi-definite matrix M. LCP is not an optimization problem. However, KKT conditions of several important optimization problems, such as Linear and Quadratic Programming problems (LP and QP), can be formulated as LCP. In addition, many problems from engineering, transportation, finance etc. can be directly or indirectly written as LCP. Hence, it is of significant interest to consider $LCPs$ and find efficient methods to solve it.

$LCPs$ can be solved by using some non-polynomial pivot based algorithms such as Lemke's Method. These algorithms are very successful in practice. Especially, for low dimensional problems, these algorithms have very good performance. However, when dimension of the problem increases, then the complexity of the algorithm and number of iterations increase as well. Furthermore, if the worst case complexity theory is considered, these algorithms are not polynomial. They may require exponential number of iterations. Thus, it is important to consider other methods. The class of methods that have shown to be very efficient and with polynomial worst case complexity has been developed in past three decades and are called Interior-Point Methods (IPM). In this thesis, we consider full Newton step Infeasible IPM.

The most important property of Infeasible IPM algorithm is that it is a polynomial Newton based algorithm. Secondly, we don't calculate step size because we use full Newton step i.e., $\alpha = 1$. Infeasible IPM may find solutions for both feasible and infeasible initial starting pair (x^0, s^0). Since it is not easy to find a feasible starting pair all the time, solving problems with infeasible starting pair is essential. This situation increase importance of Infeasible IPM. The Newton Method (NM) is essential part of the $IPMs$; it is used at each iteration of IPM in essential way. In our case

$IIPM$ uses NM to solve system (4.1). However, the complementarity condition in the system (4.1) can be a problem for NM. Hence, to prevent this problem we consider perturbed complementary condition which then leads to the concept of central path. The main idea is to follow central path approximately (in a certain neighbourhood of the central path) until reaching ϵ-approximate solution.

We improved the old version of the algorithm given in [33]. Our improvement is based on the lemma proved in [1]. In the old version of the algorithm, we perform one feasibility step and a few centering steps at each iteration. After one feasibility step, the new iterates are strictly feasible but not close to central path. On the other hand, in the improved version, we only perform feasibility step. After one feasibility step, new iterates are both strictly feasible and close enough to central path. This is achieved for certain values of θ and τ. These values are given in Table 5.1. One main iteration of this algorithm can be seen in Figure 5.1. Also, pseudocode of the algorithim is given in Section 4.2.1.

The pseudocode in Section 4.2.1 is implemented via MATLAB. We performed several experiments for different size of randomly generated matrix M, randomly generated initial starting pairs (x^0, s^0), constant vector q and for different values of θ, τ and ϵ. We observed that, almost for all test problems, this algorithm converges to the solution. Although for low dimensional problems, the old version is better, for large dimensional problems, improved version of the algorithm is much more efficient. Although theoretically the algorithm doesn't guarantee the convergence for $\theta = O(1)$, we have shown that the algorithm works very efficiently and solves most of the problems in much fever number of iteration then in the version of the algorithm that guarantees convergence. We provided results for $\theta = 0.2,\ 0.5,\ 0.9$ in Tables 6.8, 6.9 and 6.10. Results of numerical tests and comments are given in Chapter 6 (Numerical Results).

The results in this thesis show that the improved version of the full Newton step $IIPM$ is both efficient in theory and in practice and performs better when compared to the old version.

REFERENCES

[1] Roos C., *An improved and simplified full-Newton step $O(n)$ infeasible interior point method for linear optimization*, SIAM J. Optim., 25(2015).

[2] Ferris M. C., Mangasarian O. L. and Wright S. J., *Linear programming with MATLAB*, SIAM, Series on Optimization, (2007).

[3] Dantzig G. B., *Linear programming and extensions*, Princeton NJ: Princeton University Press, (1963).

[4] Kumar D. N., Optimization Methods lecture notes, Indian Institute of Science, Bangalore

[5] Lesaja G., *Introducing interior point methods for introductory operations research courses and/or linear programming courses*, The Operational Research Journal, 3: pp. 1-12, (2009).

[6] Klee V. and Minty G. J., *How good is the Simplex algorithm?*, In: Shisha O, Ed. Inequalities, New York: Academic Press, pp. 159-175, (1972).

[7] Murty K. G., *Computational complexity of complementary pivot methods*, Math Program Study, 7: pp. 61-73, (1978).

[8] Khachiyan L. G., *A polynomial algorithm in linear programming*, Sov Math Dokl, 20: pp. 191-194, (1979).

[9] Shor N. Z., *Cut-off method with space extension in convex programming*, Kibernetica, 6: pp. 94-95, (1977).

[10] Arora S., Advanced Algorithm Design lecture notes, Princeton University, (Fall 2005).

[11] Karmarkar N., *A polynomial-time algorithm for linear programming*, Combinatorica, 4: pp. 373-395, (1984).

[12] Kranich E., *Inerior-point methods for mathematical programming: a bibliography*, Discussionbeitrag 171, FerUniversitat Hagen, Hagen, Germany, (1991).

[13] Gill P. E., Murray W., Saunders M. A., Tomlin J. A. and Wright M. H., *On the projected Newton barrier methods for linear programming and an equivalence to Karmarkar's projective method*, Math Program, 36: pp. 183-209, (1986).

[14] Renegar J., *A polynomial-time algorithm, based on Newton's method, for linear programming*, Math Program, 40: pp. 59-73, (1988).

[15] Ye Y., *An $O(n^3L)$ potential reduction algorithm for linear programming*, Math Program, 50: pp. 239-258, (1991).

[16] Kojima M., Mizuno S. and Yoshise A., *A primal-dual interior-point algorithm for linear programming*, In: N. Meggido, Ed.,Progress in mathematical programming: Interior-point algorithms and related methods, Springer-Verlag, Berlin, pp.29-47, (1989).

[17] Meggido N., *Pathways to the optimal set in linear programming*, In: N. Meggido, Ed.,Progress in mathematical programming: Interior-point algorithms and related methods, Springer-Verlag, Berlin, pp.131-158, (1989).

[18] Cottle R. W., Pang J. S. and Stone R. E., *The linear complementarity problem*, Academic Press, Boston, (1992).

[19] Lemke C. E., *Bimatrix equilibrium points and mathematical programming*, Management Science II, pp. 681-689, (1965).

[20] Roos C., Terlaky T. and Vial J-PH., *Interior-point approach to linear optimization: Theory and algorithms 2^{nd} ed.*, Springer-Verlag, Berlin, (2005).

[21] Lesaja G., *Interior-point methods for P*-complementarity problems*, Ph.D. thesis, University of Iowa, Iowa City, (1996).

[22] Andersen E. D. and Ye Y., *On a homogeneous algorithm for the monotone complementarity problem*, Math Program, 84(2): pp.375-399, (1999).

[23] Fachinei F. and Pang J. S., *Variational inequalities and complementarity problems*, Springer-Verlag, vol I and II, Berlin, (2003).

[24] Lesaja G. and Roos C., *Unified analysis of kernel-based interior-point methods for $P_*(\kappa)-$linear complementarity problems*, SIAM J. Optim., 20(6): pp. 3014-3039, (2010).

[25] Fiedler M. and Ptak V., *Some generalizations of positive definiteness and monotonicity*, Numerische Mathematik, 9: pp.163-172, (1966).

[26] Fiedler M. and Ptak V., *On matrices with non-positive off-diagonal elements and positive principle minors*, Czechoslovak Mathematical Journal, 12: pp.382-400, (1962).

[27] Cottle R. W. and Guu S-M., *Two characterizations of sufficient matrices*, Systems Optimization Laboratory, Stanford University, California, (1990).

[28] Cottle R. W., Pang J.-S. and Venkateswaran V., *Sufficient matrices and the linear complementarity problem*, Linear Algebra and Its Applications, 114/115: pp.231-249, (1989).

[29] Samelson H., Thrall R. and Wesler O., *A partition theorem for Euclidean n-space*, In Proceedings of American Mathematical Society, 9: pp.805-807, (1958).

[30] Kojima M., Megiddo N, Noma T. and Yoshise A., *A unified approach to interior point algorithms for linear complementarity problems*, Springer-Verlag, Berlin, (1991).

[31] Frank M. and Wolfe P., *An algorithm for quadratic programming*, Naval Research Logistics Quarterly, 3: pp.95-110, (1956).

[32] Cottle R. W. and Dantzig G. B., *Complementary pivot theory of mathematical programming*, Linear Algebra and Its Applications, 1: pp. 103-125, (1968).

[33] Drummer A. M., *Infeasible full Newton-step interior point method for the linear complementarity problems*, Master thesis, Georgia Southern University, Statesboro, (2012).

Appendix A

MATLAB CODES

In this part, we present MATLAB codes for the algorithm given in Section 4.2.1. The first program is $LCP_main.m$. In this program, we generate random LCP with positive definite matrix M and initial starting pair (x^0, s^0). Then, we send these inputs to $FNSIIPM.m$. $FNSIIPM$ is abbreviation for full Newton step infeasible interior point method.

In $FNSIIPM.m$, we implemented the algorithm in Section 4.2.1. This subroutine takes M, $x^0, s^0, q, \theta, \tau$ as inputs and generates solution x with a table showing each iteration in detail. As it mentioned before, (x^0, s^0) can be both feasible and infeasible.

The last MATLAB subroutine is $Proximity.m$. We implemented this code to determine θ and tau values which guarantee that $\delta(x^f, s^f; \mu^+) \leq \tau$.

A.1 LCPmain.m

```
1  % Main Program
2  % Mustafa Ozen
3  % Under direction of Prof. Dr. Goran Lesaja.
4  % Georgia Southern University
5  % Applied Mathematics Master Degree Thesis
6  % 2015-2016 Summer Semester
7  % Full Newton Step Infeasible Interior Point Method for Linear
8  % Complementarity Problems (LCP).
9
10 % This code solves random generated LCP for positive definite matrix M
       and
11 % vector q by using Full-Newton step Infeasible Interior Point Method.
12
```

```
% Variable List:
    % n: dimension of the problem.
    % A: random n x n matrix.
    % M: random generated positive semidefinite matrix
    % q: random generated n x 1 constant vector.
    % init_x: initial starting vector x.
    % init_s: initial starting vector s.
    % theta: barrier parameter.
    % epsilon: accuracy rate.
    % x: output vector of IIPM algorithm.
    % s: output cevtor of IIPM algorithm.
    % iteration: number of iteration.
    % expected_iter_num: theoriticaly expected number of iteration.

clc
clear
tic

% Generating matrix M, constant vector q and initial pair (x0,s0)
% Random generated inputs
n = 3;
A = rand(n,n);
M = A'*A;
q = -1*rand(n,1);
init_x = rand(n,1);
init_s = rand(n,1);

% Example 1:
%M = [0.4512 0.6328;0.6328 0.9995]
%q = [0.5441;0.6990]
%init_x = [0.0791;0.5094]
```

```
%init_s = M*init_x + q;

% Example 2:
%M = [1  0;-1  1];
%q = [-2;  -1];

% Example 3:
%M = [1 -1 -1; -1 1 -1; 1 1 0];
%q = [4; -1; -2];

%init_x = ones(n,1);
%init_s = ones(n,1);

% Setting theta, tau and epsilon values
% For different theta and epsilon values, expected number of iterations
% and formulations change. For possible theta, tau values and number of
% iterations see Table 5.1 in Chapter 5.

theta = 1/(40 + n);
%theta = 0.2;
tau = 1/4;
epsilon = 1e-4;
expected_iter_num = (40 + n)*log((33*init_x'*init_s)/(32*epsilon));
% Outputs:
[x s iteration table] = FNSIIPM(M,init_x,init_s,q,theta,epsilon);
x
s
iteration
expected_iter_num
table
toc
```

A.2 FNSIIPM.m

```
function [x s num_of_iter table] = FNSIIPM(M, init_x , init_s ,q,theta ,
    epsilon)

% function [x s num_of_iter number] = FNSIIPM(M,x,q,theta ,epsilon)
% This subroutine performs Full Newton Step Infeasible IPM Algorithm.
% It solves LCP(M,q): s = Mx + q suct that xs = 0.
% This upgraded algorithm requires only feasibility steps.

% Inputs:
    % M: n x n positive semidefinite matrix.
    % init_x: n x 1 initial starting vector x.
    % init_s: n x 1 initial starting vector s.
    % q: n x 1 constant vector.
    % theta: barrier parameter.
    % epsilon: accuracy rate.
% Outputs:
    % x: n x 1 result vector.
    % s: n x 1 result vector.
    % num_of_iter: performed number of iteration.
    % table: result table.

% Setting inputs
n = size(M,1);
x = init_x;
s = init_s;

mu = x'*s/n;
nu = 1;
r0 = s - M*x - q;
r = nu*r0;
```

```
30 e = ones(n,1);
31 comp_cond = x'*s;
32 num_of_iter = 0;
33 flag = 0;
34
35 % Algorithm, for Pseudocode see Chapter 4 - Section 4.2.1.
36 while max(comp_cond, norm(r)) >= epsilon
37     flag = 1;
38     num_of_iter = num_of_iter + 1;
39
40     X = diag(x);
41     S = diag(s);
42     dx = (S + X*M)\((1-theta)*mu*e - X*S*e + X*theta*nu*r0);
43     ds = M*dx - theta*nu*r0;
44
45     x = x + dx;
46     s = s + ds;
47     mu = (1 - theta)*mu;
48     nu = (1 - theta)*nu;
49     r = nu*r0;
50     comp_cond = x'*s;
51     v = sqrt(x.*s./mu);
52     delta_f = norm(v.^(-1) - v)/2;
53
54     if num_of_iter == 1
55         table{1} = '------------------IIPM Iteration Table
               ------------------------------';
56         table{2} = ' iteration        xs             mu             nu
                    delta_f';
57         table{3} = '
              ------------------------------------------------------------
```

```
            ';
58      end
59      table{3+num_of_iter} = ...
60              sprintf('   %3u:      % 5.8f      % 5.8f      % 5.8f      %
                    5.8f',...
61              num_of_iter,comp_cond,mu,nu,delta_f);
62  end
63
64  if flag == 0
65      disp('no result!')
66  else
67  table = char(table);
68  end
69  end
```

A.3 Proximity.m

```
% Measuring Proximity for different theta and tau values

% This subroutine measures proximity delta of new iterates after one
% feasibility step for possible theta, tau and problem dimension n.
% We want that after a feasibility step, new iterates are close enough
    to
% central path, i.e., delta_f <= tau.

% Variable List:
    % n: dimension of the problem.
    % theta: barrier parameter.
    % tau: central path neighbourhood radius.
    % delta: proximity of new iterates to the central path.
    % q, w, l and xi function definitions are given in Chapter 5.

clc
clear
n = 2;
theta = 1/(170+n);
tau = 1/2;
delta = tau;
q = delta + sqrt((delta^2) + 1);
w = (2*delta^2 + theta) + 9*(theta^2)*((2 + q)^2)*q^2 + 3*theta*...
    sqrt(4*(delta^2)+2*theta)*(2 + q)*q;
%n = 1:100;
xi = @(t) (1+t)/(1-theta) + (1-theta)/(1+t) - 2;
l = max(xi(w),xi(-w));
delta_f = (1/2)*sqrt((n-1)*xi(0)+l)

%
```

```
30 % Graphical representation of new delta after a feasibility step:
31 % n = 2:10;
32 % delta = 1/5;
33 % q = delta + sqrt((delta^2) + 1);
34 % for theta = 0:0.001:0.99
35 % w = (2*delta^2 + theta) + 9*(theta^2)*((2 + q)^2)*q^2 + 3*theta*...
36 %     sqrt(4*(delta^2)+2*theta)*(2 + q)*q;
37 % xi = @(t) [(1+t)/(1-theta)] + [(1-theta)/(1+t)] - 2;
38 % l = max(xi(w),xi(-w));
39 % delta_f = (1/2)*sqrt((n-1)*xi(0)+l);
40 % plot(theta,delta_f)
41 % hold on
42 % end
43 % axis(([0 1 0 1]))
```