



## APPROVAL SHEET

Title of Dissertation: Homogenization of Mixed Horizontal LCPs and Applications

Name of Candidate: Cosmin G. Petra,  
Doctor of Philosophy, 2009

Dissertation and Abstract Approved: \_\_\_\_\_

Florian A. Potra

Professor

Department of Mathematics and Statistics

University of Maryland, Baltimore County

Date Approved: \_\_\_\_\_

## **ABSTRACT**

**Title of Dissertation:** HOMOGENIZATION OF MIXED HORIZONTAL  
LCPs AND APPLICATIONS

Cosmin G. Petra, Doctor of Philosophy, 2009

**Dissertation directed by:** Florian A. Potra  
Professor  
Department of Mathematics and Statistics  
University of Maryland, Baltimore County

In optimization, a homogeneous model is an artificial transformation of a given problem. The transformation is done such that the homogeneous problem has always a solution even if the original problem does not. Moreover, without having any assumption on the feasibility of the original problem, the homogeneous model is able to provide the solution if it exists, or a certificate of infeasibility otherwise. The first part of the thesis will introduce a homogeneous model for mixed linear complementarity problems which represent a generalization of the standard linear complementarity problems. We also study the properties of the model and show that the interior-point methods can be efficiently used for the numerical solutions of the homogenous problem.

The second part of the thesis is concerned with a computational study on the use of an optimization-based method in the simulation of fuel motion in a pebble bed reactor. The performance of several optimization packages (BLMVM, TRON, OOQP,

and Mosek) for quadratic problems needed to simulate the system is investigated and reported. OOQP will be presented with both the default solver MA27 and our implementation based on CHOLMOD. CHOLMOD-based OOQP version is the fastest of all the packages tested. It consistently uses only about three times more memory than BLMVM, while achieving far higher precision levels. Both solvers behave predictably with the number of pebbles and can be used as robust software solutions in the simulation of the pebble bed reactor.

**HOMOGENIZATION OF MIXED HORIZONTAL LCPS AND  
APPLICATIONS**

by  
**Cosmin G. Petra**

Dissertation submitted to the Faculty of the Graduate School  
of the University of Maryland in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy in Applied Mathematics

2009

© Copyright by  
Cosmin Petra  
2009



## **DEDICATION**

*To my parents*

# Contents

<b>Dedication</b>	<b>ii</b>
<b>Contents</b>	<b>iii</b>
<b>List of Tables</b>	<b>vi</b>
<b>List of Figures</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Complementarity problems . . . . .	2
1.1.1 Linear complementarity problems . . . . .	4
1.2 Thesis outline . . . . .	6
<b>2 A Homogeneous Model for Mixed Horizontal Linear Complementarity Problems</b>	<b>9</b>
2.1 Introduction . . . . .	9
2.2 More on the monotone mixed horizontal linear complementarity problem	14
2.3 An augmented homogeneous model for MLCP . . . . .	18
2.3.1 Notations and terminology . . . . .	19
2.4 Properties of the augmented HMCP . . . . .	22
2.5 A general theory on the existence of central paths for nonlinear complementarity problems . . . . .	32
2.6 Existence and properties of the central path for HMCP . . . . .	34

2.7	Adaptation of Mehrotra’s interior-point algorithm for HMCP . . . . .	40
2.7.1	The procedure for computing the steplength . . . . .	48
2.7.2	Linear algebra . . . . .	50
2.8	Generating random monotone linear complementarity problems . . . . .	53
2.8.1	Generating monotone SLCPs . . . . .	54
2.8.2	Generating monotone HLCPs . . . . .	58
2.8.3	Generating monotone MLCPs . . . . .	61
2.9	Numerical simulations . . . . .	63
2.10	Conclusions . . . . .	68
<b>3</b>	<b>Simulating rigid body systems with contact and friction</b>	<b>70</b>
3.1	Introduction . . . . .	70
3.2	Quadratic Programming Subproblems of Time-Stepping Methods . . . . .	73
3.2.1	The Model . . . . .	74
3.2.2	The Integration Step . . . . .	80
3.2.3	Pointed friction cone and duality . . . . .	82
3.3	Algorithms and Software Packages Used . . . . .	83
3.3.1	OOQP . . . . .	83
3.3.2	TRON . . . . .	93
3.3.3	BLMVM . . . . .	94
3.3.4	MOSEK . . . . .	95
3.4	Numerical Results . . . . .	95
3.4.1	Environment and Solver Configuration . . . . .	96
3.4.2	Examples Generation . . . . .	97
3.4.3	Total Kinetic Energy Results . . . . .	97
3.4.4	Performance Results . . . . .	99

3.4.5 Discussion of the Results . . . . .	105
3.5 Conclusions and Future Work . . . . .	106
<b>Bibliography</b>	<b>108</b>

# List of Tables

2.1	Possible combination of optimal $\tau^*$ and $\kappa^*$ . . . . .	32
2.2	Certificates for a feasible monotone MLCP . . . . .	64
2.3	Certificates for an infeasible monotone MLCP . . . . .	65
2.4	Solving feasible MLCPs and corresponding reduced SLCPs via HMCP and Andersen and Ye's homogeneous models, respectively. . . . .	67
2.5	Solving infeasible MLCPs and corresponding reduced SLCPs via HMCP and Andersen and Ye's homogeneous models, respectively. . . . .	68
3.1	Performance of QP solvers for 800 pebbles and $h = 0.05$ . . . . .	100
3.2	Performance of QP solvers for 1600 pebbles and $h = 0.05$ . . . . .	101
3.3	Performance of QP solvers for 3200 pebbles and $h = 0.05$ . . . . .	101
3.4	The performance of QP solvers in solving an optimization problem from the simulation of 1000 pebbles with $h = 0.01$ . . . . .	104
3.5	The performance of QP solvers in solving an optimization problem from the simulation of 1000 pebbles with $h = 0.05$ . . . . .	104

# List of Figures

3.1	Cross-section of the reactor vessel with 3200 pebbles at the end of the simulation. The pebbles are colored based on the originating position.	73
3.2	(a) Generic contact between two bodies, (b) Tangent space at contact.	76
3.3	Energy dependence on time. Even though the system is chaotic, simulations report approximately the same total kinetic energy (top figure). Larger differences (smaller than 1.5%) that occur around the equilibrium have a numerical explanation (small denominator) rather than a physical meaning (bottom figure).	98
3.4	Memory usage dependence on the number of constraints. Shown are simulations of 600, 800, 1000, and 1200 pebbles.	102
3.5	Execution time dependence on the number of constraints. Shown are simulations of 600, 800, and 1000 pebbles.	103
3.6	Execution time dependence on the number of constraints. Shown are simulations of 1200, 1400 and 1600 pebbles.	103

# Chapter 1

## Introduction

In mathematical optimization, the complementarity condition or simply the complementarity requires that the product of two or more nonnegative quantities be zero.

The simplest example of complementarity can be seen in contact physics. When two bodies come or are about to come in contact, complementarity arises between the contact force and the distance between the bodies. The force is positive only if the bodies are in contact, *i.e.*, the distance is zero; also the distance is positive when the force is zero, that is, there is no contact.

In economics, for example, the classical Walras' law describes an equilibrium of a commodities exchange market by enforcing a complementarity condition between the price and excess demand, that is, excess demand must be zero when the price is positive and the price must be zero if there is excess demand.

In fact, a wide range of practical problems in the economics, engineering, and natural sciences are modeled as complementarity problems: contact mechanics, structural mechanics, obstacle problems, traffic equilibrium problem, structural design, elasto-hydrodynamic lubrication problems, network design, optimal control, invariant capital stock, game theory, etc, see for example [19, 34, 37] for a list of applications

of complementarity problems. Moreover, optimization problems involving inequality constraints can be reduced to and often are solved as complementarity problems. The complementarity conditions are given by the Karush-Kuhn-Tucker conditions which, in fact, state that the optimal solution to the optimization problem is the solution of a complementarity problem, provided some regularity conditions are satisfied.

Complementarity problems can be numerically solved using nonsmooth Newton methods, Lemke-type algorithms, or interior-point methods. In this thesis we will use interior-point methods for the numerical solution of complementarity problems. Interior point methods, initially developed for linear programming, have been successfully generalized for the solution of several classes of complementarity problems. The study of interior point methods has been a very active area of research in optimization over the past two decades and has contributed to great advances in mathematical optimization. On the theoretical side, interior point methods have been used to prove polynomial complexity of several classes of optimization problems, such as linear programming, quadratic programming, semidefinite programming, second-order cone programming, etc. On the practical side, interior point methods have been implemented in highly efficient software packages that are now routinely used by practitioners working in different fields of science and technology.

## 1.1 Complementarity problems

In this section we introduce the basic terminology and concepts related to the complementarity problems studied in this thesis.

The explicit complementarity problem over the non-negative orthant consists of

finding vectors  $x \in \mathbb{R}^n$  and  $s \in \mathbb{R}^n$  satisfying

$$xs = 0, \quad s = f(x), \quad x, s \geq 0, \quad (1.1)$$

where  $f(x)$  is assumed to be a continuous mapping from  $\mathbb{R}^n$  to  $\mathbb{R}^n$ . The complementarity problem (1.1) is called *monotone* if the function  $f(x)$  is monotone, *i.e.*, for every  $x_1, x_2 \in \mathbb{R}_{++}^n$  we have:

$$(x_1 - x_2)^T (f(x_1) - f(x_2)) \geq 0. \quad (1.2)$$

In 1.2 the multiplication  $uv$  of two vectors  $u, v \in \mathbb{R}^n$  refers to the componentwise product  $[u_1v_1, u_2v_2, \dots, u_nv_n]$ , also known as Hadamard product. Often the complementarity conditions are described using the regular inner product  $\langle \cdot, \cdot \rangle$  in  $\mathbb{R}^n$ , but it can be easily observed that under the nonnegativeness condition  $x, s \geq 0$ , we have  $xs = 0$  if and only if  $\langle x, s \rangle = 0$ .

In this thesis, we employ a slightly more general type of complementarity. We also allow unknowns that satisfy no complementarity condition and have no sign constraints as part of the problem. These unknowns are called *free variables*. We will use the term *mixed* whenever the problem contains both complementarity and free variables. Also, the complementarity variables  $s$  will not be defined as a function of the other group of complementarity variables  $x$  and free variables  $y$ , but they are altogether described implicitly by a continuous, possibly nonlinear, mapping  $F(x, s, y)$  from  $\mathbb{R}^{2n+m}$  to  $\mathbb{R}^{n+m}$ .

The problem corresponding to this type of complementarity will be called an *implicitly defined (mixed) complementarity problem*, formally described by:

$$xs = 0, \quad F(x, s, y) = 0, \quad x, s \geq 0. \quad (1.3)$$

The above complementarity problem is called *monotone* (or  $(x, s)$ -equilevel-monotone as in [84, 61]) if for any  $(x_1, s_1, y_1), (x_2, s_2, y_2) \in \mathbb{R}^{2n+m}$  satisfying  $F(x_1, s_1, y_1) =$

$F(x_2, s_2, y_2)$ , the following inequality holds:

$$(x_1 - x_2)^T (s_1 - s_2) \geq 0. \quad (1.4)$$

The complementarity problem (1.3) is called *feasible* if there exists a point  $(x, s, y) \in \mathbb{R}_+^n \times \mathbb{R}_+^n \times \mathbb{R}^m$  such that  $F(x, s, y) = 0$ . Any such point is called a feasible point. The problem is said to be *infeasible* if it has no feasible point. The "feasibility" or "feasibility equations" will refer to the system of equations  $F(x, s, y) = 0$ .

A point  $(x^*, s^*, y^*) \in \mathbb{R}_+^n \times \mathbb{R}_+^n \times \mathbb{R}^m$  is a *solution* to the complementarity problem if it is feasible and satisfies the complementarity condition, that is  $xs = 0$ . In this case the complementarity problem (1.3) is called *solvable*.

### 1.1.1 Linear complementarity problems

The linear complementarity problems are the complementarity problems whose feasibility equations are described by an affine mapping, or, in other words, by a linear system of equations. However, if the problem is seen as finding the nonnegative roots of a system of equations, then the term "linear" may be found misrepresentative since the system is actually quadratic, hence nonlinear, in complementarity variables.

Given the  $n \times n$  matrix  $M$  and the vector  $b$  from  $\mathbb{R}^n$ , the linear complementarity problem in standard form (SLCP) consists of finding vectors  $x, s \in \mathbb{R}^n$  satisfying

$$\begin{aligned} xs &= 0, \\ s &= Mx + b, \\ x, s &\geq 0. \end{aligned} \quad (1.5)$$

The above problem corresponds to the complementarity problem (1.1) with  $f(x) = Mx + b$ .

The horizontal linear complementarity problem (HLCP) is a generalization of the standard linear complementarity problem. While for the SLCP case the complementarity variables  $s$  are explicitly defined by an affine function of complementarity variables  $x$ , for the horizontal form  $x$  and  $s$  are implicitly defined by an affine mapping of the form  $Qx + Rs = b$ . Thus, given  $Q \in \mathbb{R}^{n \times n}$ ,  $R \in \mathbb{R}^{n \times n}$  and  $b \in \mathbb{R}^n$ , the horizontal linear complementarity problem  $\text{HLCP}(Q, R, b)$  consists of finding vectors  $x \in \mathbb{R}^n$  and  $s \in \mathbb{R}^n$  satisfying

$$\begin{aligned} xs &= 0 \\ Qx + Rs &= b \\ x, s &\geq 0. \end{aligned} \tag{1.6}$$

It can be seen that the standard form (1.5) is a particular case of the horizontal form by taking  $R = -I$  and  $Q = M$ .

We say that the HLCP (1.6) is monotone, or equivalently  $(Q, R)$  is a monotone pair, if

$$Qu + Rv = 0 \text{ implies } u^T v \geq 0, \text{ for any } u, v \in \mathbb{R}^n. \tag{1.7}$$

We can use either the above definition with  $Q = M$  and  $R = -I$ , or definition (1.2) to observe that the monotonicity of the SLCP (1.5) reduces to the positive semidefiniteness of the matrix  $M$ .

The mixed (horizontal) linear complementarity problem (MLCP) generalizes the horizontal linear complementarity problem by allowing  $m$  free variables and corresponds to the general form of complementarity given by (1.3) with  $F$  affine. Given  $A \in \mathbb{R}^{(m+n) \times n}$ ,  $B \in \mathbb{R}^{(m+n) \times n}$ ,  $C \in \mathbb{R}^{(m+n) \times m}$ , and  $b \in \mathbb{R}^{n+m}$ , the mixed horizontal linear complementarity problem consists of finding the vectors  $x \in \mathbb{R}^n$ ,  $s \in \mathbb{R}^n$  and

$y \in \mathbb{R}^m$  satisfying

$$\begin{aligned} xs &= 0 \\ Ax + Bs + Cy &= b \\ x, s &\geq 0. \end{aligned} \tag{1.8}$$

It can be seen that the horizontal form is a particular case of the mixed horizontal form by taking  $m = 0$ , *i.e.*, no free variables.

By taking  $F(x, s, y) = Ax + Bs + Cy$  in (1.4), the monotonicity of the MLCP (1.8) is equivalent to the following implication:

$$Au + Bv + Cw = 0 \text{ implies } u^T v \geq 0, \text{ for any } u, v \in \mathbb{R}^n \text{ and } w \in \mathbb{R}^m. \tag{1.9}$$

In this thesis we may also say that  $(A, B, C)$  is a monotone triplet to indicate that MLCP (1.8) is monotone.

The abbreviations SLCP, HLCP and MLCP will usually denote the class of problems: standard complementarity problems, horizontal complementarity problems, and mixed horizontal complementarity problems, respectively. To differentiate between two instances within the same class, we will use the acronym together with the instance's data, *e.g.*, SLCP( $M, b$ ) for the SLCP (1.5), HLCP( $Q, R, b$ ) for the HLCP (1.6), and MLCP( $A, B, C, b$ ) for the MLCP (1.8).

## 1.2 Thesis outline

Chapter 2 introduces a new homogenization model for monotone mixed linear complementarity problems. To the best of our knowledge, the model is the only homogenization technique that works directly with implicitly defined complementarity. Given a monotone MLCP, we artificially transform the problem into a homogeneous problem. The transformation is done such that the homogeneous problem has always a solution

even if the original problem does not. Moreover, without having any assumption on the feasibility of the original problem, the homogeneous model is able to provide the solution if it exists, or a certificate of infeasibility otherwise.

Furthermore, a method for obtaining the numerical solution to the homogeneous problem will be presented in the same chapter. Similar to Mehrotra’s predictor-corrector algorithm for linear and convex quadratic programming, the algorithm we propose tries to follow a continuation curve that starts at any strictly positive point and ends at a solution possessing the maximal complementarity property. Numerical experiments show that homogenizing the mixed linear complementarity problems is faster and exhibits smaller memory requirements than converting them to standard linear complementarity problems and homogenizing by using existing models.

Chapter 3 presents a computational study on the use of an optimization-based method for the simulation of fuel motion in a pebble bed reactor. The ”fuel” consists of several hundreds of thousand of uranium pebbles about the size of a tennis ball that move in the gravitational field and are subject to frictional contacts coming from both pebble-pebble and pebble-wall interactions.

The dynamic rigid multi body contact problem is used to predict the motion of the pebbles. The approach used for the numerical approximation of rigid multi-body dynamics with contact and friction is a velocity-impulse LCP-based time-stepping method [75, 76, 8, 13, 68]. It requires the solution of a copositive LCP at each integration step. Such LCPs are generally solved by means of Lemke-type algorithms, and solvers such as the PATH solver [28, 38] have proved to be robust. For large systems (beyond a few thousand contacts), however, the PATH solver or any other pivotal algorithm becomes impractical from a computational point of view [12, 79].

The convex relaxation previously introduced in [9] requires at each integration

step the solution of a convex quadratic problem which can be solved in polynomial time and for which a wide variety of state-of-the-art optimization packages exist. We investigate and report on the performance of several solvers for quadratic problems that appear from the relaxed formulation of the pebble bed reactor. In particular we have used BLMVM [18], TRON [54], MOSEK [1] and OOQP [41]. OOQP will be presented with both the default symmetric indefinite solver MA27 [32] and our implementation based on CHOLMOD [26, 22, 25, 24].

In the case of the pebble bed reactor, it turns out that CHOLMOD-based OOQP version is the fastest of all the packages tested. It consistently uses only about three times more memory than BLMVM, while achieving far higher precision levels. The findings suggest that both solvers behave predictably with the number of pebbles and can be used as robust software solutions in the simulation of the pebble bed reactor.

# Chapter 2

## A Homogeneous Model for Mixed Horizontal Linear Complementarity Problems

### 2.1 Introduction

By a homogeneous complementarity problem, one should understand that the system of equations defining the feasibility is homogeneous, *i.e.*, the function describing the system is homogeneous. A function  $f$  is called homogeneous (of degree 1), if for any  $x$  in the domain of  $f$ , and any real number  $t$ ,  $f(tx) = tf(x)$  holds.

In the context of constrained optimization, homogeneity is a concept used for linear programming to indicate that all right-hand sides of the constraints are zero. For example, the linear problem

$$\begin{aligned} & \text{minimize} && c^T x \\ & \text{subject to:} && Ax = b, x \geq 0 \end{aligned} \tag{2.1}$$

is homogeneous when  $b = 0$ .

In the context of interior-point methods, homogenization generally refers to an artificial transformation of a given (complementarity or constrained optimization) problem to a homogeneous problem. The homogeneity causes the transformed problem to have nicer properties than the original problem, such as a trivial feasible starting point, solvability in any circumstances, etc. The solution sets of the two problems must also be related in the sense that once the homogeneous problem is solved, a solution to the original problem or a certificate that a solution does not exist, is readily available.

Certificates of infeasibility are produced by the simplex method by detecting the unboundness of either primal or dual problem. However, the simplex method can be applied only to linear programming and its extension for linear complementarity problems, *e.g.* Lemke's method, does not offer such certificates. Standard interior-point algorithms also do not offer bullet-proof evidence, *i.e.*, a certificate, for the problem to be infeasible. Moreover, there are numerical issues in the interior-point-based implementations when solving infeasible problems, since some of the problem's variables diverge to infinity. In the last fifteen years, interior-point methods that provide infeasibility certificates have been always used in conjunction with a homogenization mechanism.

Having a proof of infeasibility is important for a couple of reasons. First, the model that gives rise to the infeasible problems may be defective. For example, obtaining infeasible problems in modeling physical phenomena would indicate this situation. Second, the problem may be infeasible because of invalid data or human error in the input process.

The homogenization of a constrained optimization problem is a concept that has been used with interior-point methods since their appearance, not necessarily as a

technique that detects infeasibility. Karmakar's algorithm [53], generally considered to be the first interior-point algorithm, transforms the original linear problem to a homogeneous linear problem (called "canonical form") in order to obtain a feasible starting point. Anstreicher [15] used homogenization to devise a polynomial-time interior algorithm which solves a linear program with no assumptions of a non-empty interior of the primal and/or dual problem. However, because a Phase I-Phase II technique is employed and the solution of a linear system twice as large as in the case of other methods needs to be found at each iteration, Anstreicher's algorithm is expensive in practice.

The homogeneous interior-point algorithm for linear programming introduced by Ye, Todd and Mizuno [83] uses a *self-dual embedding* technique to incorporate the original linear problem together with its dual problem in a larger homogeneous linear problem that turns out to be self-dual. This algorithm was the first homogenization technique that is capable of providing certificates of infeasibility of the original problem and has become a standard for homogenous interior-point methods because of its properties:

- solves the problem without any regularity assumptions concerning the existence of optimal, feasible or interior feasible points;
- can start at any positive point, feasible or infeasible;
- each iteration requires the solution of a linear system whose dimension is almost the same as for standard (primal-dual) interior-point algorithms;
- if the problem has a solution, the algorithm is convergent; if the problem is infeasible or unbounded, then the algorithm will detect this situation by providing a "certificate" of infeasibility for at least one of the primal and dual problems;

- is a one-phase algorithm and has  $O(\sqrt{n}L)$ -iteration complexity.

In addition, the method improves the behavior and computational cost of Anstreicher's algorithm.

Ye [82] showed that the above technique is also suitable for monotone linear complementarity problems in standard form. The homogenization yields a self-dual homogeneous monotone linear complementarity problem that possesses the same properties as the model for linear programming. The homogeneous linear complementarity problem is self-dual in the sense that if the original linear complementarity problem arises from a linear program, then the homogeneous linear complementarity problem represents the self-dual embedding of the linear program.

Shortly after, a homogeneous model for the more general monotone nonlinear complementarity problems in explicit form was made available in [4]. The following homogeneous model related to a monotone complementarity problem in standard form (1.1) is suggested:

$$\begin{aligned}
 xs &= 0 \\
 \tau\kappa &= 0 \\
 \begin{bmatrix} s \\ \kappa \end{bmatrix} &= \begin{bmatrix} \tau f(x/\tau) \\ -x^T f(x/\tau) \end{bmatrix} \\
 x, s, \tau, \kappa &\geq 0.
 \end{aligned} \tag{2.2}$$

This model is also called *augmented* since it contains two additional one-dimensional complementarity variables. Here, the homogenization preserves the monotonicity of the problem but, unlike for linear cases, the model is given by a nonlinear system even if the original problems are linear. However, the model exhibits the characteristics of [83] (the polynomial time complexity is obtained under the assumption of Lipschitz continuity of  $f$ ). The homogeneous model (2.2) can also be applied to standard

monotone nonlinear complementarity problems over symmetric cones as Yoshise has recently shown in [84].

Several researchers have studied homogenization in the context of conic programming. Conic programming is an extension to linear programming and consists of minimizing a linear function over the intersection of an affine space with a closed convex cone. When the cone is polyhedral, the conic programming reduces to the linear programming. Another important example of conic programming is semidefinite programming in which the cone is the cone of symmetric positive semidefinite real matrices. Homogeneous self-dual models for semidefinite programming have been independently investigated by Potra and Sheng [67] and De Klerk, Roos and Terlaky [27]. Also Luo, Sturm and Zhang [56] and Nesterov, Todd and Ye [63] have studied homogenization techniques in the general case of conic programming.

Although considerable research has been devoted to homogenization of explicitly defined complementarity problems, rather less attention has been paid to homogenization of the more general case of implicitly defined complementarity. To the best of our knowledge, there is no homogenization technique that can be applied to such problems. It is worth mentioning that, in the case of an implicitly defined *linear* complementarity problem, *i.e.*, HLCP and MLCP, the existing homogenization models [82, 4] can be applied to an equivalent SLCP. However, since the transformation of a MLCP or HLCP to a SLCP requires the inversion of a matrix, the sparsity of the data is adversely affected and poor practical performance is obtained as we show in Section 2.9.

Our primary objective in this chapter is to provide a homogeneous augmented model for monotone mixed horizontal linear complementarity problems. In addition, we study the properties and the central path associated with the homogeneous model

and show that it has the desirable properties previously listed and can be solved by means of path-following interior-point methods.

This chapter emphasizes the theoretical properties of the proposed homogeneous model and is organized as follows. Section 2.2 presents in detail the properties of the mixed horizontal linear complementarity problem. Then in Section 2.3 we introduce the homogeneous model whose properties are studied in Section 2.4. Several previous results on the existence and properties of central path for nonlinear monotone complementarity are given in Section 2.5. In Section 2.6 we show that our homogeneous model possesses the same properties as the previously mentioned homogeneous models. Section 2.7 proposes a numerical method for the solution of the homogeneous problem and presents the associated computational cost. Random generated monotone MLCPs that are generated using a technique described in 2.8 are used for the numerical experiments presented in Section 2.9. Finally, Section 2.10 concludes on the theoretical and numerical behavior of our homogenization model.

## **2.2 More on the monotone mixed horizontal linear complementarity problem**

This section discusses the class of monotone mixed horizontal linear complementarity problems in implicit form (1.8) and shows that additional conditions have to be assumed on the problem's data in order to be solved by means of interior-point methods.

We start by pointing out that the MLCP is supposed to be pre-processed. Pre-processing, also known as pre-solving, of a problem is part of any implementation based on interior-point methods. It consists of several sweeps through problem's data

and aims the removal of empty (zero) rows and columns, the detection of infeasible variables, the elimination of the fixed variables, the removal and detection of redundant equations, etc, (see [2] for a comprehensive study of pre-processing). Here we suppose that the the linear system  $Ax + Bs + Cy = b$  defining the feasible set of the MLCP contains no redundant equation, hence we have:

**Assumption 2.1.** *The matrix  $\begin{bmatrix} A & B & C \end{bmatrix}$  has full row rank  $m + n$ .*

As we mentioned in Section 1.1.1, both the horizontal linear complementarity problems and the mixed linear complementarity problems in explicit form are particular cases of our MLCP form. The following lemma shows that the free variables can be removed from a MLCP and a horizontal form of linear complementarity is obtained, but the transformation results in a HLCP only if additional conditions on the MLCP (1.8) are assumed.

**LEMMA 2.1.** *Suppose that  $\dim \text{Ker}(C^T) = k$ . Let  $\{e_1, e_2, \dots, e_k\}$  be an orthonormal basis of  $\text{Ker}(C^T)$ , with  $e_i \in \mathbb{R}^{(n+m)}$ ,  $i = 1, \dots, k$ . Also consider the matrix  $E = [e_1 e_2 \dots e_k] \in \mathbb{R}^{(m+n) \times k}$ . Then the following affirmations hold:*

(i) *MLCP( $A, B, C, b$ ) is feasible (solvable) if and only if the following complementarity problem is feasible (solvable):*

$$\begin{aligned} xs &= 0 \\ E^T Ax + E^T Bs &= E^T b \\ x, s &\geq 0. \end{aligned} \tag{2.3}$$

(ii) *MLCP( $A, B, C, b$ ) is monotone if and only if the horizontal linear complementarity problem (2.3) is monotone.*

(iii) *The matrix  $\begin{bmatrix} E^T A & E^T B \end{bmatrix} \in \mathbb{R}^{k \times 2n}$  has full row rank  $k$ .*

*Proof.* (i.) Let us first observed that  $\{(x, s) : \exists y \in \mathbb{R}^m \text{ such that } Ax + Bs + Cy = b\} = \{(x, s) : Ax + Bs + Cy - b \in \text{Ran}C\} = \{(x, s) : E^T(Ax + Bs - b) = 0\}$ , proving that  $(x^*, s^*, y^*)$  is a feasible point of  $\text{MLCP}(A, B, C, b)$  if and only if  $(x^*, s^*)$  is a feasible point of (2.3).

Furthermore,  $(x^*, s^*, y^*)$  is a complementarity solution of  $\text{MLCP}(A, B, C, b)$  if and only if it is feasible and  $(x^*)^T s^* = 0$ , or equivalently,  $(x^*, s^*)$  is a complementarity solution of (2.3).

(ii.) Monotonicity of  $\text{MLCP}(A, B, C, b)$  can be equivalently expressed as  $x^T s \geq 0$  whenever  $Ax + Bs \in \text{Ran}(C)$ . Since the last relation takes place if and only if  $E^T Ax + E^T Bs = 0$ , we have obtained that monotonicity of  $\text{MLCP}(A, B, C, b)$  implies monotonicity of (2.3) and viceversa.

(iii.) Suppose the opposite, *i.e.*,  $\begin{bmatrix} A^T E \\ B^T E \end{bmatrix}$  has linearly dependent columns. Then there exists  $u \in \mathbb{R}^k$ ,  $u \neq 0$  such that  $A^T E u = B^T E u = 0$ . Since we also have  $C^T E u = 0$ , we obtain that  $E u \in \text{Ker} \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix}$ .

On the other hand, since  $\begin{bmatrix} A & B & C \end{bmatrix}$  has full row rank  $m + n$ , we obtain  $\text{Ker} \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} = 0$ . Hence we must have  $E u = 0$ . Since the columns of  $E$  form an orthonormal basis,  $u$  must be zero, which is a contradiction. Hence,  $\begin{bmatrix} E^T A & E^T B \end{bmatrix}$  has full row rank  $k$ .  $\square$

The previous lemma indicates that the  $\text{MLCP}$  we proposed represents a more general framework than the well-known  $\text{HLCP}$ . More precisely, since  $\dim \text{Ker}(C^T) +$

$\dim \text{Ran}(C) = m + n$  and  $\dim \text{Ran}(C) \leq m$  implies  $k \geq n$ , the horizontal complementarity problem equivalent to the MLCP has the form

$$\begin{aligned} xs &= 0 \\ Fx + Gs &= b \\ x, s &\geq 0, \end{aligned} \tag{2.4}$$

where  $F, G \in \mathbb{R}^{k \times n}$ ,  $b \in \mathbb{R}^k$ , and  $k \geq n$ . Obviously, when  $k = n$  the HLCP (1.6) is obtained.

The monotonicity of (2.4) is defined in the same way as for HLCP, *i.e.*,  $Fx + Gs = 0$  implies  $x^T s \geq 0$ . The horizontal form (2.4) was studied by Güler in [47] in the context of maximal monotone operators. The paper shows that, in the monotone case, interior-point methods can be used to solve (2.4) if and only if  $k = n$ . We will assume that  $k = n$  from now on, which in fact implies  $\dim \text{Ran}(C) = (m+n) - \dim \text{Ker}(C^T) = m$ .

**Assumption 2.2.** *The matrix  $C$  has full column rank  $m$ .*

In [61], the authors have studied the behavior of path-following interior-point algorithms for the same form of implicitly defined mixed linear complementarity as in the present work. It has to be mentioned that Assumption 2.2 is present in their work, although no motivation is given. The article also proposes a reduction of a MLCP to a HLCP in the monotone case that is different than our reduction approach given by Lemma 2.1. While their transformation can be used in practice to solve MLCPs as HLCPs, Lemma 2.1 is vital in establishing important theoretical properties between matrices  $A$ ,  $B$  and  $C$ , as we later show.

It must be also mentioned that not only any LP and QP (except the ones with no inequality constraints which can be solved by solving a linear system) but also the

monotone MLCPs needed to simulate rigid body systems with contacts and friction can be written as a  $\text{MLCP}(A, B, C, b)$  that satisfies Assumption 2.2.

With the Assumption 2.2 in place, any monotone  $\text{MLCP}(A, B, C, b)$  can be reduced to a (standard) monotone horizontal  $\text{HLCP}(E^T A, E^T B, E^T b)$  by Lemma 2.1. HLCPs have been extensively studied in the last fifteen years and we are going to apply some of the existing theoretical results to  $\text{HLCP}(E^T A, E^T B, E^T b)$  to characterize  $\text{MLCP}(A, B, C, b)$ .

**LEMMA 2.2** (cf. Theorem 11 of [77]). *HLCP( $Q, R, b$ ) is monotone if and only if  $Q + R$  is nonsingular and  $-QR^T$  is positive semidefinite.*

Based on Lemma 2.1, an immediate consequence of the above lemma is the following corollary.

**COROLLARY 2.1.** *If  $\text{MLCP}(A, B, C, b)$  is monotone, then  $-E^T AB^T E$  is positive semidefinite.*

**LEMMA 2.3** (cf. Corollary 18 of [77]). *Suppose that  $\text{HLCP}(Q, R, b)$  is monotone. Then for each  $b \in \mathbb{R}^n$ , the feasibility of  $\text{HLCP}(Q, R, b)$  implies its solvability.*

## 2.3 An augmented homogeneous model for MLCP

Consider an augmented homogeneous complementarity problem (HMCP) related to MLCP (1.8):

$$\begin{aligned}
 xs &= 0 \\
 \tau\kappa &= 0 \\
 Ax + Bs + Cy - \tau b &= 0 \\
 \frac{\bar{x}^T \bar{s}}{\tau} + \kappa &= 0 \\
 x, \tau, s, \kappa &\geq 0,
 \end{aligned} \tag{2.5}$$

with  $[\bar{x}, \bar{s}, \bar{y}]$  is related to  $[x, s, y]$  by

$$\begin{bmatrix} \bar{x} \\ \bar{s} \\ \bar{y} \end{bmatrix} = P_{\text{Ker}[ABC]} \begin{bmatrix} x \\ s \\ y \end{bmatrix} + \tau \bar{b}, \quad (2.6)$$

where

$$\bar{b} = \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} (AA^T + BB^T + CC^T)^{-1}b \quad (2.7)$$

is the least-square solution of  $Ax + Bs + Cy = b$ .

The augmented complementarity problem (2.5) contains two additional complementary variables  $\tau$  and  $\kappa$  and one additional equation (constraint). The latter is not linear due to the newly added equation and therefore the augmented model has the form of an implicit mixed nonlinear complementarity problem (1.3) with  $F : \mathbb{R}_+^n \times \mathbb{R}_{++} \times \mathbb{R}_+^n \times \mathbb{R}_+ \times \mathbb{R}^m \rightarrow \mathbb{R}^{n+m+1}$ ,

$$F(x, \tau, s, \kappa, y) = \begin{bmatrix} Ax + Bs + Cy - \tau b \\ \bar{x}^T \bar{s} / \tau + \kappa \end{bmatrix}. \quad (2.8)$$

### 2.3.1 Notations and terminology

The concepts of feasibility and solvability as defined for the mixed linear complementarity problem are not applicable for our augmented complementarity problem because the homogenization process causes the domain of the problem not to be a closed set anymore. Even though vectors of the form  $[x, \tau, s, \kappa, y]$  with  $\tau = 0$  and the other components fixed are not part of the domain of the problem, they can be perfectly valid in an asymptotical approach, *i.e.*, when  $\tau$  positively approaches zero. We start by defining the concepts of *asymptotical* feasibility and solvability for the

generic nonlinear mixed complementarity problem in implicit form (1.3). The map  $F$  defining the complementarity problem is not necessarily defined on the boundary of  $\mathbb{R}_+^n \times \mathbb{R}_+^n \times \mathbb{R}^m$ , in other words we have  $\mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m \subseteq \text{dom}(F) \subseteq \mathbb{R}_+^n \times \mathbb{R}_+^n \times \mathbb{R}^m$ .

**DEFINITION 2.1.** *The complementarity problem defined by (1.3) is said to be asymptotically feasible if there exist bounded sequences  $\{x^k, s^k, y^k\} \subset \text{dom}(F)$ ,  $k = \{1, 2, \dots\}$  such that*

$$\lim_{k \rightarrow \infty} F(x^k, s^k, y^k) = 0.$$

*Moreover, any limit point  $(x^*, s^*, y^*)$  of the sequence  $\{x^k, s^k, y^k\}$  is called an asymptotically feasible point.*

**DEFINITION 2.2.** *Complementarity problem (1.3) is asymptotically solvable if there is an asymptotically feasible point  $(x^*, s^*, y^*)$  satisfying the complementarity conditions, i.e.,  $x^* s^* = 0$ .*

Observe that both asymptotical feasibility and asymptotical solvability would be equivalent to the corresponding concepts defined for MLCP if the domain of the problem were closed.

The study of nonlinear mixed complementarity in the context of interior-point methods employs several concepts not present in the linear case. The definitions given below have been initially introduced by Monteiro and Pang in the context of implicitly defined mixed nonlinear complementarity problems over the nonnegative orthant [61] and the cone of positive semidefinite matrices [62]. Yoshise [84] has adapted the concepts to work in an asymptotical approach needed for the study of a homogenization technique for explicit nonlinear monotone complementarity problems over symmetric cones.

We start by defining the monotonicity concept(s) for a nonlinear complementarity problem. Equilevel-monotonicity generalizes the monotonicity concept from the linear

case. However, the nonlinear complementarity problems have to satisfy a stronger type of monotonicity, *i.e.*, everywhere-monotonicity, in order to be solved by means of path-following interior-point algorithms. The two concepts are defined below.

**DEFINITION 2.3.** *The map  $F(x, s, y)$  is called  $(x, s)$ -equilevel-monotone on its domain if for any  $(x, s, y)$  and  $(x', s', y')$  that lie in the domain of  $F$  and satisfy  $F(x, s, y) = F(x', s', y')$ , it holds that  $(x - x')^T(s - s') \geq 0$ .*

**DEFINITION 2.4.** *The map  $F(x, s, y)$  is called  $(x, s)$ -everywhere-monotone on the domain of  $F$  if there exist continuous functions  $\phi$  from the domain of  $F$  to the set  $\mathbb{R}^{n+m}$  and  $c : \mathbb{R}^{n+m} \times \mathbb{R}^{n+m} \rightarrow \mathbb{R}$  such that  $c(r, r) = 0$  and*

$$(x - x')^T(s - s') \geq (r - r')^T(\phi(x, s, y) - \phi(x', s', y')) + c(r, r')$$

*holds for any  $(x, s, y)$  and  $(x', s', y')$  in the domain of  $F$  satisfying  $F(x, s, y) = r$  and  $F(x', s', y') = r'$ .*

It can be easily observed that  $(x, s)$ -everywhere-monotonicity implies  $(x, s)$ -equilevel-monotonicity if one takes  $r = r'$ .

The following two concepts are used to characterize a (desirable) behavior of the free variables in a nonlinear mixed complementarity framework.

**DEFINITION 2.5.** *The map  $F(x, s, y)$  is called  $y$ -bounded on its domain, if for any sequence  $\{(x^k, s^k, y^k)\}$  in the domain such that both  $\{(x^k, s^k)\}$  and  $\{F(x^k, s^k, y^k)\}$  are bounded sequences, the sequence  $\{y^k\}$  is also bounded.*

**DEFINITION 2.6.** *The map  $F(x, s, y)$  is called  $y$ -injective on its domain, if for any  $(x, s, y)$  and  $(x, s, y')$  lying in the domain of  $F$  and satisfying  $F(x, s, y) = F(x, s, y')$ , we have  $y = y'$ .*

We would like to remark that the above two definitions are satisfied for MLCP only under Assumption 2.2. The proof of this observation works in the same way as the proof of (iii) and (iv) of Theorem 2.4, but we omit it since it is not relevant to this discussion.

## 2.4 Properties of the augmented HMCP

This section presents the properties of the complementarity problem HMCP introduced in Section 2.3. First we prove that the augmented problem is an (everywhere-) monotone nonlinear homogeneous complementarity problem possessing the  $y$ -boundness and  $y$ -injectiveness properties. Second we show that the HMCP is solvable only under the assumption of monotonicity of the MLCP and its solution can be used as a certificate to prove the solvability or infeasibility of the original problem.

The orthogonal projection of  $[x, s, y]$  onto  $\text{Ker}[ABC]$  is essential in the sense that the transformation from MLCP to HMCP preserves the monotonicity, as shown in the following theorem. From now on we use  $[u, v, w]$  to denote the column vector  $[u^T v^T w^T]^T$ .

**LEMMA 2.4.** *The mapping  $F$  that defines the HMCP corresponding to a monotone MLCP( $A, B, C, b$ ) is*

- (i) *continuous and homogenous (of degree 1) on its domain.*
- (ii)  *$(x, s)$ -equilevel-monotone on its domain.*
- (iii)  *$y$ -bounded on its domain.*
- (iv)  *$y$ -injective on its domain.*
- (v)  *$(x, s)$ -everywhere-monotone on its domain.*

*Proof.* (i) We first show that the mappings  $[x, \tau, s, \kappa, y] \mapsto \bar{x}$ ,  $[x, \tau, s, \kappa, y] \mapsto \bar{s}$  and  $[x, \tau, s, \kappa, y] \mapsto \bar{y}$  are linear in  $[x, \tau, s, \kappa, y]$ . This is obvious once we write

$$\begin{bmatrix} \bar{x} \\ \bar{s} \\ \bar{y} \end{bmatrix} = P \begin{bmatrix} x \\ s \\ y \end{bmatrix} + \tau \bar{b} = \begin{bmatrix} P & \bar{b} & 0 \end{bmatrix} \begin{bmatrix} x \\ s \\ y \\ \tau \\ \kappa \end{bmatrix}.$$

The continuity of  $F$  readily follows from the above observation and from the fact that  $\tau > 0$  on the domain of  $F$ .

Now since  $\bar{x}$  and  $\bar{s}$  are linear functions of  $[x, \tau, s, \kappa, y]$ , we have

$$F(tx, t\tau, ts, t\kappa, ty) = \begin{bmatrix} Atx + Bts + Cty - t\tau b \\ \bar{x}^T \bar{s} / (t\tau) + t\kappa \end{bmatrix} = tF(x, \tau, s, \kappa, y), \quad \forall t \in \mathbb{R}$$

and hence  $F$  is homogeneous.

(ii) It is a consequence of (iv). Take  $c = 0$  and  $r = r'$  in the Definition (2.4) of everywhere-monotonicity to obtain Definition (2.3) of equilevel-monotonicity.

(iii) Consider the sequence  $\{(x^k, \tau^k, s^k, \kappa^k, y^k)\}$  in the domain of  $F$  such that  $\{(x^k, \tau^k, s^k, \kappa^k)\}$  and  $\{F(x^k, \tau^k, s^k, \kappa^k, y^k)\}$  are bounded.

Since  $C$  has full column rank we can write

$$\begin{aligned} \|y^k\| &= \|(C^T C)^{-1} C^T C y^k\| \leq \|(C^T C)^{-1} C^T\| \|C y^k\| \\ &= \|(C^T C)^{-1} C^T\| \|(Ax^k + Bs^k + Cy^k - \tau^k b) - (Ax^k + Bs^k - \tau^k b)\| \\ &\leq \|(C^T C)^{-1} C^T\| (\|Ax^k + Bs^k + Cy^k - \tau^k b\| + \|Ax^k + Bs^k - \tau^k b\|) \\ &\leq \|(C^T C)^{-1} C^T\| \left( M_1 + \left\| \begin{bmatrix} A & B & -b \end{bmatrix} \right\| M_2 \right), \end{aligned}$$

where  $M_1$  and  $M_2$  are the bounds for  $\{F(x^k, \tau^k, s^k, \kappa^k, y^k)\}$  and  $\{(x^k, \tau^k, s^k, \kappa^k)\}$ , respectively. Therefore  $\{y^k\}$  is bounded which implies that  $F$  is  $y$ -bounded according

to Definition 2.5.

(iv) If  $F(x, \tau, s, \kappa, y) = F(x, \tau, s, \kappa, y')$ , then  $Cy = Cy'$ , which implies  $y = y'$  since  $C$  is assumed to have full column rank.

(v) Consider  $(x, s, y)$  and  $(x', s', y')$  in the domain of  $F$  and let

$$\begin{bmatrix} r \\ \gamma \end{bmatrix} = \begin{bmatrix} Ax + Bs + Cy - \tau b \\ \bar{x}^T \bar{s} / \tau + \kappa \end{bmatrix}$$

and

$$\begin{bmatrix} r' \\ \gamma' \end{bmatrix} = \begin{bmatrix} Ax' + Bs' + Cy' - \tau' b \\ \bar{x}'^T \bar{s}' / \tau' + \kappa' \end{bmatrix}.$$

Since  $P_{\text{Ker}[ABC]} = I - P_{\text{Ran}[ABC]^T} = I - \begin{bmatrix} A & B & C \end{bmatrix}^T (AA^T + BB^T + CC^T)^{-1} \begin{bmatrix} A & B & C \end{bmatrix}$ , the equation (2.6) that defines  $(\bar{x}, \bar{s}, \bar{y})$  is equivalent to

$$\begin{aligned} \begin{bmatrix} \bar{x} \\ \bar{s} \\ \bar{y} \end{bmatrix} &= \begin{bmatrix} x \\ s \\ y \end{bmatrix} - \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} (AA^T + BB^T + CC^T)^{-1} (Ax + Bs + Cy) + \tau \bar{b} \\ &= \begin{bmatrix} x \\ s \\ y \end{bmatrix} - \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} (AA^T + BB^T + CC^T)^{-1} (Ax + Bs + Cy - \tau b), \end{aligned}$$

where the expression (2.7) of  $\bar{b}$  was used to obtain the last equality.

By using the expression of  $r$  and manipulating the terms in the above equation, we obtain

$$\begin{bmatrix} x \\ s \\ y \end{bmatrix} = \begin{bmatrix} \bar{x} \\ \bar{s} \\ \bar{y} \end{bmatrix} + \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} Mr,$$

where  $M := (AA^T + BB^T + CC^T)^{-1}$ . Observe that  $M$  is a symmetric (and also positive definite) matrix.

Similarly,

$$\begin{bmatrix} x' \\ s' \\ y' \end{bmatrix} = \begin{bmatrix} \bar{x}' \\ \bar{s}' \\ \bar{y}' \end{bmatrix} + \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} M r'.$$

Using the above two expressions for  $(x, s, y)$  and  $(x', s', y')$  we can now compute

$$\begin{aligned} (x - x')^T (s - s') &= (\bar{x} - \bar{x}' + A^T M (r - r'))^T (\bar{s} - \bar{s}' + B^T M (r - r'))^T \\ &= (\bar{x} - \bar{x}')^T (\bar{s} - \bar{s}') + (r - r')^T M (A(\bar{s} - \bar{s}') + B(\bar{x} - \bar{x}')) \\ &\quad + (r - r')^T M A B^T M (r - r') \\ &= (\bar{x} - \bar{x}')^T (\bar{s} - \bar{s}') + (r - r')^T M (A(s - s') + B(x - x')) \\ &\quad - (r - r')^T M (A B^T M (r - r') + B A^T M (r - r')) \\ &\quad + (r - r')^T M A B^T M (r - r') \\ &= (\bar{x} - \bar{x}')^T (\bar{s} - \bar{s}') + (r - r')^T M (A(s - s') + B(x - x')) \\ &\quad - (r - r')^T M A B^T M (r - r') \\ &= (\bar{x} - \bar{x}')^T (\bar{s} - \bar{s}') \\ &\quad + (r - r')^T M \{A(s - s') + B(x - x') - A B^T M [A(x - x') \\ &\quad + B(s - s') + C(y - y') - (\tau - \tau')b]\}. \end{aligned} \quad (2.9)$$

By multiplying (2.6) with  $\begin{bmatrix} A & B & C \end{bmatrix}$  and using (2.7) we obtain that  $A\bar{x} + B\bar{s} + C\bar{y} = \tau b$  and  $A\bar{x}' + B\bar{s}' + C\bar{y}' = \tau' b$ . We can then write  $A\bar{x}/\tau + B\bar{s}/\tau + C\bar{y}/\tau = b = A\bar{x}'/\tau' + B\bar{s}'/\tau' + C\bar{y}'/\tau'$  so  $(\bar{x}/\tau - \bar{x}'/\tau')^T (\bar{s}/\tau - \bar{s}'/\tau') \geq 0$  holds by monotonicity of  $\text{MLCP}(A, B, C, b)$ . The following inequality is obtained by multiplying the previous inequality with  $\tau\tau'$  and manipulating the terms:

$$\frac{\tau'}{\tau} \bar{x}^T \bar{s} + \frac{\tau}{\tau'} \bar{x}'^T \bar{s}' \geq \bar{x}'^T \bar{s} + \bar{x}^T \bar{s}'. \quad (2.10)$$

By using the expressions of  $\gamma$  and  $\gamma'$  we can write that  $(\tau - \tau')(\kappa - \kappa') = (\tau - \tau')(\gamma -$

$$\gamma') - (\tau - \tau')(\bar{x}^T \bar{s} / \tau - \bar{x}'^T \bar{s}' / \tau') = (\tau - \tau')(\gamma - \gamma') - (\bar{x}^T \bar{s} + \bar{x}'^T \bar{s}') + (\frac{\tau'}{\tau} \bar{x}^T \bar{s} + \frac{\tau}{\tau'} \bar{x}'^T \bar{s}').$$

Inequality (2.10) implies that

$$\begin{aligned} (\tau - \tau')(\kappa - \kappa') &\geq (\tau - \tau')(\gamma - \gamma') - (\bar{x}^T \bar{s} + \bar{x}'^T \bar{s}') + \bar{x}'^T \bar{s} + \bar{x}^T \bar{s}' \\ &= (\tau - \tau')(\gamma - \gamma') - (\bar{x} - \bar{x}')^T (\bar{s} - \bar{s}'). \end{aligned} \quad (2.11)$$

By adding (2.9) and (2.11) the following inequality is obtained

$$\begin{bmatrix} x - x' \\ \tau - \tau' \end{bmatrix}^T \begin{bmatrix} s - s' \\ \kappa - \kappa' \end{bmatrix} \geq \begin{bmatrix} r - r' \\ \gamma - \gamma' \end{bmatrix}^T (\phi(x, \tau, s, \kappa, y) - \phi(x', \tau', s', \kappa', y')),$$

where  $\phi : (\mathbb{R}_+^n \times \mathbb{R}_{++} \times \mathbb{R}_+^n \times \mathbb{R}_+) \times \mathbb{R}^m \rightarrow \mathbb{R}^{n+m+1}$  is given by

$$\phi(x, \tau, s, \kappa, y) = \begin{bmatrix} M(As + Bx) - MAB^T M(Ax + Bs + Cy - \tau b) \\ \tau \end{bmatrix}. \quad (2.12)$$

The function  $\phi$  is clearly continuous and by considering  $c := 0$  it can be easily seen that  $F$  is  $(x, s)$ -everywhere-monotone on  $(\mathbb{R}_+^n \times \mathbb{R}_{++} \times \mathbb{R}_+^n \times \mathbb{R}_+) \times \mathbb{R}^m$ , according to Definition (2.4).  $\square$

Furthermore we have the following theorem that shows that HMCP is always feasible and solvable.

**THEOREM 2.1.** *HMCP is asymptotically feasible and every asymptotically feasible point is an asymptotically complementarity solution.*

*Proof.* It can be easily verified that HMCP is asymptotically feasible by considering the sequence  $(x^l, \tau^l, s^l, \kappa^l, y^l) := ((1/2)^l e, (1/2)^l, (1/2)^l e, (1/2)^l, 0)$  and letting  $l \rightarrow \infty$ .

Let  $(x, \tau, s, \kappa, y)$  be any asymptotically feasible point of HMCP, *i.e.*:

$$\begin{aligned} Ax + Bs + Cy - \tau b &= 0 \\ \frac{\bar{x}^T \bar{s}}{\tau} + \kappa &= 0 \\ x, \tau, s, \kappa &\geq 0. \end{aligned} \quad (2.13)$$

Since

$$\begin{aligned} [\bar{x}, \bar{s}, \bar{y}] - [x, s, y] &= \tau\bar{b} - (I - P)[x, s, y] = \tau\bar{b} - P_{\text{Ran}[ABC]^T}[x, s, y] \\ &= [ABC]^T(AA^T + BB^T + CC^T)^{-1}(\tau b - Ax + Bs + Cy), \end{aligned}$$

the first feasibility equation from (2.13) implies that  $x = \bar{x}$ ,  $s = \bar{s}$  and  $y = \bar{y}$ . Then the second feasibility equation from (2.13) yields the complementarity condition

$$x^T s + \tau\kappa = 0.$$

□

The next theorem shows how the solutions of the original problem MLCP and homogeneous problem HMCP are related. More exactly, the solutions to HMCP represents certificates of solvability or infeasibility of the MLCP. Moreover, in the case when MLCP is solvable, a solution is obtained from the solution of the HMCP at no cost.

**THEOREM 2.2.** *Let  $(x^*, \tau^*, s^*, \kappa^*, y^*)$  be an asymptotical complementarity solution of the HMCP corresponding to a monotone MLCP. Then the following statements hold:*

(i) *MLCP has a solution if and only if  $\tau^* > 0$ . In this case,  $(x^*/\tau^*, s^*/\tau^*, y^*/\tau^*)$  is a complementarity solution for MLCP.*

(ii) *MLCP is infeasible if and only if  $\kappa^* > 0$ .*

*Proof.* (i) If  $(x^*, \tau^*, s^*, \kappa^*, y^*)$  is a solution for HMCP and  $\tau^* > 0$ , then

$$Ax^*/\tau^* + Bs^*/\tau^* + Cy^*/\tau^* = \tau^*b/\tau^* = b$$

and

$$(x^*/\tau^*)^T (s^*/\tau^*) = \frac{x^{*T} s^*}{\tau^{*2}} = 0,$$

that is,  $(x^*/\tau^*, s^*/\tau^*, y^*/\tau^*)$  is solution for MLCP.

Now let  $(x^*, s^*, y^*)$  be a solution for MLCP. We show that  $\hat{x} = x^*$ ,  $\hat{\tau} = 1$ ,  $\hat{s} = s^*$ ,  $\hat{\kappa} = 0$  and  $\hat{y} = y^*$  is a solution for HMCP. The two complementarity conditions of HMCP are obviously satisfied as well as the first feasibility condition. As in the proof of Theorem 2.1, one can obtain that the first feasibility condition for  $(\hat{x}, \hat{\tau}, \hat{s}, \hat{\kappa}, \hat{y})$  implies  $\bar{\hat{x}} = \hat{x}$ ,  $\bar{\hat{s}} = \hat{s}$  and  $\bar{\hat{y}} = \hat{y}$ . Then we can write

$$\frac{\bar{\hat{x}}^T \bar{\hat{s}}}{\bar{\hat{\tau}}} + \bar{\hat{\kappa}} = \frac{\hat{x}^T \hat{s}}{1} + 0 = x^{*T} s^* = 0,$$

which proves that the second feasibility equation of the HMCP holds and, hence, completes the proof of (i).

We now prove (ii). First we show that if  $(x^*, 0, s^*, \kappa^*, y^*)$ , with  $\kappa^* > 0$ , is an asymptotical solution for HMCP, then MLCP is infeasible. Assume the opposite, *i.e.*, there exist  $x \geq 0$ ,  $s \geq 0$  and  $y \in \mathbb{R}^m$  such that  $Ax + Bs + Cy = b$ .

Since  $(x^*, 0, s^*, \kappa^*, y^*)$  asymptotically solves the HMCP, we can consider the sequences  $x_k \geq 0$  with  $x_k \rightarrow x^*$ ,  $\tau_k > 0$  with  $\tau_k \rightarrow 0$ ,  $s_k \geq 0$  with  $s_k \rightarrow s^*$ ,  $y_k \rightarrow y^*$  and  $\kappa_k \geq 0$  with  $\kappa_k \rightarrow \kappa^* > 0$  satisfying

$$Ax_k + Bs_k + Cy_k \rightarrow \tau_k b \tag{2.14}$$

$$\bar{x}_k^T \bar{s}_k / \tau_k \rightarrow -\kappa^*. \tag{2.15}$$

As in the proof of Theorem 2.1, (2.14) implies

$$\bar{x}_k \rightarrow x^* \text{ and } \bar{s}_k \rightarrow s^*. \tag{2.16}$$

On the other hand, the left multiplication of  $[\bar{x}_k, \bar{s}_k, \bar{y}_k]$  from (2.6) with  $\begin{bmatrix} A & B & C \end{bmatrix}$  causes the orthogonal projection term to vanish and we have

$$[ABC][\bar{x}_k, \bar{s}_k, \bar{y}_k] = \tau_k [ABC] \bar{b} = \tau_k b,$$

or, since  $\tau_k > 0$

$$A\bar{x}_k/\tau_k + B\bar{s}_k/\tau_k + C\bar{y}_k/\tau_k = b.$$

Then we can write  $A(\bar{x}_k/\tau_k - x) + B(\bar{s}_k/\tau_k - s) + C(\bar{y}_k/\tau_k - y) = 0$ , which implies  $(\bar{x}_k/\tau_k - x)^T(\bar{s}_k/\tau_k - s) \geq 0$  by the monotonicity property of MLCP, and therefore  $\tau_k x^T s - (x^T \bar{s}_k + s^T \bar{x}_k) \geq -\bar{x}_k^T \bar{s}_k/\tau_k$ . By considering the limit when  $\tau_k \rightarrow 0$  and taking into account (2.15) and (2.16), we obtain that  $\kappa^* \leq -(x^T s^* + s^T x^*) \leq 0$ , which is a contradiction with the fact that  $\kappa^*$  is positive.

Conversely, assume that MLCP is infeasible. Want to prove that there is a complementarity solution  $(x^*, \tau^*, s^*, \kappa^*, y^*)$  of the HMCP that has  $\kappa^* > 0$ .

Consider the set  $\mathcal{P} = \{Ax + Bs + Cy - b : x, s \geq 0, y \in \mathbb{R}^m\}$ . It can be easily verified that  $\mathcal{P}$  is convex. Also the set  $b + \mathcal{P}$  is a finitely generated cone, hence closed. Then  $\mathcal{P}$  is closed.

The infeasibility of MLCP( $A, B, C, b$ ) is equivalent to  $0 \notin \mathcal{P}$ . Since  $\mathcal{P}$  is closed and convex, then there must be a separating hyperplane between  $0$  and  $\mathcal{P}$ , that is, there is a vector  $a \in \mathbb{R}^{m+n}$ ,  $a \neq 0$  and  $\xi > 0$  so that

$$a^T(Ax + Bs + Cy - b) \geq \xi > 0, \forall x, s \geq 0, \forall y \in \mathbb{R}^m. \quad (2.17)$$

Let us take  $s = 0$  and  $y = 0$  in (2.17). Then for any  $x \geq 0$  we must have  $x^T A^T a = a^T Ax \geq \xi + a^T b$ . If the  $j^{\text{th}}$  component of  $A^T a$  is negative, then  $x^T A^T a$  can be made smaller than  $\xi + a^T b$  by taking  $x_i = 0$  for  $i \neq j$  and  $x_j$  sufficiently large. Hence

$$A^T a \geq 0. \quad (2.18)$$

Similarly one can obtain that

$$B^T a \geq 0. \quad (2.19)$$

Taking  $x = s = 0$  in (2.17) leads to  $y^T C^T a \geq \xi + a^T b$  for any  $y \in \mathbb{R}^m$ . This implies that

$$C^T a = 0 \quad (2.20)$$

because the functional  $y \mapsto c^T y$  is bounded below if and only if  $c = 0$ . We obtained in fact that  $a \in \text{Ker}(C^T)$  and we can write  $a = Eu$  with  $u \in \mathbb{R}^n$ , where  $E \in \mathbb{R}^{(m+n) \times n}$  is the matrix introduced in Lemma 2.1 (its columns represent an orthonormal basis of  $\text{Ker}(C^T)$ ).

We have  $(A^T a)^T (B^T a) = u^T E^T A B^T E u \leq 0$  since  $-E^T A B^T E$  is positive semidefinite (see Lemma 2.1). On the other hand (2.18) and (2.19) imply  $(A^T a)^T (B^T a) \geq 0$ . Thus

$$(A^T a)^T (B^T a) = 0. \quad (2.21)$$

Furthermore, since the matrix  $-E^T A B^T E$  is positive semidefinite according to Lemma 2.1, equation (2.21) indicates that the nonnegative function  $q \mapsto -q^T E^T A B^T E q$  vanishes at  $u$ , which implies by the first order optimality conditions that  $E^T A B^T E u + (E^T A B^T E)^T u = 0$ , or, equivalently,  $E^T (A B^T a + B A^T a) = 0$ . Hence  $A B^T a + B A^T a \in \text{Ran}(C)$ . Therefore we have that  $A B^T a + B A^T a + C y' = 0$ , with  $y' \in \mathbb{R}^m$ . By denoting  $x' = B^T a$  and  $s' = A^T a$  we can write that

$$A x' + B s' + C y' = 0, \text{ with } x', s' \geq 0, y' \in \mathbb{R}^m \text{ and } (x')^T s' = 0. \quad (2.22)$$

Now consider  $x(t) = x' + t u^*$ ,  $\tau(t) = t$ ,  $s(t) = s' + t v^*$  and  $y(t) = y' + t w^*$ , for  $t > 0$ , where  $[u^*, v^*, w^*]$  is the block representation of the least square solution  $\bar{b}$  to  $Ax + Bs + Cy = b$  (see (2.7)). Observe based on (2.22) that

$$\lim_{t \rightarrow 0} A \frac{x(t)}{t} + B \frac{s(t)}{t} + C \frac{y(t)}{t} = A u^* + B v^* + C w^* = b \quad (2.23)$$

and

$$\begin{bmatrix} x(t) \\ s(t) \\ y(t) \end{bmatrix} = P \begin{bmatrix} x' + tu^* \\ s' + tv^* \\ y' + tw^* \end{bmatrix} + t \begin{bmatrix} u^* \\ v^* \\ w^* \end{bmatrix} = \begin{bmatrix} x' \\ s' \\ y' \end{bmatrix} + t \begin{bmatrix} u^* \\ v^* \\ w^* \end{bmatrix},$$

with the first equality being the definition (2.6), and the last equality holding because  $[x', s', y'] \in \text{Ker}[ABC]$  (see (2.22)) and  $[u^*, v^*, w^*] \in \text{Ran}[ABC]^T$ . Then we can write

$$\begin{aligned} \lim_{t \rightarrow 0} \frac{x(t)^T s(t)}{t} &= \lim_{t \rightarrow 0} \frac{(x' + tu^*)^T (s' + tv^*)}{t} \stackrel{(2.22)}{=} (s')^T u^* + (x')^T v^* \\ &= a^T (Au^* + Bv^*) \stackrel{(2.20)}{=} a^T (Au^* + Bv^* + Cw^*) = a^T b. \end{aligned}$$

We have proved that  $(A^T a, 0, B^T a, -a^T b, y')$  is an asymptotical solution of the HMCP. If we take  $x, s$  and  $y$  to be zero in (2.17) then  $-a^T b \geq \xi > 0$ , showing that there is a solution  $(x^*, \tau^*, s^*, \kappa^*, y^*)$  with  $\kappa^* > 0$ , *e.g.*  $(A^T a, 0, B^T a, -a^T b, y')$ .  $\square$

If all the solutions  $(x^*, \tau^*, s^*, \kappa^*, y^*)$  to HMCP have  $\tau^* = 0$  and  $\kappa^* = 0$ , then the MLCP is not infeasible (hence it is feasible) and is not solvable. This can not hold in the monotone case since, according to Lemma 2.3, feasibility implies solvability. Hence Theorem 2.2 implies that the HMCP must have a solution for which either  $\tau^* > 0$  and  $\kappa^* = 0$  or  $\tau^* = 0$  and  $\kappa^* > 0$ . We would like to mention that the homogeneous model for monotone complementarity problems (in explicit form) of Andersen and Ye [3] possesses the same type of strict complementarity of homogenization variables when applied to an affine monotone function, *i.e.*, to a SLCP.

Suppose  $(x^*, \tau^*, s^*, \kappa^*, y^*)$  is a maximal complementarity solution to HMCP. Table 2.1 displays all possible combinations of  $\tau^*$  and  $\kappa^*$  and the corresponding status of the solvability of MLCP. "NA" stands for "not available".

$\tau^* \setminus \kappa^*$	$= 0$	$> 0$
$= 0$	NA	MLCP is infeasible
$> 0$	$(x^*/\tau^*, s^*/\tau^*, y^*/\tau^*)$ is a solution to MLCP	NA

Table 2.1: Possible combination of optimal  $\tau^*$  and  $\kappa^*$ 

## 2.5 A general theory on the existence of central paths for nonlinear complementarity problems

In this section we briefly present several results concerning the properties of an interior-point mapping which we later use to characterize the central path of the HMCP. The results are part of a framework that has been introduced by Monteiro and Pang [61] to study nonlinear monotone implicitly defined complementarity problems over the non-negative orthant and over the cone of symmetric positive semidefinite matrices [62]. Recently Yoshise [84] has managed to prove the same type of results for nonlinear monotone implicitly defined complementarity problems over symmetric cones.

While the analysis from [61, 62] requires the complementarity problem to be defined on the entire cone, the results from [84] can be used for complementarity problems not defined on the boundary of the cone. The latter is our case and we state the results from [84].

As we mentioned above, Yoshise's results hold for symmetric cones. Here we specialize them to the nonnegative orthant which is a symmetric cone. Consider a nonlinear complementarity problem in implicit form, *i.e.*,

$$xs = 0, F(x, s, y) = 0, x, s \geq 0, \quad (2.24)$$

where  $F$  is a continuous map into  $\mathbb{R}^{m+n}$  satisfying  $\mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m \subseteq \text{dom}(F) \subseteq \mathbb{R}_+^n \times \mathbb{R}_+^n \times \mathbb{R}^m$ .

Furthermore, the trajectory of the interior point map  $H : \mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^{2n+m}$  given by

$$H(x, s, y) = \begin{bmatrix} xs \\ F(x, s, y) \end{bmatrix} \quad (2.25)$$

is characterized by means of homeomorphic continuous maps in the following theorem.

**THEOREM 2.3** (cf. Theorem 3.10 of [84]). *Suppose that the continuous map  $F$  is  $(x, s)$ -equilevel-monotone,  $y$ -bounded and  $y$ -injective on its domain. Then the map  $H$  defined by (2.25) satisfies the following properties:*

- (i)  $H$  is proper with respect to  $\mathbb{R}_{++}^n \times F(\mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m)$ .
- (ii)  $H$  maps  $\mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m$  homeomorphically onto  $\mathbb{R}_{++}^n \times F(\mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m)$ .

Under the  $(x, s)$ -everywhere monotonicity assumption on  $F$ , the set  $F(\mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m)$  is convex and open as shown by the following theorem. The convexity of this set turns out to be a key property in Section 2.6 in proving crucial properties of the central path associated with our homogeneous model.

**THEOREM 2.4** (cf. Theorem 3.12 of [84]). *Suppose that the continuous map  $F$  is  $(x, s)$ -everywhere-monotone,  $y$ -bounded and  $y$ -injective on its domain. Then the set  $F(\mathbb{R}_{++}^n \times \mathbb{R}_{++}^n \times \mathbb{R}^m)$  is an open convex set.*

## 2.6 Existence and properties of the central path for HMCP

In this section we prove the existence, uniqueness and convergence of central path for the homogenous model HMCP. We also show that sequences of interior points generated by path-following interior-point algorithms have limit points which are solutions to HMCP.

Consider the map

$$H(x, \tau, s, \kappa, y) = \begin{bmatrix} xs \\ \tau\kappa \\ F(x, \tau, s, \kappa, y) \end{bmatrix} \quad (2.26)$$

and choose a strictly feasible initial point  $(x^0, \tau^0, s^0, \kappa^0, y^0)$ . For simplicity we set  $(x^0, \tau^0, s^0, \kappa^0, y^0) := (e, 1, e, 1, 0)$ . Define

$$\begin{bmatrix} \hat{p}^0 \\ \hat{r}^0 \end{bmatrix} := H(x^0, \tau^0, s^0, \kappa^0, y^0) = \begin{bmatrix} e \\ 1 \\ F(e, 1, e, 1, 0) \end{bmatrix}.$$

**THEOREM 2.5.** *If  $\text{MLCP}(A, B, C, b)$  is monotone, then the following statements hold for the corresponding HMCP problem:*

- (i) *For any  $t \in (0, 1]$  there exist  $x(t) > 0$ ,  $\tau(t) > 0$ ,  $s(t) > 0$ ,  $\kappa(t) > 0$ , and  $y(t) \in \mathbb{R}^m$  such that*

$$H(x(t), \tau(t), s(t), \kappa(t), y(t)) = t \begin{bmatrix} \hat{p}^0 \\ \hat{r}^0 \end{bmatrix}. \quad (2.27)$$

- (ii) *The set  $\mathcal{P}$  containing all the points  $(x(t), \tau(t), s(t), \kappa(t), y(t))$  given by (i) forms a bounded path in  $\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m$ . Moreover, any accumulation point  $(x(0), \tau(0), s(0), \kappa(0), y(0))$  is an asymptotical solution of HMCP.*

*Proof.* (i) According to Lemma 2.4,  $F$  satisfies the conditions of Theorem 2.4. Thus, the set  $F(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m)$  is open and convex. Since

$$H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m) = \mathbb{R}_{++}^{n+1} \times F(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m),$$

we obtain that  $H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m)$  is also open and convex.

HMCP is asymptotically feasible by Theorem 2.1, *i.e.*,  $0 \in \text{cl}(F(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m))$ . Since

$$\text{cl}(H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m)) = \mathbb{R}_+^{n+1} \times \text{cl}(F(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m)),$$

it is clear that  $0 \in \text{cl}(H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m))$  also holds.

$H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m)$  being open and convex and  $0 \in \text{cl}(H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m))$  imply that  $t \begin{bmatrix} \hat{p}^0 \\ \hat{r}^0 \end{bmatrix} \in H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m)$  for all  $t \in (0, 1]$ . Then the conclusion from (i) follows from the fact that the map  $H$  is a homeomorphism from  $\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m$  onto  $H(\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m)$  (according to Theorem 2.3).

(ii) The homeomorphism of  $H$  also implies that  $\mathcal{P}$  is a path in  $\mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m$ .

We now prove the boundedness of  $\mathcal{P}$ . Assume  $(x(t), \tau(t), s(t), \kappa(t), y(t)) \in \mathcal{P}$ . Then  $F(x(t), \tau(t), s(t), \kappa(t), y(t)) = t\hat{r}^0 = tF(x^0, \tau^0, s^0, \kappa^0, y^0)$  and by homogeneity of  $F$  we obtain  $F(x(t), \tau(t), s(t), \kappa(t), y(t)) = F(tx^0, t\tau^0, ts^0, t\kappa^0, ty^0)$ . According to Lemma 2.4,  $F$  is equilevel-monotone, therefore we must have

$$(x(t) - tx^0)^T (s(t) - ts^0) + (\tau(t) - t\tau^0)(\kappa(t) - t\kappa^0) \geq 0,$$

or equivalently,

$$x(t)^T s^0 + s(t)^T x^0 + \tau(t)\kappa^0 + \kappa(t)\tau^0 \leq \frac{x(t)^T s(t)}{t} + \frac{\tau(t)\kappa(t)}{t} + t(x^0)^T s^0 + t\tau^0\kappa^0.$$

Moreover, any point on the path must have  $x(t)s(t) = te$  and  $\tau(t)\kappa(t) = t$ . Observe that the first equality gives  $x(t)^T s(t) = tn$ . Also we have  $(x^0)^T s^0 = n$  and  $\tau^0\kappa^0$ . Then

the above inequality transforms to

$$\begin{bmatrix} x(t) \\ \tau(t) \\ s(t) \\ \kappa(t) \end{bmatrix}^T \begin{bmatrix} e \\ 1 \\ e \\ 1 \end{bmatrix} \leq (n+1)(t+1) \leq 2(n+1).$$

But  $x(t) > 0$ ,  $\tau(t) > 0$ ,  $s(t) > 0$ ,  $\kappa(t) > 0$ , hence the above inequality forces each component of the vector  $\lceil (x(t), \tau(t), s(t), \kappa(t)) \rceil$  to be less than  $2(n+1)$ .

We have proved that  $\lceil (x(t), \tau(t), s(t), \kappa(t)) \rceil$  is bounded. Since we also have  $\|F(x(t), \tau(t), s(t), \kappa(t), y(t))\| = t \|\hat{r}^0\| \leq \|\hat{r}^0\|$ ,  $y(t)$  must be also bounded according to the  $y$ -boundedness of  $F$  (see Lemma 2.4). Hence the set  $\mathcal{P}$  is bounded.

Since  $\mathcal{P}$  is bounded at least one accumulation point  $(x(0), \tau(0), s(0), \kappa(0), y(0))$  must exist. Finally, by making use of the expression (2.27) of central path  $\mathcal{P}$  and the Definition 2.2 of an asymptotical solution, we conclude that any accumulation point is an asymptotical solution of HMCP.  $\square$

The following theorem proves that any solution to HMCP found by means of a path-following interior-point algorithm possesses the maximal complementarity property.

**THEOREM 2.6.** *Let  $\text{MLCP}(A, B, C, b)$  be monotone,  $(z^* := (x^*, \tau^*, s^*, \kappa^*), y^*)$  be an asymptotical complementarity solution of the corresponding HMCP, and also  $(z(0) := (x(0), \tau(0), s(0), \kappa(0)), y(0))$  be any accumulation point of the path  $\mathcal{P}$ . If  $z_i^* > 0$ ,  $i \in \{1, 2, \dots, 2n+2\}$ , then the  $i^{\text{th}}$  component  $[z(0)]_i$  of  $z(0)$  must also be positive.*

*Proof.* Consider  $t \in (0, 1]$ , and the corresponding point  $(z(t) := (x(t), \tau(t), s(t), \kappa(t)),$

$y(t) \in \mathcal{P}$  for which we have

$$\begin{aligned} \begin{bmatrix} r(t) \\ \gamma(t) \end{bmatrix} &:= F(x(t), \tau(t), s(t), \kappa(t), y(t)) = t \begin{bmatrix} r^0 \\ \gamma^0 \end{bmatrix} \\ x(t)s(t) &= te \\ \tau(t)\kappa(t) &= t, \end{aligned} \quad (2.28)$$

where  $\begin{bmatrix} r^0 \\ \gamma^0 \end{bmatrix} = \hat{r}^0$ .

Since  $(z^*, y^*)$  is an asymptotical solution of HMCP, there must be a sequence  $\{(z^k := (x^k, \tau^k, s^k, \kappa^k), y^k)\} \subset \mathbb{R}_{++}^{n+1} \times \mathbb{R}_{++}^{n+1} \times \mathbb{R}^m, k \in \{1, 2, \dots\}$  such that

$$\begin{aligned} (x^k, \tau^k, s^k, \kappa^k, y^k) &\rightarrow (x^*, \tau^*, s^*, \kappa^*, y^*) \\ \begin{bmatrix} r^k \\ \gamma^k \end{bmatrix} &:= F(x^k, \tau^k, s^k, \kappa^k, y^k) \rightarrow 0 \\ x^k s^k &\rightarrow x^* s^* = 0 \\ \tau^k \kappa^k &\rightarrow \tau^* \kappa^* = 0. \end{aligned} \quad (2.29)$$

The sequence  $\{z^k\}$  is bounded since it is convergent. Moreover, by Theorem 2.5 the set  $\mathcal{P}$  is also bounded, therefore there must be  $\epsilon > 0$  such that

$$\begin{aligned} \|z^k\| &\leq 1/\epsilon, \quad \forall k \text{ and} \\ \|z(t)\| &\leq 1/\epsilon, \quad \forall t \in (0, 1]. \end{aligned} \quad (2.30)$$

For a fixed  $t \in (0, 1]$ , (2.29) also implies that there exists  $k(t)$  positive integer such that

$$\begin{aligned} x_i^k s_i^k &< t\epsilon/(n+1), \quad i \in \{1, 2, \dots, 2n+m+2\}, \\ \tau^k \kappa^k &< t\epsilon/(n+1), \quad \forall k \geq k(t), \\ \|r^k\| &< t\epsilon, \end{aligned}$$

which implies that

$$(x^k)^T s^k + \tau^k \kappa^k < t\epsilon \text{ and } \|r^k\| < t\epsilon, \quad \forall k \leq k(t). \quad (2.31)$$

Since  $F$  is everywhere-monotone we can write

$$\begin{bmatrix} x^k - x(t) \\ \tau^k - \tau(t) \end{bmatrix}^T \begin{bmatrix} s^k - s(t) \\ \kappa^k - \kappa(t) \end{bmatrix} \geq \begin{bmatrix} r^k - r(t) \\ \gamma^k - \gamma(t) \end{bmatrix}^T (\phi(z^k) - \phi(z(t))),$$

where  $\phi$  is the continuous *linear* function given by (2.12). By manipulating the terms in the above inequality, we obtain that

$$\begin{aligned} s(t)^T x^k + x(t)^T s^k + \kappa(t)\tau^k + \tau(t)\kappa^k &\leq (x^k)^T s^k + x(t)^T s(t) + \tau^k \kappa^k + \tau(t)\kappa(t) \\ &\quad + \begin{bmatrix} r^k - r(t) \\ \gamma^k - \gamma(t) \end{bmatrix}^T (\phi(z(t)) - \phi(z^k)), \end{aligned}$$

and by using  $[x(t)]_i [s(t)]_i = t$ ,  $i \in \{1, 2, \dots, 2n+2\}$ , and  $\tau(t)\kappa(t) = t$  given by (2.28), we can transform the previous inequality to

$$\begin{aligned} t(z^k)^T z(t)^{-1} &\leq (x^k)^T s^k + tn + \tau^k \kappa^k + t + \begin{bmatrix} r^k - tr^0 \\ \gamma^k - t\gamma^0 \end{bmatrix}^T (\phi(z(t)) - \phi(z^k)) \\ &\leq (x^k)^T s^k + \tau^k \kappa^k + t(n+1) + \left\| \begin{bmatrix} r^k - tr^0 \\ \gamma^k - t\gamma^0 \end{bmatrix} \right\| \|\phi(z^k) - \phi(z(t))\| \\ &\leq (x^k)^T s^k + \tau^k \kappa^k + t(n+1) + \\ &\quad + \left( \left\| \begin{bmatrix} r^k \\ \gamma^k \end{bmatrix} \right\| + t \left\| \begin{bmatrix} r^0 \\ \gamma^0 \end{bmatrix} \right\| \right) \|\phi\| (\|z^k\| + \|z(t)\|) \quad (\text{by linearity of } \phi) \\ &\leq t\epsilon + t(n+1) + t \left( \epsilon + \left\| \begin{bmatrix} r^0 \\ \gamma^0 \end{bmatrix} \right\| \right) \|\phi\| (1/\epsilon + 1/\epsilon), \end{aligned}$$

where the last inequality follows by applying (2.30) and (2.31).

To conclude, we have proved that

$$\forall t \in (0, 1], \exists k(t) \text{ such that } (z^k)^T z(t)^{-1} \leq M, \quad \forall k \geq k(t),$$

where  $M := n + 1 + \epsilon + 2 \left( \epsilon + \left\| \begin{bmatrix} r^0 \\ \gamma^0 \end{bmatrix} \right\| \right) \|\phi\| / \epsilon$  does not depend on neither  $t$  or  $k(t)$ . By the convergence of  $\{z^k\}$  to  $z^*$ , we obtain that

$$(z^*)^T z(t)^{-1} \leq M, \quad \forall t \in (0, 1]. \quad (2.32)$$

Consider  $z_i^* > 0$ . If the accumulation point  $z(0)$  of  $\mathcal{P}$  satisfies  $[z(0)]_i = 0$ , then a sequence  $\{t^l\}$  of positive numbers converging to 0 and satisfying  $\lim_{l \rightarrow \infty} [z(t^l)]_i = 0$  exists. It follows that  $\{z_i^* / [z(t^l)]_i\}$  is unbounded. But this is a contradiction, since  $z_i^* / [z(t^l)]_i \leq (z^*)^T z(t^l)^{-1} \leq M$  according to (2.32). Hence  $[z(0)]_i > 0$ .  $\square$

**COROLLARY 2.2.** *If  $\text{MLCP}(A, B, C, b)$  is monotone and  $(x^*, \tau^*, s^*, \kappa^*, y^*)$  is an asymptotical solution of the corresponding HMCP with  $\tau^* > 0$  ( $\kappa^* > 0$ ), then any accumulation point  $(x(0), \tau(0), s(0), \kappa(0), y(0))$  of the path  $\mathcal{P}$  satisfies  $\tau(0) > 0$  ( $\kappa(0) > 0$ , respectively).*

As we have mentioned earlier, the HMCP has always a solution for which the pair  $(\tau^*, \kappa^*)$  possesses strict complementarity. The above theorem shows that if the HMCP has a solution having  $\tau^* > 0$  ( $\kappa^* > 0$ , respectively), then a path-following interior-point algorithm does not converge to a solution having  $\tau^* = 0$  ( $\kappa^* = 0$ , respectively). Hence the solutions found by a path-following interior-point algorithm are valid certificates (in the sense of Theorem 2.2) of solvability or infeasibility of the original MLCP .

## 2.7 Adaptation of Mehrotra's interior-point algorithm for HMCP

We have shown in Section 2.6 that the numerical solution to the homogeneous augmented model can be obtained using interior-point path-following algorithms. A path-following method is an iterative numerical process that follows a path of points, in our case  $\mathcal{P}$  defined by (2.27), in the direction of decreasing  $t$  towards the solution set of the complementarity problem. The iterates generated by the method do not necessarily stay on the curve  $\mathcal{P}$ , rather they are located in a controlled neighborhood of  $\mathcal{P}$  that is a subset of the positive orthant.

The use of a neighborhood is a key ingredient of path-following methods and keeps the iterates from moving too close to the boundary of the positive orthant. Little progress can be made in the proximity of the boundary of positive orthant because of the numerical distortions associated to small, even zero, numerical values in the complementarity variables. While the wider neighborhoods offer the best performance in practice, the smaller neighborhoods give the best theoretical complexity bounds.

In pursuing the central path, path-following methods step along two directions. The first is the affine-scaling direction which aims for a great amount of progress towards the solution and generates a point situated on the boundary of neighborhood. The second is the centering direction which is biased toward the central path such that the progress made by affine-scaling direction is at least preserved, if not improved. Moreover, the centering direction sets the scene for substantial progress along the next affine-scaling step in the idea that the better the centrality is, the more room for the affine-scaling step is. Predictor-corrector algorithms are path-following algorithms that take a separate steps along the affine-scaling direction (predictor) and centering

direction (corrector). Another type of path-following algorithms is represented by the short(long)-step path-following methods which combine the affine-scaling and centering directions in a single step and trade off between the two goals of reaching optimality and improving centrality.

In the last decade, predictor-corrector methods have emerged as a practical approach in obtaining numerical solutions to complementarity problems. Among them, the most successful and hence famous is the Mehrotra's predictor-corrector algorithm. Although Mehrotra [58] presented his algorithm in the context of linear programming, it was successfully applied also to convex quadratic programming [40] and standard monotone linear complementarity problems [85]. It also has been widely used in the implementation of several IPM based optimization packages: OB1 [57], HOPDM [42], PcX [23], LIPSOL [86], OOQP [40], etc. In this section we adapt this algorithm to an even broader class of complementarity problems, that is monotone nonlinear mixed complementarity problems of form (1.3). The algorithm makes use of no explicit neighborhood of the central path. In fact it uses the entire positive orthant to move along the affine-scaling direction and backs off by a fixed factor when the boundary of the orthant is reached. The use of no neighborhood causes the complexity and convergence analysis of the algorithm to be intractable. To date, no complexity and convergence results are known for the original form of Mehrotra's algorithm. However, polynomial complexity was proved for variants of Mehrotra's algorithm in [87, 85, 70, 71]. The Mehrotra-type algorithms from [85] and [70] also possess a fast asymptotical convergence,  $Q$ -subquadratic and superlinear, respectively.

In what follows we present how Mehrotra's algorithm is used to find the numerical solution to the augmented homogeneous model (2.5) related to a monotone  $\text{MLCP}(A, B, C, b)$  that we introduced in Section 2.3. We recall that our homogeneous

model consists in the following nonlinear monotone complementarity problem

$$\begin{aligned}
xs &= 0 \\
\tau\kappa &= 0 \\
F(x, \tau, s, \kappa, y) &= 0 \\
x, \tau, s, \kappa &\geq 0,
\end{aligned} \tag{2.33}$$

where the mapping  $F$  describing the feasible set of the complementarity problem is given by (2.8).

We now introduce some simplifying notations. The complementarity conditions and the feasibility equations can be represented as the continuous mapping  $H : \mathbb{R}_+^n \times \mathbb{R}_{++} \times \mathbb{R}_+^{n+1} \times \mathbb{R}^m \rightarrow \mathbb{R}^{2n+m+2}$ ,

$$H(x, s, y) = \begin{bmatrix} xs \\ \tau\kappa \\ F(x, s, y) \end{bmatrix}. \tag{2.34}$$

We denote the Jacobian of  $H$  by  $H'$ , which has the form

$$H'(x, \tau, s, \kappa, y) = \begin{bmatrix} S & 0 & X & 0 & 0 \\ 0 & \kappa & 0 & \tau & 0 \\ A & -b & B & 0 & C \\ d_x^T & d_\tau^T & d_s^T & 1 & d_y^T \end{bmatrix}, \tag{2.35}$$

where  $X = \text{diag}(x)$ ,  $S = \text{diag}(s)$ ,  $d_x := \frac{d}{dx} \left( \frac{\bar{x}^T \bar{s}}{\tau} \right) (x, \tau, s, \kappa, y)$ ,  $d_s := \frac{d}{ds} \left( \frac{\bar{x}^T \bar{s}}{\tau} \right) (x, \tau, s, \kappa, y)$ ,  $d_y := \frac{d}{dy} \left( \frac{\bar{x}^T \bar{s}}{\tau} \right) (x, \tau, s, \kappa, y)$ , and  $d_\tau := \frac{d}{d\tau} \left( \frac{\bar{x}^T \bar{s}}{\tau} \right) (x, \tau, s, \kappa, y)$ . The exact expressions for  $d_x$ ,  $d_s$ ,  $d_y$ , and  $d_\tau$  are not of interest at this moment and will be given later.

We first outline the adaptation of Mehrotra's algorithm for the complementarity problem (2.33).

### Mehrotra's algorithm for HMCP

Set  $(x_0, \tau_0, s_0, \kappa_0, y_0) = (e, 1, e, 1, 0)$ ;

Set  $k \leftarrow 0$ ;

**repeat**

Set  $(x, \tau, s, \kappa, y) \leftarrow (x_k, \tau_k, s_k, \kappa_k, y_k)$ ;

Compute  $\mu = \lceil x, \tau \rceil^T \lceil s, \kappa \rceil / (n + 1)$ ;

*(predictor step)*

Compute  $(u_p, \alpha_p, v_p, \beta_p, w_p)$  from

$$\begin{bmatrix} S & 0 & X & 0 & 0 \\ 0 & \kappa & 0 & \tau & 0 \\ A & -b & B & 0 & C \\ d_x^T & d_\tau^T & d_s^T & 1 & d_y^T \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ v \\ \beta \\ w \end{bmatrix} = \begin{bmatrix} -xs \\ -\tau\kappa \\ \tau b - Ax - Bs - Cy \\ -\frac{1}{\tau} \bar{x}^T \bar{s} - \kappa \end{bmatrix}; \quad (2.36)$$

Compute  $\theta_p = \arg \max\{\theta \in (0, 1] : (x, \tau, s, \kappa) + \theta(u_p, \alpha_p, v_p, \beta_p) \geq 0\}$ ;

Set  $\mu_p = (\lceil x, \tau \rceil + \theta_p \lceil u_p, \alpha_p \rceil)^T (\lceil s, \kappa \rceil + \theta_p \lceil v_p, \beta_p \rceil) / (n + 1)$ ;

Set centering parameter  $\sigma = (\mu_p / \mu)^3$ ;

*(corrector step)*

Compute  $(u, \alpha, v, \beta, w)$  from

$$\begin{bmatrix} S & 0 & X & 0 & 0 \\ 0 & \kappa & 0 & \tau & 0 \\ A & -b & B & 0 & C \\ d_x^T & d_\tau^T & d_s^T & 1 & d_y^T \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ v \\ \beta \\ w \end{bmatrix} = \begin{bmatrix} \sigma\mu e - xs - u_p v_p \\ \sigma\mu - \tau\kappa - \alpha_p \beta_p \\ \tau b - Ax - Bs - Cy \\ -\frac{1}{\tau} \bar{x}^T \bar{s} - \kappa \end{bmatrix}; \quad (2.37)$$

Compute  $\theta_{max} = \max\{\theta : \lceil x, \tau, s, \kappa \rceil + \theta \lceil u, \alpha, v, \beta \rceil \geq 0\}$

Compute steplength  $\theta_c \in (0, \theta_{max})$  according to Section 2.7.1;

Set  $(x_{k+1}, \tau_{k+1}, s_{k+1}, \kappa_{k+1}, y_{k+1}) \leftarrow (x_k, \tau_k, s_k, \kappa_k, y_k) + \theta_c(u, \alpha, v, \beta, w);$   
 Set  $k \leftarrow k + 1.$

**continue**

### Discussion of the algorithm

Suppose that the current iteration is  $(x_k, \tau_k, s_k, \kappa_k, y_k).$

In the predictor phase Mehrotra's algorithm aims to reduce both the complementarity and infeasibility while keeping the iterate in the feasible set. To achieve this, the algorithm performs one damped Newton iteration for the nonlinear system of equations represented by the complementarity conditions and feasibility equations, *i.e.*, for

$$H(x, \tau, s, \kappa, y) = 0.$$

By applying Newton's method to the above system of equations with the current iterate  $(x_k, \tau_k, s_k, \kappa_k, y_k)$  as starting point, we obtain that the predictor search directions  $(u_p, \alpha_p, v_p, \beta_p, w_p)$  must satisfy the linear system

$$H'(x_k, \tau_k, s_k, \kappa_k, y_k)(u_p, \alpha_p, v_p, \beta_p, w_p) = -H(x_k, \tau_k, s_k, \kappa_k, y_k).$$

By taking into account the expression (2.35) of the Jacobian, it can be easily seen that the linear system solved for the predictor search direction is exactly (2.36).

Since a full step along these directions usually ends outside the positive orthant, the algorithm finds the step length  $\theta_p$  to the boundary of the positive orthant, but, instead of taking the update, it just measures the efficiency of the direction by computing  $\mu_p$ . A value  $\mu_p \ll \mu$  indicates that the direction permits significant progress in reducing the complementarity measure  $\mu$ . If  $\mu_p$  is comparable to  $\mu$ , then little progress can be made because of the current iterate's lack of centrality. The centering parameter  $\sigma$  is computed by an ingenious heuristic found by Mehrotra and indicates how

much emphasis should be put on centrality during the corrector's centering phase.

Based on the amount of reduction obtained in the predictor phase, one of the corrector's objectives is to bring the iterate close to central path without compromising the progress made in the predictor. Being as close to the central path as possible usually allows longer steps and hence large reduction of the complementarity in the next predictor step. Another objective of the corrector is to reduce the error made by predictor in the attempt of solving the nonlinear system of equations (2.7) with only one Newton step. The above objectives cause the corrector search direction  $(u, \alpha, v, \beta, w)$  given by (2.37) to be a combination of an error-corrector direction and a centering direction which we describe in the following paragraphs.

The error-corrector direction tries to compensate the complementarity error  $\begin{bmatrix} x + u_p, \tau + \alpha + p \end{bmatrix} \begin{bmatrix} s + v_p, \kappa + \beta_p \end{bmatrix}$  that is made by a full step along the predictor direction because of the nonlinearity of (2.7). No reduction on the infeasibility is wanted. Observe that we have

$$\begin{aligned} \begin{bmatrix} x + u_p \\ \tau + \alpha_p \end{bmatrix} \begin{bmatrix} s + v_p \\ \kappa + \beta_p \end{bmatrix} &= \begin{bmatrix} x \\ \tau \end{bmatrix} \begin{bmatrix} s \\ \kappa \end{bmatrix} + \begin{bmatrix} x \\ \tau \end{bmatrix} \begin{bmatrix} v_p \\ \beta_p \end{bmatrix} + \begin{bmatrix} u_p \\ \alpha_p \end{bmatrix} \begin{bmatrix} s \\ \kappa \end{bmatrix} \\ &+ \begin{bmatrix} u_p \\ \alpha_p \end{bmatrix} \begin{bmatrix} v_p \\ \beta_p \end{bmatrix} = \begin{bmatrix} u_p \\ \alpha_p \end{bmatrix} \begin{bmatrix} v_p \\ \beta_p \end{bmatrix}, \end{aligned}$$

where the last equality is obtained from (2.36). In order to compensate this error, the error-corrector direction  $(u_{co}, \alpha_{co}, v_{co}, \beta_{co}, w_{co})$  is computed as the solution of the

linear system

$$\begin{bmatrix} S & 0 & X & 0 & 0 \\ 0 & \kappa & 0 & \tau & 0 \\ A & -b & B & 0 & C \\ d_x^T & d_\tau^T & d_s^T & 1 & d_y^T \end{bmatrix} \begin{bmatrix} u_{co} \\ \alpha_{co} \\ v_{co} \\ \beta_{co} \\ w_{co} \end{bmatrix} = \begin{bmatrix} -u_p v_p \\ -\alpha_p \beta_p \\ 0 \\ 0 \end{bmatrix}. \quad (2.38)$$

The centering direction is biased toward the central path and tries to move closer to the central path, so that the algorithm is in a better position to achieve substantial decrease in the complementarity measure during the next iteration. As the error-corrector direction, the centering direction does not try to reduce infeasibility either. The centering direction targets the point  $\sigma\mu e$  located on the central path, where  $\sigma$  is the centering parameter computed in the predictor case. In order to achieve this, the centering direction  $(u_c, \alpha_c, v_c, \beta_c, w_c)$  is computed as the solution of by the linear system

$$\begin{bmatrix} S & 0 & X & 0 & 0 \\ 0 & \kappa & 0 & \tau & 0 \\ A & -b & B & 0 & C \\ d_x^T & d_\tau^T & d_s^T & 1 & d_y^T \end{bmatrix} \begin{bmatrix} u_c \\ \alpha_c \\ v_c \\ \beta_c \\ w_c \end{bmatrix} = \begin{bmatrix} \sigma\mu e \\ \sigma\mu \\ 0 \\ 0 \end{bmatrix}. \quad (2.39)$$

A final observation regarding the centering phase is that, instead of targeting the point on the central path having the complementarity  $\mu_p$  obtained by the predictor, the centering direction aims the point  $\sigma\mu e$  whose complementarity is  $\sigma\mu = \left(\frac{\mu_p}{\mu}\right)^3 \mu = \mu_p \left(\frac{\mu_p}{\mu}\right)^2 \leq \mu_p$  ( $\mu_p \leq \mu$ , since it can be proved that the complementarity measure  $\mu$  is reduced along the predictor direction). Hence, the centering direction aims not only to center the iterate, but also to reduce the complementarity measure already gained by the predictor.

The corrector's search direction (2.37) is obtained by summing the predictor direction given by (2.36), the error-corrector direction given by (2.38), and the centering direction given by (2.39). The algorithm updates the iteration by moving along this direction by a step length given by a heuristic that will be later discussed. An important observation is that Mehrotra's algorithm differs from what today are called "standard predictor-corrector" (or corrector-predictor) methods in two aspects. The first is given by the absence of an update based solely on predictor directions in Mehrotra's algorithm. In contrast, a standard predictor-corrector moves along predictor directions till it reaches a point on the boundary of the neighborhood and performs the corrector step using this point. The second difference is that the error-corrector direction is not present in standard predictor-corrector algorithms which attempt only an improvement of the centrality in the corrector phase.

The characteristic of Mehrotra's method of not moving from the current iterate in the predictor phase causes, as we have already pointed out, the linear systems (2.36) and (2.37) to share the same system matrix. Thus only one expensive matrix factorization and two less expensive backsolves are needed per iteration. On the other hand, the standard predictor-corrector and corrector-predictor methods require two expensive matrix factorizations and two backsolves per iteration. Even though the number of iterations may be slightly smaller when making use of standard predictor-corrector methods employing large neighborhoods, the numerical experience of the last fifteen years has shown that Mehrotra's algorithm considerably outruns any of such methods in terms of execution time.

### 2.7.1 The procedure for computing the steplength

The steplength heuristic from the corrector step introduced by Mehrotra in [58] for linear programming was slightly modified by Wright in [40] for quadratic programming to ensure that the same step is taken in both primal and dual variables. Furthermore, the experiments (not included in this thesis) that I have done with Mehrotra's algorithm on MLCPs have shown that the steplength heuristic used for quadratic programming works well for linear complementarity problems. However, the heuristics may cause the algorithm not to converge when dealing with nonlinear complementarity problems. We have observed this on the very first HMCPs we have tried to solve using Mehrotra's algorithm and Mehrotra's heuristic for steplength. Mehrotra's heuristic and its variants assume that the infeasibility is reduced along the entire corrector direction, and only criteria related to the complementarity measure are considered in computing the steplength. The assumption is satisfied for linear complementarity problem since the feasibility is linear and Newton-type direction is a descent direction for any step in  $(0, 1]$ . But, in the case of nonlinear complementarity problems the assumption is not satisfied anymore, Newton-type direction is only a local descent direction (and only under the monotonicity property).

To avoid this shortcoming we compute the steplength along the corrector direction similarly to [3] by enforcing a sufficient decrease condition in a merit function. The merit function measures the progress towards the solution in terms of both complementarity and infeasibility, and is defined by

$$\phi(x, \tau, s, \kappa, y) = \zeta(x^T s + \tau \kappa) + \|F(x, \tau, s, \kappa, y)\|, \quad (2.40)$$

where  $\zeta$  is a positive parameter used to balance between complementarity ( $x^T s$ ) and infeasibility ( $\|F(x, s, y)\|$ ). Clearly, if the point  $(x, \tau, s, \kappa, y)$  satisfies  $\phi(x, \tau, s, \kappa, y) =$

0, then  $(x, \tau, s, \kappa, y)$  is a complementarity solution.

By enforcing the Armijo condition for the merit function along the corrector direction  $\lceil u, \alpha, v, \beta, w \rceil$ , we require the corrector step size  $\theta_c$  to satisfy

$$\phi(x^+, \tau^+, s^+, \kappa^+, y^+) \leq c_1 \theta_c \nabla \phi(x, \tau, s, \kappa, y)^T \lceil u, \alpha, v, \beta, w \rceil, \quad (2.41)$$

where  $\lceil x^+, \tau^+, s^+, \kappa^+, y^+ \rceil = \lceil x, \tau, s, \kappa, y \rceil + \theta_c \lceil u, \alpha, v, \beta, w \rceil$  is the candidate for update, and  $c_1$  is a constant of the algorithm.

We also require an additional condition on the length  $\theta_c$  of the corrector step. This can also be found in [3] and has the role of preventing the iterates from converging prematurely towards the boundary of the nonnegative orthant (which can cause numerical difficulties):

$$x_i^+ s_i^+ \geq c_2 \mu^+, \text{ for } i = 1, 2, \dots, n, \text{ and } \tau^+ \kappa^+ \geq c_2 \mu^+, \quad (2.42)$$

where  $\mu^+ = ((x^+)^T s^+ + \tau^+ \kappa^+) / (n + 1)$ , and  $c_2$  is a constant of the algorithm.

#### **Procedure for computing the step size $\theta_c$**

Set  $\theta_c \leftarrow \theta_{max}$ ;

Set  $k \leftarrow 0$ ;

#### **Repeat**

Set  $\lceil x^+, \tau^+, s^+, \kappa^+, y^+ \rceil = \lceil x, \tau, s, \kappa, y \rceil + \theta_c \lceil u, \alpha, v, \beta, w \rceil$ ;

If (2.41) and (2.42) are satisfied then

Accept and return  $\theta_p$ ;

Else

$$\theta_c = c_3^{2^k} \theta_c;$$

If  $\theta_c \leq c_4$  then

Return error;

Set  $k \leftarrow k + 1$ ;

**continue**

A key question is if the above procedure is well-defined in the sense that the corrector direction  $[u, \alpha, v, \beta, w]$  is a descent direction for the merit function  $\phi$ . By taking separately the two components of the merit function, it can be easily proved that the corrector direction is always a descent direction for the infeasibility component ( $\|F(x, \tau, s, \kappa, y)\|$ ) but not for the complementarity component  $\zeta(x^T s + \tau \kappa)$ . However, when considered together, sufficient decrease of the merit is obtained along corrector direction as indicated by the numerical experiments.

Based on extensive numerical experiments we have settled to the values for the algorithm's constants  $c_1 = 10^{-4}$ ,  $c_2 = 10^{-6}$ ,  $c_3 = 0.85$ ,  $c_4 = 10^{-4}$ , and  $\zeta = 1/\sqrt{n+1}$ .

## 2.7.2 Linear algebra

We start by computing the derivatives  $d_x$ ,  $d_s$ ,  $d_y$ , and  $d_\tau$  that appear in the expression of the linear systems (2.36) and (2.37). Let us denote  $[x, s, y]$  by  $z$ , and  $[\bar{x}, \bar{s}, \bar{y}]$  by  $\bar{z}$ . Also let  $H$  be the  $(2n + m) \times (2n + m)$  matrix defined by

$$H = \begin{bmatrix} 0 & I & 0 \\ I & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Through the rest of this section we denote by  $P$  and  $R$  the orthogonal projection onto  $\text{Ker}[ABC]$  and  $\text{Ran}[ABC]^T$ , respectively.

First of all, based on (2.6) and (2.7), let us write  $\bar{x}^T \bar{s} = \frac{1}{2} \bar{z}^T H \bar{z} = \frac{1}{2} (Pz +$

$\tau\bar{b})^T H(Pz + \tau\bar{b}) = \frac{1}{2}(z^T PHPz + 2\tau z^T PH\bar{b} + \tau^2 \bar{b}^T H\bar{b})$ , so that

$$\begin{bmatrix} d_x \\ d_s \\ d_y \end{bmatrix} = \frac{d}{dz} \left( \frac{\bar{x}^T \bar{s}}{\tau} \right) = \frac{1}{\tau} (PHPz + \tau PH\bar{b}) = \frac{1}{\tau} PH\bar{z}. \quad (2.43)$$

In obtaining  $d_\tau$ , first observe that  $\frac{d}{d\tau}(\bar{x}^T \bar{s}) = z^T PH\bar{b} + \tau\bar{b}^T H\bar{b} = \bar{z}^T H\bar{b}$  and then, by the product rule that

$$d_\tau = \frac{d}{d\tau} \left( \frac{\bar{x}^T \bar{s}}{\tau} \right) = \frac{1}{\tau} \bar{z}^T H\bar{b} - \frac{1}{\tau^2} \bar{x}^T \bar{s}. \quad (2.44)$$

At each iteration of the algorithm the directions for predictor and corrector are obtained by solving the linear systems (2.36) and (2.37), respectively. Since they share the same system matrix, the numerical factorization is done only once. Moreover, since the systems are sparse, unsymmetric and indefinite we chose to use MA48 [30] solver from HSL library. MA48 solves sparse unsymmetric systems of linear equations using Gaussian elimination. It is written in Fortran and has capabilities such as: finding the pivot order, factorizing using a given pivot order, forward and backsolves and error estimates.

However, when a particular class of problems is solved, the linear systems may have special properties as symmetry, positive-definiteness, etc. In this case the performance of the interior-point solver can be dramatically increased. For example, in solving linear or convex quadratic problems as LCPs a symmetric positive definite linear solver is recommended to be used [66].

At each iteration of the algorithm, the vectors  $[\bar{x}, \bar{s}, \bar{y}]$  (see (2.6)) and  $[d_x, d_s, d_y]$  (see (2.43)) are computed by finding the projection onto  $\text{Ker}[ABC]$  of the vector  $[x, s, y]$  and  $H\bar{z} = [\bar{s}, \bar{x}, 0]$ , respectively. Since  $\text{Ker}[ABC]$  is perpendicular to  $\text{Ran}[ABC]^T$ , we can write  $[x, s, y] = P_{\text{Ker}[ABC]}[x, s, y] + P_{\text{Ran}[ABC]}[x, s, y]$  and

hence

$$P \begin{bmatrix} x \\ s \\ y \end{bmatrix} = \begin{bmatrix} x \\ s \\ y \end{bmatrix} - \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} M^{-1} \begin{bmatrix} A & B & C \end{bmatrix} \begin{bmatrix} x \\ s \\ y \end{bmatrix}, \quad (2.45)$$

where by  $M$  we denoted the matrix  $AA^T + BB^T + CC^T$ .

The vector  $[\bar{x}, \bar{s}, \bar{y}]$  is computed using the equation (2.45) for  $P[x, s, y]$  and the expression (2.7) for  $\bar{b}$ :

$$\begin{aligned} \begin{bmatrix} \bar{x} \\ \bar{s} \\ \bar{y} \end{bmatrix} &= P \begin{bmatrix} x \\ s \\ y \end{bmatrix} + \tau \bar{b} \\ &= \begin{bmatrix} x \\ s \\ y \end{bmatrix} + \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} M^{-1}(b - Ax - Bs - Cy). \end{aligned} \quad (2.46)$$

Once  $[\bar{x}, \bar{s}, \bar{y}]$  is known, the derivative information whose expression is given by (2.43) is obtained by projecting the vector  $H\bar{z} = [\bar{s}, \bar{x}, 0]$  onto  $\text{Ker}[ABC]$  using (2.45):

$$\begin{bmatrix} d_x \\ d_s \\ d_y \end{bmatrix} = \frac{1}{\tau} \left( \begin{bmatrix} \bar{s} \\ \bar{x} \\ 0 \end{bmatrix} - \begin{bmatrix} A^T \\ B^T \\ C^T \end{bmatrix} M^{-1}(A\bar{s} + B\bar{x}) \right). \quad (2.47)$$

As it can be seen from (2.46) and (2.47), the algorithm requires two additional linear systems to be solved at each iteration. The matrix  $M = AA^T + BB^T + CC^T$  of the two linear systems is sparse symmetric positive definite hence a specialized linear solver should be used. Performance, reliability as well as availability were the main aspects we considered while choosing a sparse direct solver for symmetric positive definite linear systems of equations. We chose CHOLMOD 1.4 [26, 22, 25, 24]

based on the evaluations from [44] and its readily availability. CHOLMOD splits the factorization of a matrix in two main parts. The first is the so called symbolic analysis and consists of computations that typically depend only on the nonzero pattern, not on the numerical values. It finds a permutation of the matrix so that the amount of fill-in in the factors is minimized (or at least significantly decreased). It also determines the exact sparsity pattern of the factor. The second part of the CHOLMOD factorization process is the numerical factorization based on a Cholesky-based algorithm. For sparse matrices, the symbolic analysis is usually much more expensive than the numerical factorization.

The system matrix  $M$  appearing in (2.46) and (2.47) does not change with the algorithm's iterations, so that both symbolic analysis and numerical factorization are done only once, when the algorithm starts. Hence computing  $[d_x, d_s, d_y]$  and  $[\bar{x}, \bar{s}, \bar{y}]$  at each iteration resumes to two relatively inexpensive backsolves with the factors of  $M$ .

## 2.8 Generating random monotone linear complementarity problems

In this section we present a method that generates monotone mixed linear complementarity problems with sparse random data. When constructing such problems, not only their monotonicity and randomness, but also an a priori information on their solution set, *e.g.* no solution, multiple solutions, strict complementarity solution, etc, is of interest. The generated problems will be used in Section 2.9 to observe and test the numerical behavior of the augmented homogeneous model discussed in this chapter.

We first construct a monotone linear complementarity problem in standard form. Then, the standard form will be transformed in the horizontal form and finally in the mixed horizontal form.

### 2.8.1 Generating monotone SLCPs

The goal of this section is to generate random monotone linear complementarity problems in standard form defined by (1.5), *i.e.*,

$$\begin{aligned} xs &= 0, \\ s &= Mx + b, \\ x, s &\geq 0. \end{aligned} \tag{2.48}$$

As we mentioned before, monotonicity of the  $\text{SLCP}(M, b)$  means that the matrix  $M$  is positive semidefinite, not necessarily symmetric. In what follows we present our approach for generating a sparse positive definite and semidefinite matrix with real entries.

For clarity purposes, we now define the main concepts and state some basic results regarding positive (semi)definite matrices.

**DEFINITION 2.7.** (i) A matrix  $M \in \mathbb{R}^{n \times n}$  is called *positive definite* if  $x^T M x > 0$  for any nonzero vector  $x \in \mathbb{R}^n$ .

(ii) A matrix  $M \in \mathbb{R}^{n \times n}$  is called *positive semidefinite* if  $x^T M x \geq 0$  for any vector  $x \in \mathbb{R}^n$ .

**PROPOSITION 2.1.** A matrix  $M \in \mathbb{R}^{n \times n}$  is *positive definite* (*positive semidefinite*) if and only if the symmetric part  $\frac{1}{2}(M + M^T)$  is *positive definite* (*positive semidefinite*).

The following result is known as the "Gersgorin Circle Theorem".

**THEOREM 2.7.** Let  $A \in \mathbb{C}^{n \times n}$  and  $R_i = \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}|$ . Then each eigenvalue of  $A$  is in at least one of the disks  $\{z : |z - a_{ii}| \leq R_i\}$ ,  $i = 1, \dots, n$ .

**THEOREM 2.8.** (i) All eigenvalues of a symmetric matrix  $A \in \mathbb{R}^{n \times n}$  are real.

(ii) A symmetric matrix  $M \in \mathbb{R}^{n \times n}$  is positive definite (positive semidefinite) if and only if its eigenvalues are positive (nonzero).

We generate a positive definite matrix  $M$  having in mind that the symmetrized matrix  $\frac{1}{2}(M + M^T)$  should be also positive definite. We start with a sparse nonsingular matrix and we modify it based on the Gersgorin Circle Theorem in a way that assures the positiveness of the symmetrized matrix by Theorem 2.8 as shown in the following proposition.

**PROPOSITION 2.2.** Let  $M \in \mathbb{R}^{n \times n}$  be sparse and nonsingular with (small) positive diagonal entries. Define  $R_i^s = \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}|$  and  $R_i^c = \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ji}|$ ,  $i = 1, \dots, n$ . The matrix  $\widetilde{M} \in \mathbb{R}^{n \times n}$  defined by  $\widetilde{m}_{ii} = m_{ii} + \frac{1}{2}(R_i^s + R_i^c)$ ,  $\widetilde{m}_{ij} = m_{ij}$ ,  $i, j = 1, \dots, n$ ,  $i \neq j$  is positive definite.

*Proof.* The Gersgorin's circle theorem will be used to prove that the symmetric matrix  $S = \frac{1}{2}(\widetilde{M} + \widetilde{M}^T)$  has real positive eigenvalues. For this we take a look at  $\widetilde{R}_i := \sum_{\substack{j=1 \\ j \neq i}}^n |s_{ij}| = \sum_{\substack{j=1 \\ j \neq i}}^n \frac{1}{2} |\widetilde{m}_{ij} + \widetilde{m}_{ji}| = \sum_{\substack{j=1 \\ j \neq i}}^n \frac{1}{2} |m_{ij} + m_{ji}|$ , hence we can write

$$\widetilde{R}_i \leq \frac{1}{2}(R_i^s + R_i^c). \quad (2.49)$$

The Gersgorin's disks corresponding to the matrix  $S$  have the form  $\{z : |z - s_{ii}| \leq \widetilde{R}_i\} = \{z : |z - \widetilde{m}_{ii}| \leq \widetilde{R}_i\} = \{z : |z - m_{ii} - \frac{1}{2}(R_i^s + R_i^c)| \leq \widetilde{R}_i\}$ . Using inequality (2.49) and the fact that  $m_{ii}$  are positive we obtain that the real numbers contained in

the disks must be positive. Furthermore, we obtain that  $S$  has positive eigenvalues, which implies that  $S$  is positive definite since it is symmetric.

Then  $\widetilde{M}$  is positive definite by Proposition 2.1.  $\square$

There is an alternative technique to Proposition 2.2 for generating positive definite matrices. The alternative would consist of generating a symmetric positive definite matrix  $M$  (for example as the product of a nonsingular matrix with its transpose) and perturbing it with a sparse skew symmetric matrix  $H$ , that is  $H = -H^T$ . The resulting matrix  $\widetilde{M} = M + H$  is positive definite since its symmetric part  $\frac{1}{2}(\widetilde{M} + \widetilde{M}^T) = \frac{1}{2}(M + H + M^T - H) = M$  is positive definite. But, observe that even though  $\widetilde{M}$  is not symmetric, it has a symmetric sparsity pattern. This is the reason for which, when working with sparse matrices, we prefer the method described in Proposition (2.2) since it generates non-symmetric positive definite matrices with non-symmetric sparsity pattern.

In what follows we will show how one can generate positive semidefinite matrices. While positive definite matrices have always full rank, positive semidefinite matrices are rank deficient. Therefore it makes sense to generate positive semidefinite matrices of a given rank, as shown in the following proposition.

**PROPOSITION 2.3.** *Let  $n$  and  $r$  be natural numbers with  $r < n$ ,  $M \in \mathbb{R}^{r \times r}$  a positive definite matrix and  $R \in \mathbb{R}^{r \times r-n}$ . The matrix*

$$\widetilde{M} = \begin{bmatrix} M & MR \\ R^T M & R^T MR \end{bmatrix} \in \mathbb{R}^{n \times n} \quad (2.50)$$

*is positive semidefinite of rank  $r$ .*

*Proof.* The rank of  $\widetilde{M}$  is at most  $r$  since each of its last  $n - r$  columns and rows is a linear combinations of the first  $r$  columns and rows, respectively. But since  $M$  is positive definite, it has full rank  $r$ , so that  $\widetilde{M}$  has rank  $r$ .

We will show that  $\widetilde{M}$  is positive semidefinite using Definition 2.7. Let us consider  $z = [x, y] \in \mathbb{R}^n$ ,  $x \in \mathbb{R}^r$ ,  $y \in \mathbb{R}^{n-r}$  and compute

$$\begin{aligned} z^T \widetilde{M} z &= \begin{bmatrix} x^T & y^T \end{bmatrix} \begin{bmatrix} Mx + MRy \\ R^T Mx + R^T MRy \end{bmatrix} \\ &= x^T Mx + x^T MRy + y^T R^T Mx + y^T R^T MRy \\ &= x^T M(x + Ry) + y^T R^T M(x + Ry) = (x + Ry)^T M(x + Ry). \end{aligned}$$

We can conclude based on the positive definiteness of  $M$  that  $z^T \widetilde{M} z \geq 0$  for any  $z \in \mathbb{R}^n$ , which shows that  $\widetilde{M}$  is positive semidefinite.  $\square$

The generation of a sparse positive semidefinite matrix using the previous proposition requires the generation of a sparse positive definite matrix  $M$  using Proposition 2.50 and a sparse random matrix  $R$ . It must be pointed out that  $R$  should be sparser than  $M$  so that the block  $R^T MR$  does not become too dense.

A SLCP( $M, b$ ) with a rank-deficient positive semidefinite matrix  $M$  can have multiple solutions or no solution at all, depending on the right-hand side  $b$ . On the other hand, if  $M$  is positive definite, then SLCP( $M, b$ ) has a unique solution for any vector  $b$ .

At this point we have the necessary instrumentary for generating solvable monotone SLCPs. First, the matrix  $M$  should be generated (positive definite for unique solution, positive semidefinite for multiple solutions). After that we generate random vectors  $x, s \in \mathbb{R}^n$ , so that they satisfy the complementarity conditions  $xs = 0$  and the non-negativity constraints  $x, s \geq 0$ . The right-hand side of the SLCP is obtained from  $b = s - Mx$ . The presence of a non-strict complementarity solution can be obtained by generating  $x$  and  $s$  accordingly.

A SLCP of dimension  $n$  not having any solution can be generated as follows. First we generate a random positive definite matrix  $M \in \mathbb{R}^{(n-1) \times (n-1)}$  and a random vector

$b \in \mathbb{R}^{n-1}$ . Let the SLCP have the form

$$\begin{aligned} \begin{bmatrix} x \\ \alpha \end{bmatrix} \begin{bmatrix} s \\ \beta \end{bmatrix} &= 0, \\ \begin{bmatrix} s \\ \alpha \end{bmatrix} &= \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \beta \end{bmatrix} + \begin{bmatrix} b \\ -1 \end{bmatrix}, \\ x, \alpha, s, \beta &\geq 0. \end{aligned} \tag{2.51}$$

It is clear that the above problem is monotone and that the affine equality constraints from (2.51) imply  $\alpha = -1$ , which make the SLCP infeasible.

## 2.8.2 Generating monotone HLCPs

We now show how to generate monotone horizontal random linear complementarity problems of form (1.6), *i.e.*,

$$\begin{aligned} xs &= 0 \\ Qx + Rs &= b \\ x, s &\geq 0. \end{aligned}$$

The technique relies on the generation of random monotone SLCPs described in the previous section.

Once a monotone SLCP( $M, b$ ) is generated, a straightforward technique for obtaining a monotone HLCP would be to multiply the affine system of equations  $s = Mx + b$  with a nonsingular matrix  $N$  to obtain the horizontal form HLCP( $NM, -N, Nb$ ). The obtained horizontal form is monotone since  $NMu - Nv = 0$  implies  $Mu - v = 0$  by the nonsingularity of  $N$ , and finally  $u^T v = u^T Mu \geq 0$ , by the monotonicity of SLCP( $M, b$ ). However, the HLCP generated by using this method always has a full rank matrix  $N$  corresponding to the complementarity variable  $x$ . We find this situation particular since, in general, for a HLCP( $Q, R, b$ ) both  $Q$  and  $R$  can be rank

deficient in the monotone case, as indicated by Gowda in [46]. The work from [46] is very useful in determining the connection between HLCP and SLCP under the monotonicity assumption. The paper shows that a monotone HLCP can be transformed to a monotone SLCP using linear algebra operations such as finding the maximal linearly independent column set of a given matrix and inverting a nonsingular matrix. The reduction technique itself is not of interest in this discussion, but the proof of the fact that the transformation is possible gives useful insight on how to go from a SLCP to a HLCP( $Q, R, b$ ) with rank deficient matrices  $Q$  and  $R$ . Our technique for obtaining a HLCP from a previously generated SLCP is based on the following proposition.

**PROPOSITION 2.4.** *Let  $n$  and  $r$  be natural numbers with  $r < n$ ,  $M \in \mathbb{R}^{r \times r}$  be a positive definite matrix,  $R_1 \in \mathbb{R}^{r \times r_1}$ ,  $R_2 \in \mathbb{R}^{r \times r_2}$  and  $X \in \mathbb{R}^{r_1 \times r_3}$ , where  $r_1 + r_2 + r_3 = n - r$ . Also consider the matrices*

$$A = \begin{bmatrix} I_r & -MR_1 & 0 & -MR_1X \\ 0 & -R_1^T MR_1 & 0 & -R_1^T MR_1X \\ 0 & -R_2^T MR_1 & I_{r_2} & -R_2^T MR_1X \\ 0 & -X^T R_1^T MR_1 & 0 & -X^T R_1^T MR_1X \end{bmatrix} \in \mathbb{R}^{n \times n}, \quad (2.52)$$

and

$$B = \begin{bmatrix} -M & 0 & -MR_2 & 0 \\ -R_1^T M & I_{r_1} & -R_1^T MR_2 & 0 \\ -R_2^T M & 0 & -R_2^T MR_2 & 0 \\ -X^T R_1^T M & 0 & -X^T R_1^T MR_2 & I_{r_3} \end{bmatrix} \in \mathbb{R}^{n \times n}. \quad (2.53)$$

Then both  $A$  and  $B$  are rank deficient matrices and the pair  $(A, B)$  is monotone.

*Proof.* It can be seen that each of last  $r_3$  columns of  $A$  is a linear combination of the columns  $r + 1, r + 2, \dots, r + r_1$ , showing the rank-deficiency of  $A$ . The rank-deficiency

of  $B$  follows similarly since each of the columns  $r + r_1 + 1$  to  $r + r_1 + r_2$  is linear combination of the first  $r$  columns.

To prove the monotonicity of the pair  $(A, B)$  consider the vectors  $u, v \in \mathbb{R}^n$  with the block structure  $u = [u_1, u_2, u_3, u_4]$  and  $v = [v_1, v_2, v_3, v_4]$ , satisfying  $Au + Bv = 0$ . A rearrangement of the variables yields the equivalent equality

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} M & MR_1 & MR_2 & MR_1X \\ R_1^T M & R_1^T MR_1 & R_1^T MR_2 & R_1^T MR_1X \\ R_2^T M & R_2^T MR_1 & R_2^T MR_2 & R_2^T MR_1X \\ X^T R_1^T M & X^T R_1^T MR_1 & X^T R_1^T MR_2 & X^T R_1^T MR_1X \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}.$$

Let us denote by  $\widetilde{M}$  the matrix from the right-hand side. Also observe that  $u^T v = [u_1, v_2, u_3, v_4]^T [v_1, u_2, v_3, u_4]$ .

If we take a look at Proposition 2.3 with  $R = [R_1 R_2 R_1 X]$ , then, since  $M$  is positive definite, we obtain that  $\widetilde{M}$  is positive semidefinite (of rank  $r$ ). This implies  $[u_1, v_2, u_3, v_4]^T [v_1, u_2, v_3, u_4] \geq 0$ , thus  $u^T v \geq 0$ , which proves the monotonicity of the pair  $(A, B)$ .  $\square$

In order to obtain a  $\text{HLCP}(Q, R, b)$ , we generate a positive definite matrix  $M$  by using Proposition 2.2, we construct the matrices  $A$  and  $B$  by using Proposition 2.4, we generate a (sparse) nonsingular matrix  $N$ , and finally we take  $Q = NA$  and  $R = NB$ . The generation of the vector  $b$  depends on whether we want an empty solution set or not. To obtain a  $\text{HLCP}(Q, R, b)$  possessing solutions, we generate first the complementary vectors  $x$  and  $s$  and then we take  $b = Qx + Rs$ . If an infeasible problem is desired, then we use the same idea from standard form. Given a feasible

HLCP( $Q, R, b$ ), consider the following horizontal form:

$$\begin{aligned} & \begin{bmatrix} x \\ \alpha \end{bmatrix} \begin{bmatrix} s \\ \beta \end{bmatrix} = 0, \\ \begin{bmatrix} Q & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \alpha \end{bmatrix} + \begin{bmatrix} R & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} s \\ \beta \end{bmatrix} &= \begin{bmatrix} b \\ -1 \end{bmatrix}, \\ & x, \alpha, s, \beta \geq 0. \end{aligned} \tag{2.54}$$

It can be observed that the above problem is monotone and has no solution.

### 2.8.3 Generating monotone MLCPs

Mixed horizontal random linear complementarity problems are constructed by introducing  $m$  free variables in a HLCP as follows.

**PROPOSITION 2.5.** *Suppose that the HLCP( $Q, R, b$ ) is monotone, where  $Q, R \in \mathbb{R}^{n \times n}$ . Consider the orthogonal matrix  $G \in \mathbb{R}^{(m+n) \times (m+n)}$  whose columns represents an orthonormal basis of  $\mathbb{R}^{m+n}$  with the block structure  $G = \begin{bmatrix} G_n & G_m \end{bmatrix}$ , where  $G_n \in \mathbb{R}^{(m+n) \times n}$  and  $G_m \in \mathbb{R}^{(m+n) \times m}$ . Also let  $K$  be a  $m \times m$  nonsingular matrix. Then the MLCP( $G_n Q, G_n R, G_m K, G_n b$ ) is monotone and infeasible or solvable whenever HLCP( $Q, R, b$ ) is infeasible or solvable, respectively.*

*Proof.* We use Lemma 2.1 to prove this proposition. First we observe that  $K^T G_m^T G_n = 0$  and  $\dim \text{Ker}(K^T G_m^T) = m + n - \dim \text{Ran}(G_m K) = n$ , hence the columns of  $G_n$  represent an orthonormal basis of  $\text{Ker}(K^T G_m^T)$ . Then Lemma 2.1 shows that MLCP( $G_n Q, G_n R, G_m K, G_n b$ ) is monotone and infeasible or solvable if and only if HLCP( $G_n^T G_n Q, G_n^T G_n R, G_n^T G_n b$ ) is monotone and infeasible or solvable, respectively. But the HLCP( $G_n^T G_n Q, G_n^T G_n R, G_n^T G_n b$ ) is exactly HLCP( $Q, R, b$ ) and the proposition is proved.  $\square$

Obtaining a sparse MLCP from a sparse HLCP by using the above proposition is possible only if both the random nonsingular matrix  $K$  and the random orthogonal matrix  $G$  are sparse. The generation of a random nonsingular sparse matrix with a desired sparsity percentage can be done by using MATLAB's built-in functions *sprand* and *sprandn*. Since MATLAB does not have a function for the generation of a random *sparse* orthogonal matrix, we have developed a routine that constructs such matrices. In order to generate a  $n \times n$  random sparse orthogonal matrix  $G$  we first generate random dense orthogonal matrices  $H_i \in \mathbb{R}^{k \times k}$ ,  $i = 1, 2, \dots, \lfloor \frac{n}{k} \rfloor$  and  $H_{\lfloor n/k \rfloor + 1} \in \mathbb{R}^{(n \bmod k) \times (n \bmod k)}$ . The natural number  $k$  is chosen to be much smaller than  $n$ , usually  $k \in [4, 20]$ , depending on the sparsity percentage wanted for  $G$ . The matrices  $H_i$  are obtained from QR factorizations of dense random matrices of corresponding sizes. The matrix  $G$  is obtained by randomly permuting the rows and columns of the matrix  $H$  having the matrices  $H_i$  along its diagonal, *i.e.*,

$$G = P_1 H P_2, \quad H = \begin{bmatrix} H_1 & 0 & \dots & 0 \\ 0 & H_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & H_{\lfloor n/k \rfloor + 1} \end{bmatrix}$$

with  $P_1, P_2 \in \mathbb{R}^{n \times n}$  being random permutation matrices.

Since

$$H H^T = \begin{bmatrix} H_1 H_1^T & 0 & \dots & 0 \\ 0 & H_2 H_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & H_{\lfloor n/k \rfloor + 1} H_{\lfloor n/k \rfloor + 1}^T \end{bmatrix} = I,$$

and

$$H^T H = \begin{bmatrix} H_1^T H_1 & 0 & \dots & 0 \\ 0 & H_2^T H_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & H_{[n/k]+1}^T H_{[n/k]+1} \end{bmatrix} = I,$$

it follows immediately that  $H$  is orthogonal. Also  $P_1$  and  $P_2$  are orthogonal since they are permutations matrices. Hence we have  $GG^T = P_1 H P_2 P_2^T H^T P_1^T = I = P_2^T H^T P_1^T P_1 H P_2 = G^T G$ , showing that our technique yields an orthogonal matrix  $G$ .

## 2.9 Numerical simulations

In this section we will present numerical experiments involving sparse monotone MLCPs that are randomly generated according to Section 2.8. We show that the homogenization technique introduced in Section 2.3 successfully retrieves the solution or detects the infeasibility of this class of MLCPs. We also present numerical evidence that advocates the use of our homogenization method when dealing with sparse MLCPs.

The first aspect we are interested in is to check if the numerical values of  $\tau$  and  $\kappa$  found by using the numerical algorithm proposed in Section 2.7 represent valid certificates of solvability or infeasibility in the sense of Theorem 2.2 and Table 2.1. We have used random monotone MLCPs generated with the technique introduced in Section 2.8. Tables 2.2 and 2.3 list the results of experiments for solvable and infeasible problems, respectively. The first two columns of these tables list the dimensions  $n$  and  $m$  of the MLCPs. The numerical values of  $\tau$  and  $\kappa$  at the solution are shown in the third and fourth columns. The last two columns show the complementarity measure  $\mu$ , and the norm of the infeasibility  $\|r\| = \|Ax + Bs + Cy - b\|$  of the original

$n$	$m$	$\tau$	$\kappa$	$\mu$	$\ r\ $
500	125	1.51e+000	4.11e-010	1.95e-009	2.00e-014
1000	250	1.17e+000	3.69e-009	9.90e-009	2.90e-013
1500	375	1.11e+000	8.27e-010	2.26e-009	3.00e-014
2000	500	1.10e+000	3.01e-009	7.90e-009	3.90e-013
2500	625	1.13e+000	1.49e-009	3.96e-009	4.00e-014
3000	750	1.19e+000	1.67e-009	4.63e-009	6.40e-013
3500	875	1.22e+000	8.95e-010	2.79e-009	7.00e-014
4000	1000	1.26e+000	5.82e-010	1.93e-009	1.20e-013
4500	1125	1.32e+000	6.93e-010	2.10e-009	1.20e-013
5000	1250	1.27e+000	1.59e-009	4.92e-009	5.20e-013

Table 2.2: Certificates for a feasible monotone MLCP

MLCP in the feasible case (Table 2.2), and  $\|r\| = \|F(x, \tau, s, \kappa, y)\|$  of the homogeneous problem in the infeasible case (Table 2.3).

As it can be seen in Table 2.2, solving the HMCP actually solves the original MLCP. Another observation would be the fact that the values of  $\tau$  at the solution are  $O(1)$  which causes the solution of the MLCP to have the same accuracy as that of the HMCP (recall that the solution of the MLCP is obtained from the solution of the HMCP via a rescaling with  $1/\tau$ ). Also Table 2.3 shows that the certificates of infeasibility, i.e.,  $\kappa > 0$ , are correctly retrieved. A final observation would be the maximal complementarity property of  $\tau$  and  $\kappa$  for both the feasible and infeasible cases.

In what follows we compare the performance in terms of execution time and memory requirements of our homogenization method and Andersen and Ye's homogeniza-

$n$	$m$	$\tau$	$\kappa$	$\mu$	$\ r\ $
500	125	5.33e-008	5.93e+000	4.40e-007	5.67e-009
1000	250	1.11e-007	1.27e+001	1.83e-006	1.25e-008
1500	375	1.05e-007	1.90e+001	2.60e-006	1.33e-008
2000	500	1.43e-008	2.23e+001	3.80e-006	2.72e-009
2500	625	1.85e-007	2.01e+001	3.14e-006	3.24e-008
3000	750	5.11e-007	1.52e+001	3.86e-006	1.10e-007
3500	875	5.66e-007	1.75e+001	3.66e-006	1.44e-007
4000	1000	7.60e-007	1.57e+001	3.83e-006	2.35e-007
4500	1125	9.57e-007	1.48e+001	4.07e-006	3.50e-007
5000	1250	9.85e-007	2.11e+001	3.90e-006	3.22e-007

Table 2.3: Certificates for an infeasible monotone MLCP

tion method [4]. While the former homogenization works directly on the MLCPs, the latter can be applied only to SLCPs since it can deal only with explicitly defined complementarity problems. However, under the monotonicity assumption, a MLCP can be transformed to a SLCP, hence, one can use Andersen and Ye's homogenization model to find the numerical solution or to detect infeasibility of a MLCP.

The transformation of a monotone MLCP( $A, B, C, b$ ) to an equivalent monotone SLCP is realized in two phases. In the first phase the free variables of the MLCP are removed using the technique from Lemma 2.1 and, since the MLCP satisfies Assumption 2.2, a monotone HLCP( $Q, R, b$ ) is obtained. The orthonormal basis of  $C^T$  required by the above lemma is found by performing the singular value decomposition of  $C^T$ .

In the second phase, the monotone HLCP( $Q, R, b$ ) is transformed to a monotone SLCP( $M, b$ ) using the reduction method from [46]. We choose to briefly describe it

here for clarity purposes. The reduction requires a maximal set of linear independent columns of  $Q$ . Using subscripts to denote columns, let  $Q_{i_1}, Q_{i_2}, \dots, Q_{i_L}$  denote this set. Also define the matrices  $C$  and  $D$  by

$$C_j = \begin{cases} Q_j, & \text{if } j \in i_1, i_2, \dots, i_L \\ R_j, & \text{otherwise} \end{cases} \quad \text{and} \quad D_j = \begin{cases} R_j, & \text{if } j \in i_1, i_2, \dots, i_L \\ Q_j, & \text{otherwise} \end{cases}.$$

It can be proved that the matrix  $C$  is invertible and  $\text{SLCP}(C^{-1}D, C^{-1}b)$  is monotone. Moreover, once a solution to  $\text{SLCP}$  is available, the solution of  $\text{HLCP}$  is readily obtained from a rearrangement of complementarity variables. The fact that even though  $C$  is sparse,  $C^{-1}$  is full, turns out to be crucial in the performance of the numerical scheme used to solve reduced  $\text{SLCP}$ s, as we will show later. Our implementation uses a LU factorization of  $Q$  to find maximal set of linear independent columns of  $Q$  and the MATLAB function *inv* to compute  $C^{-1}$ .

In building Tables 2.4 and 2.5 we have randomly generated feasible and infeasible monotone sparse  $\text{MLCP}$ s of various sizes,  $n = 500, 1000, \dots, 5000$ ,  $m = n/4$  and we have solved the corresponding  $\text{HMCP}$ s problems. The third and fourth columns of Tables 2.4 and 2.5 show the number of iterations and execution times needed by the predictor-corrector Mehrotra-type algorithm presented in Section 2.7. After that we have transformed the  $\text{MLCP}$  to a  $\text{SLCP}$  as described in the previous paragraphs and recorded the conversion time (column five in Tables 2.4 and 2.5). By the conversion time we mean the execution time needed to accomplish the two reduction phases. The obtained monotone  $\text{SLCP}$  is then homogenized using Andersen and Ye's method [4] and solved with the interior-point method proposed by the same authors in [3]. The sixth column in Tables 2.4 and 2.5 shows the number of iterations needed by this algorithm to solve the  $\text{SLCP}$ . The execution times of the algorithm when using MATLAB's sparse and dense linear algebra are displayed in the seventh and eighth columns, respectively. "OOM" (out of memory) shows that the problem was not

$n$	$m$	MLCP		Conv	SLCP		
		iter	$t$	$t$	iter	$t_{sparse}$	$t_{dense}$
500	125	12	1.29	1.94	13	44.96	4.31
1000	250	12	7.76	12.30	15	608.90	28.27
1500	375	13	26.42	39.93	13	3187.58	72.79
2000	500	12	59.11	98.95	14	8284.77	164.53
2500	625	12	114.70	201.80	11	14550.27	243.53
3000	750	12	247.08	363.57	14	36358.05	515.56
3500	875	12	361.29	577.16	12	OOM	846.33
4000	1000	12	552.47	869.26		OOM	OOM
4500	1125	12	861.67	1205.81		OOM	OOM
5000	1250	11	1089.22	1699.90		OOM	OOM

Table 2.4: Solving feasible MLCPs and corresponding reduced SLCPs via HMCP and Andersen and Ye’s homogeneous models, respectively.

solved due to the lack of memory. Let us mention that execution times were measured in seconds.

Several interesting observations can be made based on Tables 2.4 and 2.5. First of all, notice that converting the MLCP to SLCP takes more time than solving the problem via HMCP. Moreover, the execution time needed to solve the SLCP with Andersen and Ye’s method is considerably larger than the execution time needed to solve the same problem via HMCP.

It must be pointed out that different percentages of fill-in in the MLCP’s data may lead to different conclusions when comparing the execution times of the two approaches. However, in terms of the memory requirements, sparse MLCPs can be

$n$	$m$	MLCP		Conv	SLCP		
		iter	$t$	$t$	iter	$t_{sparse}$	$t_{dense}$
500	125	13	1.34	1.80	12	41.48	3.99
1000	250	12	7.48	12.52	13	363.46	24.80
1500	375	12	29.20	40.22	13	2257.53	73.06
2000	500	12	60.08	98.33	13	8689.07	155.90
2500	625	12	123.84	201.74	11	16256.95	246.69
3000	750	12	214.75	369.63	11	28970.54	409.69
3500	875	12	448.54	577.56	11	OOM	724.17
4000	1000	12	674.99	876.33		OOM	OOM
4500	1125	12	959.86	1194.37		OOM	OOM
5000	1250	12	1311.67	1612.35		OOM	OOM

Table 2.5: Solving infeasible MLCPs and corresponding reduced SLCPs via HMCP and Andersen and Ye’s homogeneous models, respectively.

solved via HMCP as long the problem’s data fits the memory, but via SLCP-based alternative only up to  $n = 3500$  (no matter how sparse the MLCP is!).

## 2.10 Conclusions

This chapter has introduced a new homogenization technique for monotone mixed linear complementarity problems. To the best of our knowledge, it is the first homogeneous model that deals with implicitly defined complementarity. We have proved that the transformed problem offers an infeasibility certificate or provide a solution of the original problem. We have also shown that the interior-point path-following

methods can be used to obtain the numerical solutions of the augmented problem and safely retrieve the certificates of infeasibility/solvability. We also proposed a practical Mehrotra-type interior-point numerical scheme for solving the homogeneous problems. Numerical experiments performed on randomly generated MLCPs show that the numerical method we proposed is faster and solves larger problems than other homogenization methods.

# Chapter 3

## Simulating rigid body systems with contact and friction

### 3.1 Introduction

The dynamic rigid multi-body contact problem is concerned with predicting the motion of several rigid bodies in contact and is one of the fundamental paradigms in modern computational science. It appears in the description of fuel motion in the pebble bed reactor [43], in the compaction of nanopowders [52, 17], and in the study of biological membranes [72, 51, 81, 45]. Such simulations are also used extensively in structural engineering [35], pedestrian evacuation dynamics [50], granular matter [69], robotics simulation and design [36], and virtual reality [10].

Approaches used in the past for the numerical approximation of rigid multi-body dynamics with contact and friction include piecewise differential-algebraic equations approaches [49], acceleration-force linear complementarity problem (LCP) approaches [16, 65, 80], penalty approaches [29, 73, 74, 64], and velocity-impulse LCP-based time-

stepping methods [75, 76, 8, 13, 68]. LCP-based time-stepping schemes for simulating multi-body systems are formulated as LCPs with copositive matrices. Such LCPs are generally solved by means of Lemke-type algorithms, and solvers such as the PATH solver [28, 38] have proved to be robust. For large systems (beyond a few thousand contacts), however, the PATH solver or any other pivotal algorithm becomes impractical from a computational point of view [12, 79].

The computational difficulties associated with the standard frictional LCP formulation cannot be avoided even for small friction coefficients. In this context, in [11], a simple example is used to show that the solution set of the underlying LCP fails to be convex for any nonzero friction coefficients and therefore no polynomial time algorithms are known to exist.

When solving large-scale multi-body dynamics, a relaxation of the standard (copositive) LCP formulation is desirable. The convex relaxation introduced in [9] is convergent in the same weak sense as the original, nonconvex scheme [76]. This relaxation formulates the integration step as a convex quadratic program (QP) for which state-of-the-art solvers are available.

In this chapter we investigate the performance of several solvers for QP problems that appear from the relaxed formulation of multirigid body dynamics with contact and friction. The examples are obtained by simulating granular flow motion in a system similar to the pebble-bed reactor described in [43]. A description of the simulated multi-body system is given in Section 3.2.1.

We note that QP approaches are popular in the graphics community for simulating rigid-body dynamics with contact and friction [59, 48, 33]. Their most common instance insofar as friction is concerned is probably the box friction model approach. There, the normal force is prescribed either directly or following a frictionless presolve,

which is also a convex QP [48]. Variants of this approach are used as an option in many major physics engines for games, such as ODE (Open Dynamics Engine), OpenTissue, and Vortex (some discussion in [20]). All the QPs appearing in such approaches have a sparsity structure with an incidence graph closely related to the graph with nodes in the center of the bodies and edges between bodies in potential contact. The sparsity we encounter here is based on the same graph. The number of bodies (thousands) and contacts (tens of thousand) we consider is comparable or larger to the one considered in most of the applications targeted by such physics engines [33]. We therefore expect that our findings will extend to those applications as well. We point out, however, that to our knowledge the method introduced in [9] is the only optimization-based method for rigid-body dynamics that is provably convergent, at least in a weak sense [76], to a continuous-time solution for any friction coefficient. This observation motivates our choice of subject for the computational experiment.

This chapter is organized in two main parts. In Section 3.2 we introduce the optimization problems to be solved. In this context we specialize the QP-formulation of [9] to the application described in Section 3.2.1 and address the modeling of the pebble bed reactor application. We plan to exploit two QP formulations: the primal QP formulation and the dual QP formulation, which takes the form of a bound constrained minimization problem. For these formulations, four QP-solvers are used to compare computational efficiency and to analyze ensemble properties of the simulated trajectories. We use two interior-point solvers, MOSEK [1] and OOQP [41] (with two formulations of the linear algebra, one using MA27 [32] and the other based on CHOLMOD [26, 22, 25, 24]), and two projected gradients solvers (on the dual formulation), TRON [54] and BLMVM [18]. A comparison between these solvers

is given in Section 3.3, details of the numerical experiments are given in Section 3.4, and a summary of our observations is given in Section 3.5.

## 3.2 Quadratic Programming Subproblems of Time-Stepping Methods

We now discuss the origin and structure of the QP subproblems in the time-stepping approach in [9]. We will apply the formulation to the problem of the simulation of the granular flow of the fuel pebbles in a pebble-bed reactor (PBR) [43]. This example problem has the advantage that we can easily scale it up in the number of the rigid bodies and contacts, making it ideal for extrapolating the behavior of the solvers for increasingly larger systems.

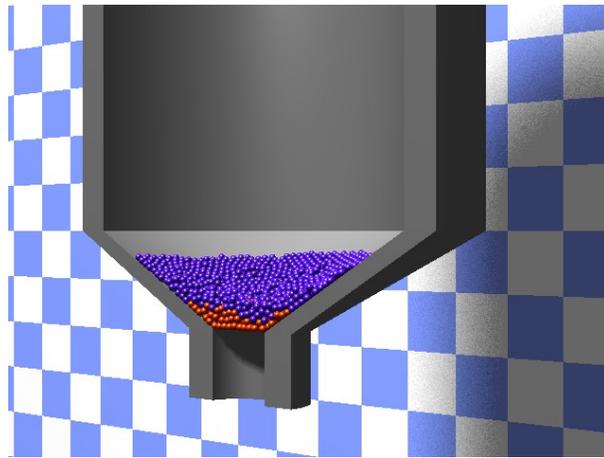


Figure 3.1: Cross-section of the reactor vessel with 3200 pebbles at the end of the simulation. The pebbles are colored based on the originating position.

We present an image of the reactor vessel in Figure 3.1, with all pebbles at rest. The rigid bodies – the pebbles – are tennis-ball-sized and contain uranium oxide. They are extracted from the bottom of the vessel and reinserted through the top. When

fully loaded, the reactor has about 400,000 such bodies, but important assessments can be extracted from simulations involving a smaller number. The reactor vessel in which the motion of the pebbles is simulated is composed of a truncated cone and a cylinder that is opened at both ends, see Figure 3.1. For modeling purposes, we index the cylindrical surface of the vat by  $-3$ , the lateral surface of the truncated cone by  $-2$ , and the bottom of the vat by  $-1$ .

Most of the following modeling steps apply to general rigid multi-body dynamics. However, some topics particular to the PBR simulation will be pointed out as they occur.

### 3.2.1 The Model

In this section we describe the various modeling steps that convert a rigid multi-body dynamics problem with contact and friction to a sequence of quadratic programs such as (3.10). The model is based on the convex relaxation introduced in [9]. The merits and shortcomings of that relaxation, insofar accuracy of predicting frictional behavior is concerned, are presented in [9] and are not the subject of this work. Here, we focus on the setup and resolution of (3.10).

Assume that the rigid-body system is composed of  $N$  bodies. The position of body  $i$  is  $q^{(i)} = (x_i, y_i, z_i, \theta_i, \alpha_i, \gamma_i)^T$ ,  $i = 0, \dots, N-1$ , where the first three components are the Cartesian coordinates of the center (in a fixed inertial frame) and the last three represent the orientation of a reference point  $P^{(i)}$  on the sphere. The position  $q$  of the entire system is obtained by concatenating the positions of each body, namely,  $q = \left( q^{(0)T}, q^{(2)T}, \dots, q^{(N-1)T} \right)^T$ . In a similar fashion, we define the generalized velocity of the system to be  $v \in \mathbb{R}^{6N}$ ,  $v = \Gamma(q) \frac{dq}{dt}$ . Here  $\Gamma(q)$  is a smooth mapping that converts the derivatives of the position coordinates to the generalized velocities [49].

For rigid bodies, as opposed to material points,  $\Gamma(q)$  is different from the identity [49].

**Nonpenetration constraints.** For bodies  $i$ , and  $j$ , we assume that we have signed gap functions  $\Phi^{(i,j)}(q)$ , such that

$$\Phi^{(i,j)}(q) = \begin{cases} < 0 & \text{the two bodies penetrate,} \\ = 0 & \text{the two bodies are touching,} \\ > 0 & \text{the two bodies are separated.} \end{cases}$$

In this case, the nonpenetration constraints simply become

$$\Phi^{(i,j)}(q) \geq 0.$$

For PBR, the nonpenetration constraints are imposed between *pebble-pebble* and *pebble-wall* interaction. The constraint indices are  $(i, j)$ , for  $(i \in \{-3, \dots, (N - 1)\}, j \in \{0, \dots, (N - 1)\}$  and  $j > i$ ). In addition, the signed gap functions are particularly easy to define. For example, for two spheres of radius  $R$ , and with centers of mass at positions  $x_1, y_1, z_1$  and  $x_2, y_2, z_2$ , a signed gap function is

$$\Phi^{(i,j)} = (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 - 4R^2.$$

The pebble-wall gap functions use the distance from a point to a simple surface and are also immediate to define.

**Contact specifications.** For each contact we define the normal and tangential directions in generalized coordinates as follows. Consider the generic contact depicted in Figure 3.2. Let  $\vec{n}$  denote the unit outward normal for the horizontally aligned body. Let  $C_1$  and  $C_2$  be the centers of mass for each body, and let  $\vec{r}_1$  and  $\vec{r}_2$  be the position vectors of the contact point relative to  $C_1$  and  $C_2$  respectively (all vectors are represented in world frame coordinates). Then the 3-dimensional vector

$\vec{n}$  is mapped in generalized coordinates into the 12-dimensional vector  $n$ , defined as follows

$$\vec{n} \mapsto n := \begin{pmatrix} \vec{n} \\ \vec{r}_1 \times \vec{n} \\ -\vec{n} \\ -\vec{r}_2 \times \vec{n} \end{pmatrix}. \quad (3.1)$$

In a similar fashion one obtains the generalized coordinates vectors  $t_1$  and  $t_2$  from  $\vec{t}_1$  and  $\vec{t}_2$ , respectively.

The Coulomb friction model prescribes that the tangential force is inside a disk proportional to the one in Figure 3.2. To allow for the use of a quadratic programming approach (or an LCP in the unrelaxed case), we use a polygonal approximation of the friction disk in Figure 3.2(b) [75, 8]. The disk is approximated by an inscribed polygon, whose quality depends on the number  $p$  of tangent directions used.

We define those  $p$  tangent vectors,  $\vec{d}_s$ , by

$$\vec{d}_s := \cos\left(\frac{2\pi s}{p}\right) \vec{t}_1 + \sin\left(\frac{2\pi s}{p}\right) \vec{t}_2, \quad s = 1, \dots, p. \quad (3.2)$$

Clearly the vectors  $\vec{d}_s$  are direction vectors in  $\mathbb{R}^3$ , and any point in the  $\text{span}\{\vec{d}_1, \dots, \vec{d}_p\}$  can be written as a nonnegative linear combination of these vectors. It is easy to see

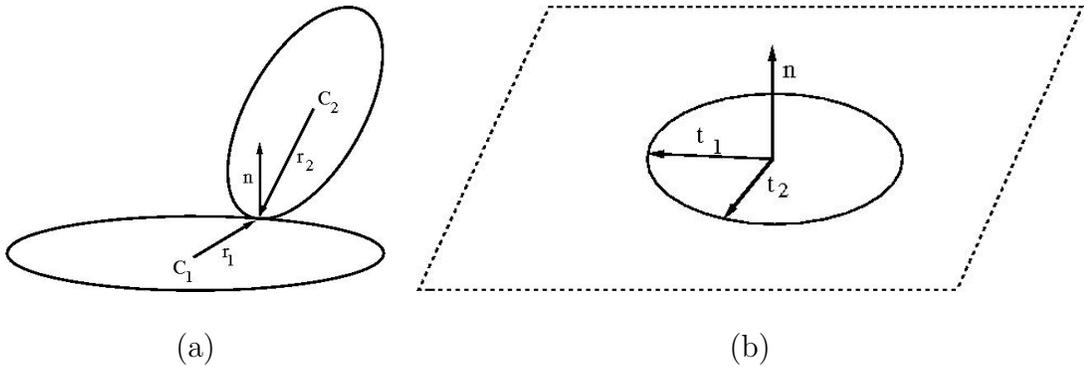


Figure 3.2: (a) Generic contact between two bodies, (b) Tangent space at contact.

that the generalized coordinate version of the directions in (3.2) are obtained in the same fashion, namely,

$$d_s := \cos\left(\frac{2\pi s}{p}\right)t_1 + \sin\left(\frac{2\pi s}{p}\right)t_2, \quad s = 1, \dots, p. \quad (3.3)$$

Given that, for a system of  $N$  bodies the configuration space (the space where  $q$  lives) has dimension  $6N$ , we embed the contact data in  $\mathbb{R}^{6N}$ . More precisely, assume that the  $k$ -th indexed contact is established between body  $i$  ( $i \geq 0$ ) and body  $j$  ( $j > i$ ). The normal direction in the  $6N$  dimensional space is then given by

$$n^{(k)} = \left( O_{1,6(i-1)}, \vec{n}_k^T, (\vec{r}_i \times \vec{n}_k)^T, O_{1,6(j-i-1)}, -\vec{n}_k^T, (-\vec{r}_j \times \vec{n}_k)^T, O_{1,6(N-j)} \right)^T, \quad (3.4)$$

where  $O_{1,\alpha}$  represents a zero row vector of length  $\alpha$ , whenever  $\alpha > 0$  and the empty vector otherwise. Here  $n_k$  is the three-dimensional normal vector at the contact  $k$ .

In the case of PBR, for a *pebble-wall* interaction  $(i, j)$ ,  $i < 0$ , only the second nonzero block will contribute to  $n^{(k)}$ .

In the same fashion one defines the tangential directions  $d_s^{(k)} \in \mathbb{R}^{6N}$ ,  $s = 1, \dots, p_k$ , namely,

$$d_s^{(k)} = \left( O_{1,6(i-1)}, \vec{d}_{sk}^T, (\vec{r}_i \times \vec{d}_{sk})^T, O_{1,6(j-i-1)}, -\vec{d}_{sk}^T, (-\vec{r}_j \times \vec{d}_{sk})^T, O_{1,6(N-j)} \right)^T. \quad (3.5)$$

In (3.5),  $d_{sk}$ ,  $s = 1, \dots, p_k$  are the  $\mathbb{R}^{p_k}$  direction vectors used in the approximation of the friction disk at contact  $k$ . The matrix associated with the polyhedral approximation of the friction disk at contact  $(k)$  is the matrix having its columns the directions  $d_s^{(k)}$ , *i. e.*,  $D^{(k)} \in \mathbb{R}^{6N \times p_k}$ ,  $D^{(k)} = \left( d_1^{(k)}, \dots, d_{p_k}^{(k)} \right)$ , where  $p_k$  represents the number of friction generators for contact  $(k)$ .

In the formulation of the integration step we will deal with matrices  $\widehat{D}^{(k)} \in \mathbb{R}^{6N \times p_k}$

of the form

$$\widehat{D}^{(k)} = \left( n^{(k)} + \mu d_1^{(k)}, \dots, n^{(k)} + \mu d_{p_k}^{(k)} \right), \quad (3.6)$$

where  $\mu \in (0, 1]$  is the friction coefficient, which is assumed to be the same for all contacts. We are interested in the maximal value of  $p_k$  ( $p_k < 6N$ ) for which the matrix  $\widehat{D}^{(k)}$  has full column rank,  $\text{rank}(\widehat{D}^{(k)}) = p_k$ . It is easy to see that for  $p_k = 3$ ,  $\text{rank}(\widehat{D}^{(k)}) = p_k$ , while for  $p_k > 3$  we obtain a rank-deficient matrix. This observation will be used when formulating the integration timestep as a bound-constrained minimization problem (this is what we call the dual formulation). For the PBR simulation we choose  $p_k = 3$  for all contacts  $k$ .

**External and inertial forces.** We denote by  $M \in \mathbb{R}^{6N \times 6N}$  the generalized mass matrix of the system. In general, in a fixed coordinate frame, this matrix depends explicitly on the position  $q$ , *i.e.*,  $M := M(q)$ . We also denote by  $k_{app}$  the sum between the external forces and the inertial forces. The inertial forces involve derivatives of  $M(q)$  [49].

For the PBR example, we are dealing only with spherical bodies. The generalized mass matrix is diagonal with positive entries and constant with respect to  $q$  [49]. Since the mass matrix is constant, the inertial forces are zero. This implies that, besides contact forces, the only forces acting on the system are external forces.

For PBR, because of the heaviness of uranium, we can ignore the effect of the cooling flow over the dynamics [43]. Therefore, we can assume that only gravitational forces are acting, and the (noncontact) applied forces  $k_{app} \in \mathbb{R}^{6N}$  can be written as

$$k_{app} = (u^T, \dots, u^T)^T.$$

Here  $u \in \mathbb{R}^6$ ,  $u = (0, 0, -g, 0, 0, 0)^T$ , for some positive constant  $g$ .

**Linearized active contacts constraints.** Given a current position  $q$  of the system, we compute the set of active contacts by using the signed distance functions  $\Phi^{(i,j)}(q)$ .

Given a positive scalar  $\epsilon$ , a contact  $(k)$  corresponding to the rigid-body pair  $(i, j)$  is considered active if

$$\Phi^{(k)}(q) = \Phi^{(i,j)}(q) \leq \epsilon.$$

For a given configuration  $q$ , we denote by  $\mathcal{A}(q, \epsilon)$  the set of all active contacts. More precisely,

$$\mathcal{A}(q, \epsilon) = \{k \in \mathbb{Z}_+^2 \mid (k) = (i, j), \Phi^{(i,j)}(q) \leq \epsilon\}. \quad (3.7)$$

If  $(k) \notin \mathcal{A}(q, \epsilon)$ , the corresponding nonpenetration constraint is simply ignored by the QP (3.10). If  $(k) \in \mathcal{A}(q, \epsilon)$ , then the nonpenetration constraint  $\Phi^{(i,j)} > 0$  is enforced at time  $t^l$  and position  $q^l$  by

$$(n^{(k)}(q^l))^T v + \mu (d_s^{(k)}(q^l))^T v \geq -\frac{1}{h} \Phi^{(k)}(q^l), \quad s = 1, 2, \dots, p_k. \quad (3.8)$$

Here  $h$  is the timestep of the scheme. It does not need to be constant for stability [7], but we choose it so for graphical simplicity. The first and last terms in this equation are the linearization of the nonpenetration constraints. The last term is stable as  $h \rightarrow 0$  [7, 9]. The middle term is unique to the scheme in [9]. Its physical significance is based on a microscopic realization of surface asperities that result in macroscopic friction coefficient  $\mu$ .

It has been shown that the value of  $\epsilon$  does not affect the convergence as  $h \rightarrow 0$  [7, 9]. But its choice has important practical consequences. A value that is too large results in an exceedingly large QP. The QP is still consistent but may be computationally expensive. A value that is too small may result in too many nonpenetration constraints being dropped and in excessive penetration at time step  $l + 1$ . An appropriate value for  $\epsilon$  should be comparable to the product between the maximum velocity in the system and the timestep.

**Newton's second law.** The discretized version of Newton's second law is written

at the velocity-impulse level as

$$M (v^{l+1} - v^l) - z^{l+1} = hk_{app}, \quad (3.9)$$

where  $hk_{app}$  are the external impulses and  $z^{l+1}$  represent the contact impulses (normal and tangential/frictional contact impulses):

$$z^{l+1} = \sum_{k \in \mathcal{A}(q^l, \epsilon)} \sum_{s=1}^{p_k} \beta_s^{(k)} (n^{(k)}(q^l) + \mu d_s^{(k)}(q^l)).$$

Here  $M$  is the mass matrix, and each vector of multipliers satisfies  $\beta^{(k)} = \left( \beta_1^{(k)}, \beta_2^{(k)}, \dots, \beta_{p_k}^{(k)} \right) \in \mathbb{R}^{p_k}$  and  $\beta^{(k)} \geq 0$ .

### 3.2.2 The Integration Step

In what follows we present the formulation proposed in [9] and its dual. Here, we consider only the case of totally plastic collisions. The scheme can be modified to accommodate partially elastic or totally elastic collisions by means of a restitution coefficient [5]. We point out, however, that the issue of predictive modeling of simultaneous nonplastic collisions is far from being settled in rigid body dynamics [21].

#### Primal Formulation

Let  $h > 0$  denote the size of the integration timestep. We denote by  $q^l$  the position and by  $v^l$  the velocity of the system at time  $t_l = lh$ . In the optimization-based time-stepping scheme introduced in [9], the new velocity  $v^{l+1}$  is obtained by solving the following quadratic problem:

$$\begin{aligned} \min \quad & \frac{1}{2} v^T M v + (f^l)^T v \\ \text{s.t.} \quad & (n^{(k)}(q^l))^T v + \mu \left( d_s^{(k)}(q^l) \right)^T v \geq -\frac{1}{h} \Phi^{(k)}(q^l) \\ & k \in \mathcal{A}(q^l, \epsilon), \quad s = 1, 2, \dots, p_k. \end{aligned} \quad (3.10)$$

In (3.10),  $f^l$  is obtained by the following formula:

$$f^{(l)} = -v^l - hk_{app}. \quad (3.11)$$

Equations (3.9) and (3.8) are satisfied as part of the optimality conditions for (3.10).

### Dual Formulation

The dual formulation can be obtained by standard duality techniques. In our case, we assign the Lagrange multipliers  $\lambda$  to the constraints in (3.10). Then, we write the optimality conditions for (3.10), and we eliminate  $v$  using the positive definiteness of  $M$ . This procedure results in the dual QP program (3.13).

Let us denote by  $A^l$  and  $b^l$  the matrix and the right-hand side, respectively, of the inequality constraints in (3.10). In terms of the notation introduced in (3.6), the matrix  $A^l$  has the form

$$A^l = \begin{pmatrix} \left( \widehat{D}^{k_1}(q^l) \right)^T \\ \vdots \\ \left( \widehat{D}^{k_p}(q^l) \right)^T \end{pmatrix}, \quad (3.12)$$

where the active set  $\mathcal{A}(q^l, \epsilon) = \{k_i \mid i = 1, \dots, p\}$ . The vector  $b^l$  is composed of block vectors in  $\mathbb{R}^3$ , with the block corresponding to contact  $k_i$  having all its components equal to  $-\frac{1}{h}\Phi^{(k_i)}(q^l)$ . In the notation described above, the dual problem takes the form

$$\begin{aligned} \min \quad & \frac{1}{2}\lambda^T P^l \lambda + (\kappa^l)^T \lambda, \\ \text{s.t.} \quad & \lambda \geq 0 \end{aligned} \quad (3.13)$$

where  $P^l = A^l M^{-1} (A^l)^T$  and  $\kappa^l = -b^l - A^l f^l$ .

The dual formulation (3.13) is a bound-constrained quadratic programming problem. Therefore, for solving it, we can use not only general-purpose quadratic programming algorithms, such as interior points, but also iterative algorithms of the

projected gradient type. Therefore, it is a good formulation for benchmarking the performance of various solvers for quadratic programming.

### 3.2.3 Pointed friction cone and duality

Consider the  $\epsilon$ -active set  $\mathcal{A}(q^l, \epsilon)$ , which is obtained from (3.7) for the current configuration  $q^l$ . For  $k \in \mathcal{A}(q^l, \epsilon)$  and let  $\tilde{D}^{(k)}$  denote the matrix of generalized tangential directions, namely,

$$\tilde{D}^{(k)}(q^l) := \tilde{D}^{(k)} = \left( d_1^{(k)}, d_2^{(k)}, \dots, d_{p_k}^{(k)} \right), \quad (3.14)$$

where the generalized tangential directions  $d_s^{(k)} := d_s^{(k)}(q^l)$ ,  $s = 1, 2, \dots, p_k$  are given in (3.3). We now define the  $\epsilon$ -active friction cone  $\mathcal{FC}(q^l, \epsilon)$  by

$$\mathcal{FC}(q^l, \epsilon) = \left\{ \sum_{k \in \mathcal{A}(q^l, \epsilon)} \tilde{D}^{(k)} \beta^{(k)} \mid \beta^{(k)} \in \mathbb{R}^{p_k}, \beta^{(k)} \geq 0, \right\}. \quad (3.15)$$

We say that the friction cone is **pointed** if the following implication holds:

$$z = \sum_{k \in \mathcal{A}(q^l, \epsilon)} \tilde{D}^{(k)} \beta^{(k)} \in \mathcal{FC}(q^l, \epsilon), z = 0 \Rightarrow \beta^{(k)} = 0, \forall k \in \mathcal{A}(q^l, \epsilon). \quad (3.16)$$

In other words  $\mathcal{FC}(q^l, \epsilon)$  is pointed if it doesn't contain any proper subspaces.

It has been shown in [12] that pointedness of the friction cone implies that the Mangasarian-Fromowitz constraint qualification (MFCQ) holds for the convex program (3.10). Whenever MFCQ holds the multipliers  $\lambda$  in (3.13) are bounded [39]. Therefore, pointedness of the friction cone together with the existence of a feasible point for (3.10) guarantees no duality gap. This result is essential when comparing QP solvers that use either the primal or the dual formulation. Loss of the pointedness regularity assumption corresponds, from a physical point of view, to jamming [6], a phenomenon that does not occur in our simulations.

For isolated QPs we can compare the correctness of the results obtained from solving the two different formulations, by either measuring the duality gap or by passing from a dual solution to its primal correspondent. When running an entire simulation, however, the use of different solvers will likely cause totally different individual configurations. This is motivated by the fact that, by nature, the system is chaotic. Therefore, to measure the correctness of an entire simulation, at least partially, we use ensemble properties, such as the kinetic energy of the entire system.

### 3.3 Algorithms and Software Packages Used

We now describe the properties of several packages used to solve (3.13). We use two types of packages. The first, of the interior-point type, OOQP and MOSEK, solve the primal-dual formulation of both (3.13) and (3.10). For the OOQP solver, we use two formulations, one of which involves our adaptation of the CHOLMOD linear algebra package for use with OOQP. The second, of the projected gradient type, TRON and BLVM, solve the dual problem (3.13).

#### 3.3.1 OOQP

OOQP (Object-Oriented software for Quadratic Programming) is a C++ package for solving convex quadratic programming problems. It is based on primal-dual interior-point methods and can be used to solve a variety of forms of quadratic problems such as general sparse QPs, QPs with "box" constraints, QPs coming from support vector machines, and Huber regression problems. Its object-oriented design allows easy adaptation for specialized QP formulations or use of new linear algebra solvers.

In our experiments OOQP's general sparse formulation is used to solve the primal

form (3.10). Two distinct linear algebra packages are used: MA27 for which an interface is included in OOQP distribution and CHOLMOD for which we developed an interface and reformulated the linear systems. We denote the two versions of OOQP by OOQP-MA27 and OOQP-CHOL, respectively.

### OOQP general sparse formulation

In OOQP the convex quadratic problem subject to linear constraints is considered in the following general form:

$$\begin{aligned}
& \text{minimize} && \frac{1}{2}x^T Qx + c^T x \\
& \text{subj to:} && Ax = b \\
& && c_l \leq Cx \leq c_u \\
& && x_l \leq x \leq x_u,
\end{aligned} \tag{3.17}$$

where  $Q \in \mathbb{R}^{n \times n}$ ,  $c \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m_y \times n}$ ,  $C \in \mathbb{R}^{m_z \times n}$ ,  $c_l, c_u \in \mathbb{R}^{m_z}$ , and  $x_l, x_u \in \mathbb{R}^n$ .

It is well known that at each iteration of the interior-point method one or more linear systems need to be solved. In what follows, we describe the way OOQP manages the KKT conditions and builds the linear systems.

The Lagrangian function corresponding to the quadratic problem (QP) (3.17) is  $L(x) = \frac{1}{2}x^T Qx + c^T x + y^T (b - Ax) + \lambda^T (c_l - Cx) + \pi^T (Cx - c_u) + \gamma^T (x_l - x) + \phi^T (x - x_u)$ , hence, the KKT conditions can be written as follows:

$$\begin{aligned}
Qx - A^T y - C^T \lambda + C^T \pi - \gamma + \phi + c &= 0 \\
Ax &= b \\
0 \leq Cx - c_l = t \perp \lambda &\geq 0 \\
0 \leq c_u - Cx = u \perp \pi &\geq 0 \\
0 \leq x - x_l = v \perp \gamma &\geq 0
\end{aligned}$$

$$0 \leq x_u - x = w \perp \phi \geq 0.$$

The interior-point iterations consist of solving the perturbed KKT systems

$$F(x, y, t, u, v, w, \lambda, \pi, \gamma, \phi) = \begin{bmatrix} Qx - A^T y - C^T \lambda + C^T \pi - \gamma + \phi + c \\ -Ax + b \\ Cx - t - c_l \\ -Cx - u + c_u \\ x - v - x_l \\ -x - w + x_u \\ \lambda t - \mu_k e \\ \pi u - \mu_k e \\ \gamma v - \mu_k e \\ \phi w - \mu_k e \end{bmatrix} = 0,$$

for a sequence of positive  $\{\mu_k\}$  that converges to zero, while maintaining the positivity of  $t, u, v, w, \lambda, \pi, \gamma, \phi$ . The Newton's method is used to (approximately) solve the above nonlinear system, so the linear algebra consists of solving linear systems of

the form  $F'\Delta d = -F$ , where the Jacobian  $F'$  is given by

$$F' = \begin{bmatrix} Q & -A^T & 0 & 0 & 0 & 0 & -C^T & C^T & -I & I \\ -A & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ C & 0 & -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -C & 0 & 0 & -I & 0 & 0 & 0 & 0 & 0 & 0 \\ I & 0 & 0 & 0 & -I & 0 & 0 & 0 & 0 & 0 \\ -I & 0 & 0 & 0 & 0 & -I & 0 & 0 & 0 & 0 \\ 0 & 0 & \Lambda & 0 & 0 & 0 & T & 0 & 0 & 0 \\ 0 & 0 & 0 & \Pi & 0 & 0 & 0 & U & 0 & 0 \\ 0 & 0 & 0 & 0 & \Gamma & 0 & 0 & 0 & V & 0 \\ 0 & 0 & 0 & 0 & 0 & \Phi & 0 & 0 & 0 & W \end{bmatrix},$$

with  $\Lambda = \text{diag}(\lambda)$ ,  $\Pi = \text{diag}(\pi)$ ,  $\Gamma = \text{diag}(\gamma)$ ,  $\Phi = \text{diag}(\phi)$ ,  $T = \text{diag}(t)$ ,  $U = \text{diag}(u)$ ,  $V = \text{diag}(v)$ , and  $W = \text{diag}(w)$ .

The remaining of this section goes into the details of solving the linear system  $F'\Delta d = -F$ . Using the expression of  $F'$ , we can write the equivalent form

$$Q\Delta x - A^T\Delta y - C^T\Delta\lambda + C^T\Delta\pi - \Delta\gamma + \Delta\phi = r_Q \quad (3.18)$$

$$A\Delta x = r_y \quad (3.19)$$

$$C\Delta x - \Delta t = r_t \quad (3.20)$$

$$-C\Delta x - \Delta u = r_u \quad (3.21)$$

$$\Delta x - \Delta v = r_v \quad (3.22)$$

$$-\Delta x - \Delta w = r_w \quad (3.23)$$

$$\Lambda\Delta t + T\Delta\lambda = r_\lambda \quad (3.24)$$

$$\Pi\Delta u + U\Delta\pi = r_\pi \quad (3.25)$$

$$\Gamma\Delta v + V\Delta\gamma = r_\gamma \quad (3.26)$$

$$\Phi\Delta w + W\Delta\phi = r_\phi. \quad (3.27)$$

The exact form  $-F$  of the right-hand side is not of interest in this discussion of the linear algebra layer since it is specific to the interior-point algorithm. We denoted the right-hand side vectors by  $r_Q, r_y, r_t, r_u, r_v, r_w, r_\lambda, r_\pi, r_\gamma$ , and  $r_\phi$ .

Expressing  $\Delta\gamma$  and  $\Delta\phi$  from equations (3.26) and (3.27) respectively, allows to write  $\Delta\gamma - \Delta\phi = -V^{-1}\Gamma\Delta v + W^{-1}\Phi\Delta w + V^{-1}r_\gamma - W^{-1}r_\phi$ . Furthermore,  $\Delta v$  and  $\Delta w$  can be written in terms of  $\Delta x$  by using equations (3.22) and (3.23) respectively. Hence, we get  $\Delta\gamma - \Delta\phi = -V^{-1}\Gamma(\Delta x - r_v) + W^{-1}\Phi(-\Delta x - r_w) + V^{-1}r_\gamma - W^{-1}r_\phi$ . And if we denote  $D_1 = V^{-1}\Gamma + W^{-1}\Phi$ , then

$$\Delta\gamma - \Delta\phi = -D_1\Delta x + (V^{-1}r_\gamma - W^{-1}r_\phi + V^{-1}\Gamma r_v - W^{-1}\Phi r_w). \quad (3.28)$$

Let  $\Delta z = \Delta\lambda - \Delta\pi$ . Similar to the previous paragraph we have  $\Delta\lambda = -T^{-1}\Lambda\Delta t + T^{-1}r_\lambda$  from equation (3.24) and  $\Delta\pi = -U^{-1}\Pi\Delta u + U^{-1}r_\pi$  from equation (3.25). Now express  $\Delta t = C\Delta x - r_t$  and  $\Delta u = -C\Delta x - r_u$  by making use of equations (3.20) and (3.21) respectively. Then  $\Delta z = -T^{-1}\Lambda(C\Delta x - r_t) + U^{-1}\Pi(-C\Delta x - r_u) + T^{-1}r_\lambda - U^{-1}r_\pi = -(T^{-1}\Lambda + U^{-1}\Pi)C\Delta x + T^{-1}r_\lambda - U^{-1}r_\pi + T^{-1}\Lambda r_t - U^{-1}\Pi r_u$ . Therefore we can write

$$C\Delta x + D_2^{-1}\Delta z = D_2^{-1}(T^{-1}r_\lambda - U^{-1}r_\pi + T^{-1}\Lambda r_t - U^{-1}\Pi r_u), \quad (3.29)$$

where  $D_2 = T^{-1}\Lambda + U^{-1}\Pi$ .

Both  $D_2^{-1}$  and  $D_1^{-1}$  exist, since  $T^{-1}\Lambda + U^{-1}\Pi$  and  $V^{-1}\Gamma + W^{-1}\Phi$  are diagonal matrices with strictly positive diagonal entries. We substitute the expression (3.28) of  $\Delta\gamma - \Delta\phi$  and  $\Delta z = \Delta\lambda - \Delta\pi$  in equation (3.18) and use equations (3.29) and (3.19) to obtain

$$\left\{ \begin{array}{l} (Q + D_1)\Delta x - A^T\Delta y - C^T\Delta z = \bar{r}_Q \\ A\Delta x = \bar{r}_y \\ C\Delta x + D_2^{-1}\Delta z = \bar{r}_z, \end{array} \right. \quad (3.30)$$

where  $\bar{r}_Q = r_Q + (V^{-1}r_\gamma - W^{-1}r_\phi + V^{-1}\Gamma r_v - W^{-1}\Phi r_w)$ ,  $\bar{r}_y = r_y$  and  $\bar{r}_z = D_2^{-1}(T^{-1}r_\lambda - U^{-1}r_\pi + T^{-1}\Lambda r_t - U^{-1}\Pi r_u)$ .

Once the solution  $(\Delta x, \Delta y, \Delta z)$  of (3.30) is found, the unknowns  $\Delta t, \Delta u, \Delta v, \Delta w, \Delta \lambda, \Delta \pi, \Delta \gamma$ , and  $\Delta \phi$  can be computed by using only diagonal matrix-vector products and vector-vector additions as follows:

$$\left\{ \begin{array}{l} \Delta t = C\Delta x + r_t \\ \Delta u = -C\Delta x + r_u \\ \Delta v = \Delta x - r_v \\ \Delta w = -\Delta x - r_w \\ \Delta \lambda = T^{-1}(r_\lambda - \Lambda\Delta t) \\ \Delta \pi = U^{-1}(r_\pi - \Pi\Delta u) \\ \Delta \gamma = V^{-1}(r_\gamma - \Gamma\Delta v) \\ \Delta \phi = W^{-1}(r_\phi - \Phi\Delta w). \end{array} \right.$$

OOQP solves the symmetric system (3.31) obtained by performing the substitution  $(\Delta x, \Delta y, \Delta z) = (\Delta \tilde{x}, -\Delta \tilde{y}, -\Delta \tilde{z})$  in (3.30):

$$\begin{bmatrix} Q + D_1 & A^T & C^T \\ A & 0 & 0 \\ C & 0 & -D_2^{-1} \end{bmatrix} \begin{bmatrix} \Delta \tilde{x} \\ \Delta \tilde{y} \\ \Delta \tilde{z} \end{bmatrix} = \begin{bmatrix} \tilde{r}_x \\ \tilde{r}_y \\ \tilde{r}_z \end{bmatrix}. \quad (3.31)$$

### About OOQP-MA27

The linear system (3.31) is known in the interior-point community as the augmented system. OOQP's linear algebra layer for sparse general convex quadratic problems

solves the augmented system by using a sparse symmetric indefinite linear solver. The sparse, symmetric, indefinite linear systems are solved by using a Bunch-Parlett factorization for a matrix  $A$ . Such a factorization produces permutation matrices  $P$ , lower triangular matrix  $L$ , and the block diagonal matrix  $D$  with nonsingular  $1 \times 1$  and  $2 \times 2$  blocks that satisfy  $PAP^T = LDL^T$ . They are applied to the linear system (3.31).

The OOQP distribution contains interfaces to MA27 [32] and to the newer MA57 [31] linear solvers contained in Harwell Subroutine Library (HSL). We use MA27 because the MA57 solver is not available (free) for U.S. academics. The HSL code MA27 is a collection of FORTRAN routines for solving sparse systems of linear equations by a variant of Gauss elimination. The code and more documentation can be found at <http://hsl.rl.ac.uk/archive/hslarchive/packages/packages.html>.

### About our implementation OOQP-CHOL

In what follows we present the implementation of a new linear algebra layer in OOQP. As we mentioned, the OOQP's default linear algebra layer solves the indefinite symmetric system (3.31). One may take advantage of the particular structure of this system and perform further block elimination. A simple algebraic manipulation of the third equation in (3.31) reveals

$$\Delta\tilde{z} = D_2C\Delta\tilde{x} - D_2\tilde{r}_z. \quad (3.32)$$

By substituting (3.32) in the first equation of (3.31), we reduced the linear system to

$$\begin{cases} (Q + C^T D_2 C + D_1)\Delta\tilde{x} + A^T \Delta\tilde{y} &= \tilde{r}_x + C^T D_2 \tilde{r}_z \\ A\Delta x &= \tilde{r}_y. \end{cases} \quad (3.33)$$

We express  $\Delta\tilde{x}$  in terms of  $\Delta\tilde{y}$  by multiplying the first equation of (3.33) with the inverse of  $(Q + C^T D_2 C + D_1)$ . Notice that  $M_1 := (Q + C^T D_2 C + D_1)$  is always invertible and symmetric positive definite since  $Q$  and  $C^T D_2 C$  are symmetric positive semidefinite and  $D_1$  is a diagonal matrix with strictly positive diagonal entries. Using the new expression of  $\Delta\tilde{x}$ , we rewrite the second equation of (3.33) as

$$-AM_1^{-1}A^T\Delta\tilde{y} + AM_1^{-1}(\tilde{r}_x + C^T D_2 \tilde{r}_z) = \tilde{r}_y.$$

We denote  $M_2 = AM_1^{-1}A^T$  and obtain the following expression for  $\Delta\tilde{y}$

$$M_2\Delta\tilde{y} = AM_1^{-1}(\tilde{r}_x + C^T D_2 \tilde{r}_z) - \tilde{r}_y. \quad (3.34)$$

Once  $\Delta\tilde{y}$  is known,  $\Delta\tilde{x}$  can be obtained from

$$M_1\Delta\tilde{x} = -A^T\Delta\tilde{y} + \tilde{r}_x + C^T D_2 \tilde{r}_z. \quad (3.35)$$

As we mentioned,  $M_1$  is a symmetric positive definite matrix. This implies that  $M_2 = AM_1^{-1}A^T$  is also symmetric positive definite. Hence, the use of a Cholesky-based linear solver for solving (3.34) and (3.35) is an appropriate choice.

Performance, reliability, and availability were the main aspects we considered while choosing a sparse direct solver for symmetric positive definite linear systems of equations. We chose CHOLMOD 1.4 [26, 22, 25, 24] based on the evaluations from [44] and its ready availability.

In what follows, we describe how our new linear algebra layer within OOQP manages to solve (3.35) and (3.34) for  $\Delta\tilde{x}$  and  $\Delta\tilde{y}$ , respectively. The two interior-point methods implemented in OOQP, Mehrotra and Gondzio, use one matrix factorization per iteration and at least two backsolves. In other words, at least two linear systems having the same system matrix have to be solved at each iteration of the interior-point algorithm. Therefore, any factorization and other work that is not dependent

on the right-hand side must be performed once in the so-called factorization phase. Any other right-hand-side dependent operation is accomplished in the solve phase.

At any iteration of the interior-point method, the factorization phase consists of

- Cholesky factorization  $M_1 = L_1 L_1^T$ ;
- computing  $X = L_1^{-1} A^T$  by performing  $m_y$  backsolves:  $L_1 X = A^T$ ;
- computing  $M_2 = X^T X$ ;
- Cholesky factorization  $M_2 = L_2 L_2^T$ .

The solve phase computes  $\Delta \tilde{y}$ , then  $\Delta \tilde{x}$ , and finally  $\Delta \tilde{z}$  from (3.34), (3.35), and (3.32), respectively. It consists of

- computing  $r_1 := M_1^{-1}(\tilde{r}_x + C^T D_2 \tilde{r}_z)$  from (3.34) by performing a backsolve and a forward substitution:  $L_1 L_1^T r_1 = \tilde{r}_x + C^T D_2 \tilde{r}_z$ ;
- computing  $r_2 := A r_1 - \tilde{r}_y$ ;
- finding  $\Delta \tilde{y}$  from (3.34) by performing a backsolve and a forward substitution:  $L_2 L_2^T \Delta \tilde{y} = r_2$ ;
- computing the right-hand side  $r_3 := -A^T \Delta \tilde{y} + \tilde{r}_x + C^T D_2 \tilde{r}_z$  from (3.35)
- finding  $\Delta \tilde{x}$  from (3.35) by performing a backsolve and a forward substitution:  $L_1 L_1^T \Delta \tilde{x} = r_3$ ;
- finding  $\Delta \tilde{z}$  from (3.32).

Before describing the way CHOLMOD was integrated in our new linear algebra in OOQP, we give some of the main concepts related to the factorization of the (positive definite) matrices. The CHOLMOD factorization of a matrix is split in two

parts. The first is the so-called symbolic analysis and consists of computations that typically depend only on the nonzero pattern, not the numerical values. The main goal of this phase is to find a permutation of the matrix so that the amount of fill-in in the factors is minimized (or at least significantly decreased). This is also known as finding the fill-reducing ordering. The symbolic analysis phase also includes the symbolic factorization, which consists of finding the explicit representation of the nonzero pattern of the factor(s). The second part of the CHOLMOD factorization process is the numerical factorization based on a Cholesky-based algorithm. An important observation is that the symbolic analysis is usually much more expensive than the numerical factorization.

There is a key aspect in using CHOLMOD in the context of interior-point methods. Since the numerical factorization is based on the Cholesky algorithm, no numerical pivoting is needed to maintain numerical stability. This implies that the permutation found by the symbolic analysis does not have to be recomputed when factorizing a matrix with different numerical values but the same sparsity pattern.

On the other hand, the matrices  $M_1$  and  $M_2$  from (3.35) and (3.34) have a special property that turns out to be crucial in our discussion. During the iterations of the interior-point method, only the matrices  $D_1$  and  $D_2$  change. Since they are diagonal matrices (with positive diagonal entries), we obtain that the sparsity pattern of  $M_1$  remains the same during the interior-point iterations. Consequently,  $M_1^{-1}$  has the same pattern, which implies that the structure of  $M_2$  remains the same. Our code, like other interior-point Cholesky-based softwares (LIPSOL [86], PCx [23], HOPDM [42]), incorporates the above observations and performs the symbolic analysis phase only once at the first iteration of the interior-point method. Any other subsequent factorization need is backed up by a fast CHOLMOD numerical factorization.

CHOLMOD 1.4 offers the possibility to choose between up to nine fill-reducing ordering heuristics and two symbolic factorization methods. In our implementation, we let CHOLMOD decide on the best fill-reducing and symbolic factorization methods. Since both  $M_1$  and  $M_2$  have the special form  $AA^T$ , the usual choices of CHOLMOD were COLAMD for fill-reducing ordering and supernodal for symbolic factorization.

### 3.3.2 TRON

TRON is a trust region Newton method for bound constrained optimization problems. The algorithm uses a quadratic model function, projected searches during the subspace minimization phase and a preconditioned conjugate gradient method to determine the minor iterates. The limited memory preconditioner used is the incomplete Cholesky factorization described in [55].

The Cauchy step at iteration  $k$ ,  $s_k^C$  is of the form  $s_k(\alpha_k)$ , where the function  $s_k : \mathbb{R} \rightarrow \mathbb{R}^n$  is defined by

$$s_k(\alpha) = P[x_k - \alpha \nabla f(x_k)] - x_k.$$

Here  $P$  is the projection onto the (bound constrained) feasible set,  $x_k$  is the current iterate and  $f$  the objective function. An iterative scheme that is guaranteed to terminate in a finite number of steps is used to compute the Cauchy point by generating a sequence  $\{\alpha_k^{(l)}\}$  of trial values. The sequence can be either decreasing or increasing, based on the value of  $\alpha_k^{(0)}$ , where  $\alpha_k^{(0)}$  is set to 1 in the (main-loop/major) first iteration and  $\alpha_{k-1}$  otherwise.

Once the Cauchy point is obtained, a Newton step is sought subject to trust region constraints and with an active set choice determined by the one of the Cauchy points. If sufficient decrease is obtained compared to the one produced by the Cauchy

point, the step is accepted. The algorithm is superlinearly convergent for nonlinear objective functions.

### 3.3.3 BLMVM

BLMVM [18] is a projected gradient solver for nonlinear bound-constrained optimization problems. Like the unconstrained BFGS method, BLMVM creates a convex quadratic model function

$$m_k(d) = f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T B_k d,$$

where  $f(x)$  is the objective function,  $x_k$  is the current iterate, and the matrix  $B_k$  is updated at each iteration using correction pairs  $s_k$  and  $y_k$ . Unlike the unconstrained BFGS method, BLMVM defines the correction pairs  $s_k$  and  $y_k$  by

$$s_k = x_{k+1} - x_k, \quad y_k = T_\Omega \nabla f(x_{k+1}) - T_\Omega \nabla f(x_k).$$

Here  $T_\Omega$  is the projection operator, with the  $i$ th component of  $T_\Omega \nabla f(x)$  given by

$$(T_\Omega \nabla f(x))_i = \begin{cases} \partial_i f(x) & \text{if } x_i \in (l_i, u_i) \\ \min\{\partial_i f(x), 0\} & \text{if } x_i = l_i \\ \max\{\partial_i f(x), 0\} & \text{if } x_i = u_i \end{cases}$$

where  $\partial_i f(x)$  is the partial derivative of  $f$  with respect to the  $i$ th variable  $x_i$  and  $\Omega = \{x \mid l \leq x \leq u\}$  is the bound constrained feasible set.

To reduce the cost of storing the inverse Hessian approximation, BLMVM uses the limited memory BFGS method (L-BFGS). The algorithm uses a projected line search to enforce the bounds on the variables.

### 3.3.4 MOSEK

The MOSEK Optimization Software ([www.mosek.com](http://www.mosek.com)) is a collection of tools for solution of large-scale optimization problems. MOSEK provides specialized solvers for linear programming, mixed integer programming, and many types of nonlinear and convex optimization problems, such as convex quadratic problems, conic quadratic problems, and quadratically constrained problems. In particular, for solving convex quadratic problems subject to linear constraints, MOSEK employs an homogeneous interior-point algorithm for monotone complementarity problems [1]. This homogeneous model is able to solve the problem without any regularity assumption on the existence of optimal, feasible or strictly interior feasible points. If the problem has a solution, the algorithm generates a sequence that approaches feasibility and optimality simultaneously. If the problem is infeasible, it generates a sequence that converges to a certificate proving infeasibility. The algorithm can start at any positive point (feasible or infeasible) and converges in no more than  $O(\sqrt{n} \log(1/\epsilon))$  iterations, the best-known complexity for linear complementarity problems. In our experiments we used MOSEK 4.0 through the C optimizer application programming interface (API) to solve the primal form (3.10).

## 3.4 Numerical Results

In this section we present the details of applying the algorithms and solvers from Section 3.3.

### 3.4.1 Environment and Solver Configuration

We have used RedHat Enterprise Linux 5 to run our experiments. The jobs were submitted to a SUN Grid Engine 6.0u8 running on 10 dual-processor computers each having between 2 GB and 4 GB of physical memory. All the processors in the grid were Pentium 4 at 3.06 MHz with 512KB L2 cache. Our simulation code used one processor at a time.

The simulation code was written in C and C++ and compiled by using GNU C and C++ compilers. The optimization solvers and their requirements were also compiled by using GNU tools. The code optimization flag was set to  $-O3$  for all compilers.

We used all software packages with their default stopping criteria. Changing these parameters may affect the conclusions but also make the results difficult to report. The difficulty of projected gradient methods, such as those used by BLMVM and TRON, to obtain a solution for very stringent tolerance is well documented, and we do not investigate that here. Our goal is to provide a useful benchmark for engineering applications, which in many cases accept errors of the order of those in the stopping criteria of BLMVM and TRON. We thus believe that the fairest comparison of usefulness of these packages is for their default settings.

We used for both OOQP-MA27 and OOQP-CHOL the default stopping criteria, which consist of relative gap  $\mu < 10^{-8}$  and relative norm of the residual less than  $10^{-8}$ . The relative norm of the residual is the ratio between the norm of the residual and the absolute value of the largest magnitude element in the problem's data. We also run MOSEK with the default stopping parameters; namely, primal and dual feasibility tolerance and relative gap less than  $10^{-8}$ . TRON and BLMVM consider a problem solved when the norm of the gradient falls under  $10^{-4}$  and  $10^{-3}$ , respectively.

### 3.4.2 Examples Generation

In all of our experiments, in a unitless representation, the dimensions of the vat are 60 for the radius of the cylinder and 20 for the small radius of the truncated cone (see Figure 3.1). The height of the cylinder and the truncated cone are 80 and 40, respectively. The radius of each pebble is set to 1.

For the simulation experiments the pebbles are initially randomly arranged in horizontal planes. On each horizontal plane the pebbles are distributed in several inner circles. For the optimization experiments we also place pebbles on the bottom of the vat. The number of such bodies is approximately one-fourth the number of the suspended ones. Since we are interested in the situation when some of the falling balls are still in the air, while the others are interacting with the walls of the vat and with the pebbles from the bottom of the vat, the optimization problem is chosen after several seconds of simulation.

For any configuration, we solve the problem (3.13), which is set up as described in Section 3.2.1, for given  $q^l, v^l$  and time step  $h$ . This produces the multipliers  $\lambda^{(l+1)}$ . We replace them in (3.9), to obtain  $v^{(l+1)}$ , the solution of (3.10). After this, the new position variables are obtained from  $q^{(l+1)} = q^{(l)} + h\Gamma(q^{(l)})v^{(l+1)}$ .

### 3.4.3 Total Kinetic Energy Results

It is well known that granular flow simulation is chaotic [60]. This means that the tiniest difference in position of the particles at a given time step is amplified exponentially in time. Therefore, comparing the outcome of the various solvers for individual particles is essentially hopeless beyond extremely small and few time steps. In order to compare the prediction of the various solvers, it may be more illuminating to use aggregate quantities. To that end, a meaningful quantity is the total kinetic energy.

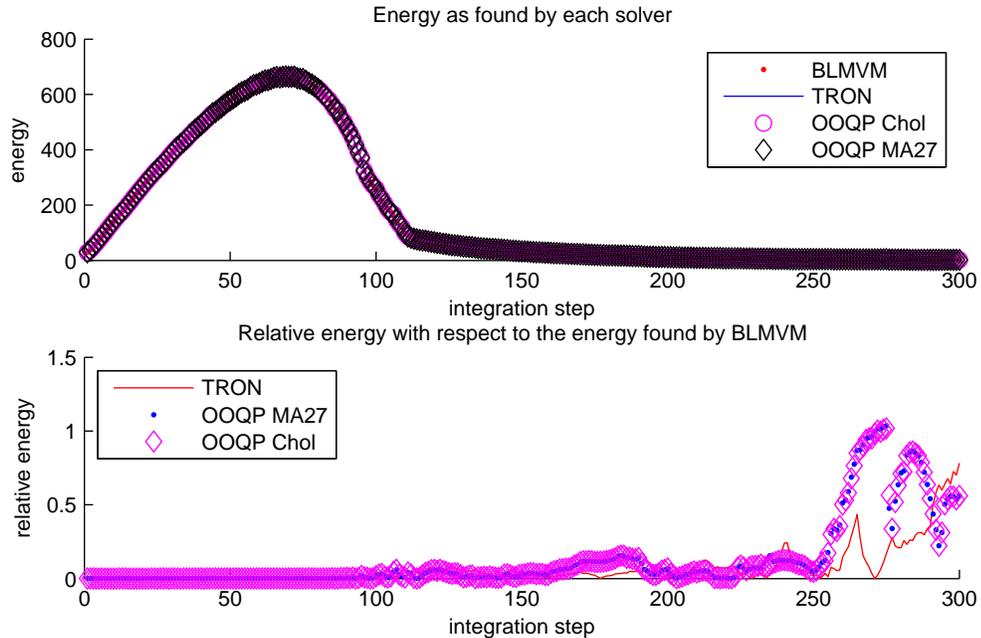


Figure 3.3: Energy dependence on time. Even though the system is chaotic, simulations report approximately the same total kinetic energy (top figure). Larger differences (smaller than 1.5%) that occur around the equilibrium have a numerical explanation (small denominator) rather than a physical meaning (bottom figure).

The first plot of Figure 3.3 shows how the kinetic energy of a system consisting of 800 pebbles changes in time. The definition of the kinetic energy is  $E(t) = \frac{1}{2}v(t)^T M v(t)$ . Its value was found by simulating the same configuration with the four solvers. In the second plot we represent the relative energy found by OOQP-MA27, OOQP-CHOL, and MOSEK with respect to the energy found by BLMVM. The relative energy is  $E_{rel} = \frac{|E_s - E_b|}{E_b}$ , where  $E_b$  and  $E_s$  are the energies found by making use of BLMVM and one of the remaining solvers, respectively.

Even though the system is chaotic, simulations report approximately the same total kinetic energy (top figure). We note that the relative error in energy is insignifi-

cant, *in physical interpretation terms*, except after a large amount of simulation time. We also note from the bottom figure that this error ( $< 1.5\%$ ) occurs only at very small value of the kinetic energy (essentially, around the time the pebbles have stopped) and is due primarily to a small denominator rather than a physical cause. In addition, as can be seen from Tables 3.1, 3.2, and 3.3, some of the errors are due to the fact that BLMVM and TRON are iterative solvers that are stopped with larger error in both primal and dual than the interior point solvers.

### 3.4.4 Performance Results

The tests involving MOSEK are performed on a Windows XP SP2 machine with 1.5GB memory running a P4 2.8 MHz processor with 512 KB L2 cache. On Windows, MOSEK was statically linked using Microsoft Visual Studio 7.1. We were not able to test MOSEK on Linux because only the Windows license was available to us. We are aware that the use of different hardwares and operating systems leads to different execution times. To have an idea about how big this difference is, we ran multiple simulations on both Windows and Linux computers. The simulations with OOQP-CHOL and BLMVM as the optimization solvers revealed that the execution is 30-35 percent slower on the Windows machine. One should keep in mind this difference when comparing execution times obtained by MOSEK on Windows with the execution times of the other four solvers on Linux.

We present two types of experiments comparing the solver performance. In the first experiment, the simulation test, we compare the solver performance for all the QPs encountered in the simulation, for different total numbers of pebbles. For such comparisons, QPs with the same number of dual variables are not necessarily the same. Therefore, different QPs with the same number of dual variables may be

Solver	Pri/Dual	Primal Infeas	Int. steps	Total Time	Avg time
BLMVM	Dual	1.482e-04	324	9738.880	30.058
OOQP MA27	Both	0	370	59693.310	161.330
OOQP Chol	Both	0	371	9351.450	25.206
TRON	Dual	1.070e-02	487	282763.981	580.624
Mosek <sup>1</sup>	Both	0	407	1120797.148	2753.801

Table 3.1: Performance of QP solvers for 800 pebbles and  $h = 0.05$ 

solved with different performance parameters, and the comparisons must be carried out only in terms of trends. This can be seen in the scatter plots of Figures 3.4 and 3.5.

A second experiment, the optimization test, progresses the simulation with one solver, OOQP-CHOL, up to a time where the QP to be solved is sufficiently large. At that point, the same QP is solved by all software packages, and the performance results are compared on the same problem.

### Simulation test

Tables 3.1, 3.2, and 3.3 show the performance of each optimization solver in running simulations of 800, 1600, and 3200 pebbles, respectively. The second column indicates whether the primal (3.10) or dual (3.13) form was solved. The third column, *Primal Infeas* lists the primal infeasibility at the last integration step. The number of integration steps needed to run the simulation is shown in the fourth column. The last two columns represent the total time in seconds needed for a simulation and the

<sup>1</sup>Reported data is obtained on Windows.

Solver	Primal/Dual	Primal Infeas	Int. steps	Total Time	Avg time
BLMVM	Dual	6.235e-05	394	73147.070	185.652
OOQP MA27	Both	0	319	345773.404	1083.929
OOQP Chol	Both	0	310	38097.440	122.894

Table 3.2: Performance of QP solvers for 1600 pebbles and  $h = 0.05$ 

Solver	Primal/Dual	Primal Infeas	Int. steps	Total Time	Avg time
BLMVM	Dual	4.946e-05	284	221411.620	779.618
OOQP Chol	Both	0	296	175534.870	593.023

Table 3.3: Performance of QP solvers for 3200 pebbles and  $h = 0.05$ 

average time in seconds per integration step. The simulations were stopped when the kinetic energy fell under a specific value: 0.2 for 800 pebbles, 0.8 for 1600, and 2.5 for 3200.

Both MOSEK and TRON crash while running the simulation involving 1600 pebbles. When solving the optimization problem from the first integration step MOSEK freezes in the preprocessing phase for several hours and then crashes. The memory usage before the abnormal termination was close to the maximum available. We believe that the lack of memory caused the failure of a memory allocation routine and consequently, MOSEK's crash. Although TRON was able to run the first several integration steps, as soon as the pebbles started to interact with the walls, and the size of the dual increased, it crashed. The source of the crash was a memory allocation failure in the FORTRAN 77 code. An important observation is that there was enough physical memory to satisfy the allocation request. Hence the failure is probably caused by FORTRAN 77 memory management routines. We present the

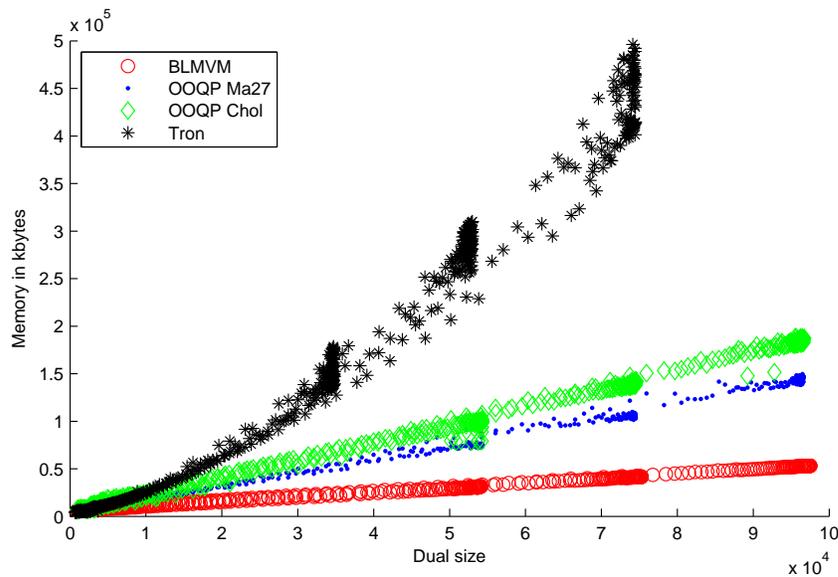


Figure 3.4: Memory usage dependence on the number of constraints. Shown are simulations of 600, 800, 1000, and 1200 pebbles.

memory use for BLMVM, OOQP, and TRON for this test in Figure 3.4.

TRON and MOSEK were not used for the experiment involving 3200 pebbles. The same simulation with OOQP MA27 was stopped because of the huge amount of time needed to solve the optimization problems (more than 20 times the time needed by OOQP-Chol for the same integration step).

### Optimization test

In the simulation test the solvers may solve different problems at each step since the system trajectories may be different due to accumulation of the numerical error. In this test, we compare the performance of all solvers for the same QP problem.

As described in Section 3.4.2, the optimization problems are chosen from a sim-

---

<sup>2</sup>Reported data is obtained on Windows.

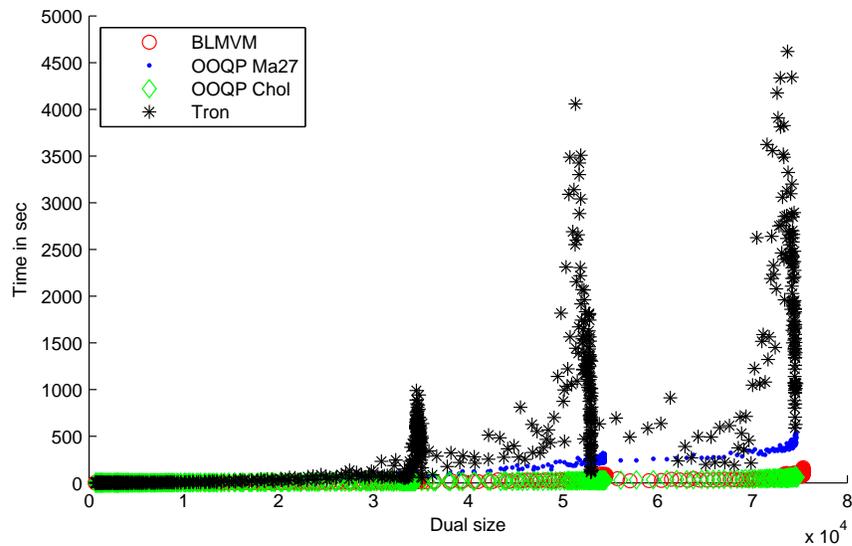


Figure 3.5: Execution time dependence on the number of constraints. Shown are simulations of 600, 800, and 1000 pebbles.

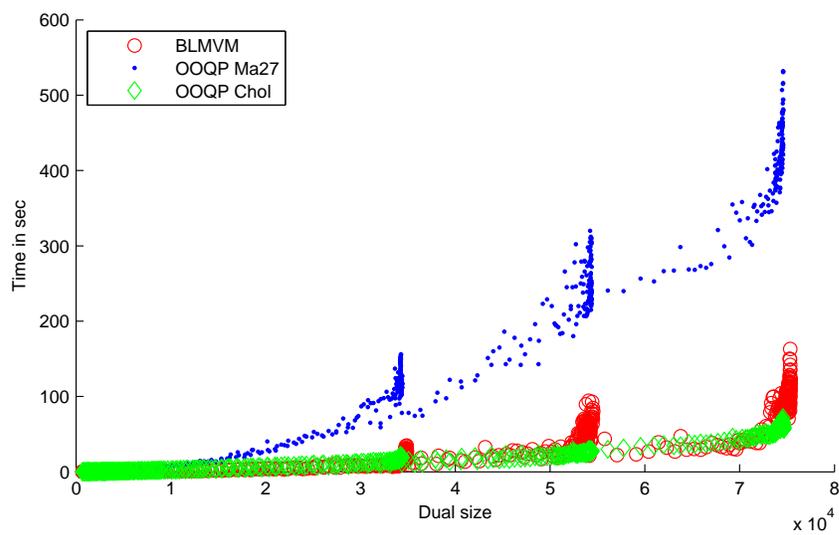


Figure 3.6: Execution time dependence on the number of constraints. Shown are simulations of 1200, 1400 and 1600 pebbles.

Solver	Primal/Dual	Primal Size	Dual Size	No. Iter	Average Time	Total Time
BLMVM	Dual	6000	62826	2501	0.127	318.016
OOQP Ma27	Primal/Dual	6000	62826	33	21.120	696.984
OOQP Chol	Prima/Dual	6000	62826	33	3.115	102.812
Mosek <sup>2</sup>	Primal/Dual	6000	62826	24	465.362	11168.688
TRON	Dual	6000	62826			CRASH

Table 3.4: The performance of QP solvers in solving an optimization problem from the simulation of 1000 pebbles with  $h = 0.01$ .

Solver	Primal/Dual	Primal Size	Dual Size	No. Iter	Average Time	Total Time
BLMVM	Dual	6000	62760	1729	0.128	221.640
OOQP Ma27	Primal/Dual	6000	62760	31	21.351	661.881
OOQP Chol	Prima/Dual	6000	62760	31	3.016	93.500
Mosek*	Primal/Dual	6000	62760	24	456.534	10956.832
TRON	Dual	6000	62760			CRASH

Table 3.5: The performance of QP solvers in solving an optimization problem from the simulation of 1000 pebbles with  $h = 0.05$ .

ulation of 1000 pebbles (800 staying on the bottom of the vat and 200 falling) after 3.3 seconds. The integration was done by using the timestep  $h = 0.01$  for Table 3.4 and  $h = 0.05$  for Table 3.5. Tables 3.4 and 3.5 list the solver name, the formulation solved, the number of unknowns in primal and dual, the number of iterations needed by each solver to solve the optimization problem, the average time per iteration, and the total time taken to solve the problem.

### 3.4.5 Discussion of the Results

We conclude that the ranking from most to least performing of the five solvers is OOQP-Chol (our linear algebra interface and implementation), BLMVM, OOQP-MA27, TRON, and MOSEK. This conclusion is sustained for both the simulation test (see results in Tables 3.1, 3.2 and 3.3 and in Figures 3.5 and 3.6), and the optimization test (see results in Tables 3.4 and 3.5).

The tabulated results show that this ordering holds on average, whereas the figures show that these results hold even when accounting for the spread of performance criteria for the same dual size. In addition, we also see from Figure 3.4 that both interior-point algorithms need more memory only by a factor of between 2 and 3 compared to BLMVM, which, as a limited-memory method, is quite memory-use conscious. From the memory results, it is also interesting to extrapolate what size of a problem will be held by a desktop. If the trends in Figure 3.4 hold, then a 4 GB architecture can hold a 150,000-pebble configuration, whereas a 32 GB architecture can hold a 600,000-pebble configuration.

We note that our OOQP-Chol implementation, using open source tools, is consistently seven times faster (and sometimes more than twenty times faster) compared to the OOQP-MA27 implementation. We also note that the time to solution performance of BLMVM and both OOQP implementations behaves fairly close to linear with the size of the problem. So both algorithms scale reasonably for this problem. In addition, our kinetic energy monitoring reveals that all solvers give comparable results.

Several caveats should accompany our conclusions. The first is that we do not require BLMVM to solve the problem to the same precision as for the interior-point solvers. Nonetheless, we believe that its results are useful, for reasons described in

Subsection 3.4.1. The second is that, for license issues, we were not able to run MOSEK on the same architecture as the other solvers. Given our experience with the performance discount between Windows and Linux, we believe that the conclusions would not change when running MOSEK on Linux, for reasons described in Section 3.4.4. We also note that the class of problems solved here, while of wide engineering interest, is limited insofar type of QPs encountered. For other QP types, it is conceivable that the performance ranking will change.

### 3.5 Conclusions and Future Work

We investigated the performance of four software packages for the resolution of quadratic programming problems with bound constraints that appear in the resolution of rigid multi-body dynamics with contact and friction. These packages are TRON [55], BLMVM [18], MOSEK [1], and OOQP [41]. OOQP is investigated both with the default MA27 linear algebra and with our new implementation using Cholesky factorizations by means of the CHOLMOD package. We call the first instance OOQP-MA27 and the second OOQP-Chol.

We conclude that, for such problems, our OOQP-Chol implementation is the fastest of all the tested packages. It consistently uses only about three times more memory than BLMVM, while achieving far higher precision levels. Its behavior with the size of the problem is predictable, as can be seen from Figures 3.4, 3.5, and 3.6.

An important further research question is whether this performance holds for a parallel implementation. We note that a multithreaded version of CHOLMOD exists (see <http://www.cise.ufl.edu/research/sparse/cholmod/>) but does not currently exhibit good performance because of BLAS issues; these are expected to be fixed in

the near future. The good parallel speedup of BLMVM, the closest competitor in terms of execution speed, is well documented [18].

Another important direction is to work directly with the disk constraint on the tangential force. This results in conic constraints on the contact force, which, in turn, leads to a quadratic program with conic constraints. It was recently shown that this formulation leads to a conic complementarity problem [78, 14] and can be solved using a splitting scheme, of the Gauss-Seidel or Jacobi type [14]. At the moment, however, the set of codes available to us that support conic constraints includes only MOSEK, so we are not yet in the position to profile such codes for rigid multi-body applications.

# Bibliography

- [1] E. Andersen and Y. Ye. On a homogeneous algorithm for the monotone complementarity problem. *Mathematical Programming*, 84(2):375–399, 1999.
- [2] E. D. Andersen, J. Gondzio, C. Mészáros, and X. Xu. Implementation of interior point methods for large scale linear programming. In *Interior Point Methods in Mathematical Programming*, pages 189–252. Kluwer Academic Publishers, 1996.
- [3] E.D. Andersen and Y. Ye. A computational study of the homogeneous algorithm for large-scale convex optimization. *Comput. Optim. Appl.*, 10(3):243–280, 1998.
- [4] E.D. Andersen and Y. Ye. On a homogeneous algorithm for the monotone complementarity problem. *Math. Program.*, 84(2, Ser. A):375–399, 1999.
- [5] M. Anitescu. A fixed time-step approach for multibody dynamics with contact and friction. *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4:3725 – 3731, 2003.
- [6] M. Anitescu, J.F. Cremer, and F.A. Potra. On the existence of solutions to complementarity formulations of contact problems with friction. *Complementarity and Variational Problems: State of the Art*, 1997.

- [7] M. Anitescu and G.D. Hart. A constraint-stabilized time-stepping approach for rigid multibody dynamics with joints, contact and friction. *Int. J. Numer. Methods Eng.*, 60(14):2335–2371, 2004.
- [8] M. Anitescu and F. A. Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, 14:231–247, 1997.
- [9] Mihai Anitescu. Optimization-based simulation of nonsmooth dynamics. *Mathematical Programming*, 105:113–143, 2006.
- [10] Mihai Anitescu, James F. Cremer, and Florian A. Potra. Formulating 3D contact dynamics problems. *Mechanics of Structures and Machines*, 24(4):405–437, 1996.
- [11] Mihai Anitescu and Gary D. Hart. Solving nonconvex problems of multibody dynamics with joints, contact and small friction by sequential convex relaxation. *Mechanics Based Design of Machines and Structures*, 31(3):335–356, 2003.
- [12] Mihai Anitescu and Gary D. Hart. A fixed-point iteration approach for multibody dynamics with contact and small friction. *Mathematical Programming*, 101:3–32, 2004.
- [13] Mihai Anitescu, Florian A. Potra, and David Stewart. Time-stepping for three-dimensional rigid-body dynamics. *Computer Methods in Applied Mechanics and Engineering*, 177:183–197, 1999.
- [14] Mihai Anitescu and Alessandro Tasora. An iterative approach for cone complementarity problems for nonsmooth dynamics. Preprint ANL/MCS-P1413-0507, Argonne National Laboratory, Argonne, Illinois, 2007.

- [15] K. Anstreicher. On interior algorithms for linear programming with no regularity assumptions. *Operations Research Letters*, 11(7):209–212, 1992.
- [16] David Baraff. Issues in computing contact forces for non-penetrating rigid bodies. *Algorithmica*, 10:292–352, 1993.
- [17] Guido Bartels, Tams Unger, Dirk Kadau, Dietrich E. Wolf, and Jnos Kertesz. The effect of contact torques on porosity of cohesive powders. *Granular Matter*, 7(2-3):139–143, 2005.
- [18] Steven J. Benson and Jorge Moré. A limited-memory variable-metric algorithm for bound-constrained minimization. Technical Report ANL/MCS-P909-0901, Mathematics and Computer Science Division, Argonne National Laboratory, 2001.
- [19] Stephen C. Billups and Katta G. Murty. Complementarity problems. *J. Comput. Appl. Math*, 124:303–318, 2000.
- [20] A. Boeing and T. Bräunl. Evaluation of real-time physics simulation systems. *Proceedings of the 5th international conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia*, pages 281–288, 2007.
- [21] A. Chatterjee and A. Ruina. A new algebraic rigid-body collision law based on impulse space considerations. *ASME, Transactions, Journal of Applied Mechanics*, 65(4):939–951, 1998.
- [22] Yanqing Chen, Timothy A. Davis, William W. Hager, and Sivasankaran Rajamanickam. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software*, 35(3):1–14, 2008.

- [23] J. Czyzyk, S. Mehrotra, and S. J. Wright. Pcx user guide. Technical Report OTC 96/01, Optimization Technology Center, Argonne National Laboratory and Northwestern University, October 1996.
- [24] T. A. Davis and W. W. Hager. Modifying a sparse cholesky factorization. *SIAM Journal on Matrix Analysis and Applications*, 20(3):606–627, 1999.
- [25] T A. Davis and W. W. Hager. Row modifications of a sparse cholesky factorization. *SIAM Journal on Matrix Analysis and Applications*, 26(3):621–639, 2005.
- [26] Timothy A. Davis and William W. Hager. Dynamic supernodes in sparse cholesky update/downdate and triangular solves. *ACM Transactions on Mathematical Software*, 35(4):1–23, 2009.
- [27] E. de Klerk, C. Roos, and T. Terlaky. Initialization in semidefinite programming via a self-dual skew-symmetric embedding. Report of the Faculty of Technical Mathematics and Informatics 96-10, Delft University of Technology, January 1996.
- [28] S. Dirkse and M. Ferris. The PATH solver: A non-monotone stabilization scheme for mixed complementarity problems. *Optimization Methods and Software*, 5:123–156, 1995.
- [29] Bruce R. Donald and Dinesh K. Pai. On the motion of compliantly connected rigid bodies in contact: a system for analyzing designs for assembly. In *Proceedings of the Conf. on Robotics and Automation*, pages 1756–1762. IEEE, 1990.
- [30] I. S. Duff and J. K. Reid. The design of ma48: a code for the direct solution of sparse unsymmetric linear systems of equations. *ACM Transactions on Mathematical Software*, 22(2):187–226, 1996.

- [31] Iain S. Duff. MA57—a code for the solution of sparse symmetric definite and indefinite systems. *ACM Trans. Math. Softw.*, 30(2):118–144, 2004.
- [32] JS DUFF and JK REID. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM transactions on mathematical software*, 9(3):302–325, 1983.
- [33] K. Erleben. Stable, Robust, and Versatile Multibody Dynamics Animation. *Ph.D. thesis, University of Copenhagen, Copenhagen*, 2004.
- [34] F. Facchinei and J. S. Pang. *Finite-dimensional variational inequalities and complementarity problems*, volume 1 of *Springer Series in Operations Research*. Springer-Verlag, New York, 2003.
- [35] F.Camborde, C.Mariotti, and F.V.Donze. Numerical study of rock and concrete behaviour by discrete element modeling. *Computers and Geotechnics*, 27:225–247, 2000.
- [36] Roy Featherstone. *Robot Dynamics Algorithms*. Kluwer Academic Publishers, Boston, 1987.
- [37] M.C. Ferris and J.S. Pang. Engineering and economic applications of complementarity problems. *SIAM Review*, 39(4):p669 –, 19971201.
- [38] Michael Ferris and Todd Munson. Interfaces to PATH 3.0: Design, implementation and usage. *Computational Optimization and Applications*, 12:207–227, 1999.
- [39] Jacques Gauvin. A necessary and sufficient regularity condition to have bounded multipliers in nonconvex programming. *Mathematical Programming*, 12:136–138, 1977.

- [40] E. M. Gertz and S. J. Wright. Object-oriented software for quadratic programming. *ACM Transactions on Mathematical Software*, 29(1):58–81, 2003.
- [41] E. Michael Gertz and Stephen J. Wright. Object-oriented software for quadratic programming. *ACM Transactions on Mathematical Software*, 29(1):58–81, March 2003.
- [42] Jacek Gondzio. Hopdm (version 2.12) - a fast lp solver based on a primal-dual interior point method. *European Journal of Operational Research*, 85:221–225, 1995.
- [43] Hans David Gougar. *Advanced core design and fuel management for pebble-bed reactors*. Ph.D Thesis, Department of Nuclear Engineering, Penn State University, 2004.
- [44] N. I. M. Gould, J. A. Scott, and Y. Hu. A numerical evaluation of sparse direct solvers for the solution of large sparse symmetric linear systems of equations. *ACM Transactions on Mathematical Software*, 33(2):10, 2007.
- [45] David Goulding, Jean-Pierre Hansen, and Simone Melchionna. Size selectivity of narrow pores. *Physical Review Letters*, 85(5):1132–1135, 2000.
- [46] M. S. Gowda. Reducing a monotone horizontal LCP to an LCP. *Appl. Math. Lett.*, 8(1):97–100, 1995.
- [47] O. Güler. Generalized linear complementarity problems. *Math. Oper. Res.*, 20(2):441–448, 1995.
- [48] S. Hasegawa and M. Sato. Real-time rigid body simulation for haptic interactions based on contact volume of polygonal objects. *Computer Graphics Forum*, 23(3):529–538, 2004.

- [49] E. J. Haug. *Computer-Aided Kinematics and Dynamics of Mechanical Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1989.
- [50] Dirk Helbing, I.J. Farkas, and T. Vicsek. Simulating dynamical features of escape panic. *Nature*, 407:487–490, 2000.
- [51] W.L. Jorgensen, J. Chandrasekhar, J.D. Madura, R.W. Impey, and M.L. Klein. Comparison of simple potential functions for simulating liquid water. *Journal of Chemical Physics*, 79(2):926–935, 1983.
- [52] D. Kadau, G. Bartels, L. Brendel, and D.E. Wolf. Pore stabilization in cohesive granular systems. *Phase Transitions: A Multinational Journal*, 76(4-5):315–331, 2003.
- [53] N. K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.
- [54] C.-J. Lin and J. J. Moré. Newton’s method for large bound-constrained optimization problems. *SIAM Journal on Optimization*, 9(4):1100–1127, 1999.
- [55] C.J. Lin and J.J. Moré. Incomplete Cholesky Factorizations with Limited Memory. *SIAM Journal on Scientific Computing*, 21(1):24–45, 1999.
- [56] Z. Q. Luo, J. F. Sturm, and S. Zhang. Conic convex programming and self-dual embedding. *Optimization Methods & Software*, 14:169 – 218, 1998.
- [57] I.J. Lustig, R. E. Marsten, and D. F. Shanno. On implementing mehrotra’s predictor–corrector interior-point method for linear programming. *SIAM Journal on Optimization*, 2(3):435–449, 1992.

- [58] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.
- [59] V.J. Milenkovic and H. Schmidl. Optimization-based animation. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 37–46, 2001.
- [60] R. Moeckel. Chaotic dynamics near triple collision. *Archive for Rational Mechanics and Analysis*, 107(1):37–69, 1989.
- [61] R. D. C. Monteiro and Jong-Shi Pang. Properties of an interior-point mapping for mixed complementarity problems. *Mathematics of Operations Research*, 21(3):629–654, 1996.
- [62] R. D. C. Monteiro and Jong-Shi Pang. On two interior-point mappings for nonlinear semidefinite complementarity problems. *Mathematics of Operations Research*, 23(1):39–60, 1998.
- [63] Y. Nesterov, M.J. Todd, and Y. Ye. Infeasible-start primal-dual methods and infeasibility detectors for nonlinear programming problems. *Mathematical Programming*, 84(2):p227 –, 19990201.
- [64] Jong-Shi Pang, Vijay Kumar, and Peng Song. Convergence of time-stepping method for initial and boundary-value frictional compliant contact problems. *SIAM J. Numer. Anal.*, 43(5):2200–2226, 2005.
- [65] Jong-Shi Pang and Jeffrey C. Trinkle. Complementarity formulations and existence of solutions of dynamic multi-rigid-body contact problems with coulomb friction. *Math. Program.*, 73(2):199–226, 1996.

- [66] C. Petra, B. Gavrea, M. Anitescu, and F. A. Potra. A computational study of the use of an optimization-based method for simulating large multibody systems. *Optimization Methods and Software*, to appear, 2009.
- [67] F. A. Potra and R. Sheng. On homogeneous interior-point algorithms for semidefinite programming. *Optimization Methods and Software*, 9:161–184, 1998.
- [68] Florian A. Potra, Mihai Anitescu, Bogdan Gavrea, and Jeff Trinkle. A linearly implicit trapezoidal method for integrating stiff multibody dynamics with contact and friction. *International Journal for Numerical Methods in Engineering*, 66(7):1079–1124, 2006.
- [69] Farhang Radjai, Michel Jeanand Jean-Jacques Moreau, and S. Roux. Force distributions in dense two-dimensional granular systems. *Physical Review Letters*, 77(2):274–277, 1996.
- [70] M. Salahi, J. Peng, and T. Terlaky. On mehrotra-type predictor-corrector algorithms. *SIAM J. on Optimization*, 18(4):1377–1397, 2007.
- [71] M. Salahi and T. Terlaky. Mehrotra-type predictor-corrector algorithm revisited. *Optimization Methods Software*, 23(2):259–273, 2008.
- [72] Marco Saraniti, Shela Aboud, and Robert Eisenberg. The simulation of ionic charge transport in biological ion channels: an introduction to numerical methods. *Reviews in Computational Chemistry*, 22:229–293, 2006.
- [73] Peng Song, P. Kraus, Vijay Kumar, and P. Dupont. Analysis of rigid-body dynamic models for simulation of systems with frictional contacts. *Journal of Applied Mechanics*, 68(1):118–128, 2001.

- [74] Peng Song, Jong-Shi Pang, and Vijay Kumar. A semi-implicit time-stepping model for frictional compliant contact problems. *International Journal of Numerical Methods in Engineering*, 60(13):267–279, 2004.
- [75] D. E. Stewart and J.C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International J. Numer. Methods Engineering*, 39(15):281–287, 1996.
- [76] David E. Stewart. Rigid-body dynamics with friction and impact. *SIAM Review*, 42(1):3–39, 2000.
- [77] R. Sznajder and M. S. Gowda. Generalizations of P0- and P-properties; extended vertical and horizontal linear complementarity problems. *Linear Algebra and its Applications*, 223-224:695 – 715, 1995. Honoring Miroslav Fiedler and Vlastimil Ptak.
- [78] Alessandro Tasora and Mihai Anitescu. *ECCOMAS Thematic Conference in Multibody Dynamics*, chapter A Fast NCP Solver for Large Rigid-Body Problems with Contacts, Friction, and Joints. Springer Verlag, Berlin, 2008. To appear.
- [79] Alessandro Tasora, E. Manconi, and M. Silvestri. Un nuovo metodo del semplice per il problema di complementarit lineare mista in sistemi multibody con vincoli unilateri. In *Proceedings of AIMETA 05*, Firenze, Italy, 2005.
- [80] Jeffrey Trinkle, Jong-Shi Pang, Sandra Sudarsky, and Grace Lo. On dynamic multi-rigid-body contact problems with coulomb friction. *Zeithschrift fur Angewandte Mathematik und Mechanik*, 77:267–279, 1997.

- [81] A. Wallqvist and R.D. Mountain. Molecular models of water: Derivation and description. In *Reviews in Computational Chemistry*, volume 13, pages 183–247. John Wiley and Sons, 1999.
- [82] Y. Ye. On homogeneous and self-dual algorithms for LCP. *Math. Programming*, 76(1, Ser. B):211–221, 1997. Interior point methods in theory and practice (Iowa City, IA, 1994).
- [83] Y. Ye, M. J. Todd, and S. Mizuno. An  $O(\sqrt{n}L)$ -iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research*, 19(1):53–67, 1994.
- [84] A. Yoshise. Interior point trajectories and a homogeneous model for nonlinear complementarity problems over symmetric cones. *SIAM Journal on Optimization*, 17(4):1129 – 1153, 2006.
- [85] D. Zhang and Y. Zhang. A Mehrotra-type predictor-corrector algorithm with polynomiality and  $Q$ -subquadratic convergence. *Ann. Oper. Res.*, 62:131–150, 1996. Interior point methods in mathematical programming.
- [86] Y. Zhang. Solving large-scale linear programs by interior-point methods under the matlab environment. Technical Report TR96-01, University of Maryland Baltimore County, February 1996.
- [87] Y. Zhang and D. Zhang. On the polynomiality of the Mehrotra-type predictor-corrector interior point algorithms. *Mathematical Programming*, 68(3):303–318, 1995.

