On Inexact Newton Directions in Interior Point Methods for Linear Optimization

Ghussoun Al-Jeiroudi

Doctor of Philosophy
University of Edinburgh

2009

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(Ghussoun Al-Jeiroudi)

Abstract

In each iteration of the interior point method (IPM) at least one linear system has to be solved. The main computational effort of IPMs consists in the computation of these linear systems. Solving the corresponding linear systems with a direct method becomes very expensive for large scale problems.

In this thesis, we have been concerned with using an iterative method for solving the reduced KKT systems arising in IPMs for linear programming. The augmented system form of this linear system has a number of advantages, notably a higher degree of sparsity than the normal equations form. We design a block triangular preconditioner for this system which is constructed by using a nonsingular basis matrix identified from an estimate of the optimal partition in the linear program. We use the preconditioned conjugate gradients (PCG) method to solve the augmented system. Although the augmented system is indefinite, short recurrence iterative methods such as PCG can be applied to indefinite system in certain situations. This approach has been implemented within the HOPDM interior point solver.

The KKT system is solved approximately. Therefore, it becomes necessary to study the convergence of IPM for this inexact case. We present the convergence analysis of the inexact infeasible path-following algorithm, prove the global convergence of this method and provide complexity analysis.

Acknowledgements

I would like to express my sincere thanks to Professor Jacek Gondzio. I can honestly say I have been extremely fortunate to have him as my supervisor. He has been my encyclopaedia of research knowledge. I would like to thank him for giving me this opportunity and having belief in me.

I would like to thank Dr. Julian Hall for giving me the opportunity to work in programming with him. I have learnt a lot from him. I have been honoured to work with such an enlightened individual.

I would also like to thank all who have given me motivation and helped me through out my Ph.D. Thanks to my friends who have shared with me hard moments as well as beautiful moments. I would like to thank all my friends who have introduced me to many different cultures and have contributed to an experience that I will never forget.

The study could not have taken place without a sponsor. I would like to acknowledge the University of Damascus for sponsoring me throughout my Ph.D.

I would also like to take this opportunity and thank my family, for their love and support in all my pursuits in life.

Contents

Chapter 1

Introduction

Interior point methods constitute the core of many popular solvers for linear and nonlinear optimization. In linear programming however, that was not always the case due to the total dominance of the simplex method. The simplex method was invented by Dantzig in 1947. It is an iterative technique, where the iterates move from vertex to vertex until an optimal vertex is found. The simplex method may visit every vertex of the feasible polyhedron. That makes the complexity result of this method poor: the worst-case complexity of the simplex method is exponential in the problem dimension. Accordingly, there was great interest in finding a method with polynomial complexity. In 1984 Karmarkar presented a new polynomial-time algorithm for linear programming. He claimed to be able to solve linear programs up to 50 times faster than the simplex method. That was the start of the "interior point revolution" [48], which like many other revolutions, includes old ideas that are rediscovered or seen in a different light, along with genuinely new ones. See [3, 27, 76].

An interior point method (IPM for short) is a powerful tool to solve linear, quadratic and nonlinear programming problems. In this thesis we are concerned with the use of primal-dual interior point methods to solve large-scale linear programming problems. A primal-dual method is applied to the primal-dual formulation of linear program

$$\begin{array}{llll@{\qquad}llll} & \text{Primal} & & & & \text{Dual} & \\ (P) & \min & c^T x & & (D) & \max & b^T y \\ & \text{s.t.} & Ax = b, & & & \text{s.t.} & A^T y + s = c, \\ & & x \geq 0; & & & & y \text{ free}, \;\; s \geq 0, \end{array}$$

where $A \in \mathcal{R}^{m\times n}$, $x, s, c \in \mathcal{R}^n$ and $y, b \in \mathcal{R}^m$. x, y and s are primal, dual and slack variables respectively. We assume that $m \leq n$ and the matrix A has full row rank. Primal-dual techniques are usually faster and more reliable than pure primal or pure dual approaches [3, 38, 77]. In order to solve problem (P), we need to find the solution of the Karush-Kuhn-Tucker (KKT) optimality conditions:

$$\begin{aligned} Ax - b &= 0 \\ A^T y + s - c &= 0 \\ XSe &= 0 \\ (x, s) &\geq 0. \end{aligned} \tag{1.1}$$

where $X = \text{diag}(x)$, $S = \text{diag}(s)$ and $e \in \mathcal{R}^n$ is the vector of all ones. Interior point methods approach the optimal solution by moving through the interior of the feasible region. This is done by introducing a central path C joined with a parameter $\tau > 0$. The central path C is an arc of strictly feasible points, which is defined as

$$C = \{(x, y, s) \in \mathcal{F}^0 : x_i s_i = \tau \text{ for all } i = 1, ..., n\},$$

where $\mathcal{F}^0$ is the primal-dual strictly feasible set defined by

$$\mathcal{F}^0 = \{(x, y, s) : Ax = b, A^T y + s = c, (x, s) > 0\}.$$

The KKT conditions are replaced by the following conditions:

$$\begin{aligned} Ax - b = 0 \\ A^T y + s - c = 0 \\ XSe = \tau e \\ (x, s) > 0. \end{aligned} \tag{1.2}$$

These conditions differ from the KKT conditions only in the term μ and in the requirement for (x, s) to be strictly positive. The central path C is well defined because the system (1.2) has unique solution for each $\tau > 0$. Furthermore, the points on the central path C converges to a primal-dual solution of the linear program (P) when τ converges to zero if $\mathcal{F}^0$ is nonempty. τ is equal to or smaller than the current barrier parameter $\mu = x^T s/n$. The target value $\tau = \sigma\mu$ is used, where $\sigma \in [0, 1]$ is the centering parameter. See [77].

The previous system (1.2) can be rewritten as the following

$$\begin{gathered} F(t) = \begin{bmatrix} Ax - b \\ A^T y + s - c \\ XSe - \sigma\mu e \end{bmatrix} = 0, \\ x > 0, \quad s > 0, \end{gathered} \tag{1.3}$$

where $t = (x, y, s)$.

Most primal-dual algorithms take Newton steps toward points on central path C for which $\mu > 0$, where the direction at each iteration is computed according to

$$F^{'}(t)\Delta t = -F(t), \tag{1.4}$$

where $F^{'}(t)$ is the derivative of $F(t)$. That yields

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = - \begin{bmatrix} Ax - b \\ A^T y + s - c \\ XSe - \sigma\mu e \end{bmatrix}. \tag{1.5}$$

In computational practice, (1.5) is reduced: after substituting

$$\Delta s = -X^{-1}S\Delta x - s + \sigma\mu X^{-1}e, \tag{1.6}$$

in the second row we get the following symmetric indefinite system of linear equations, usually called the augmented system

$$\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}, \tag{1.7}$$

where $\Theta = XS^{-1}$, $f = A^T y - c + \sigma\mu X^{-1}e$ and $g = Ax - b$. In many implementations, (1.7) is further reduced to the normal equations form

$$A\Theta A^T \Delta y = A\Theta f + g. \tag{1.8}$$

1.1 Motivation

The goal of this thesis is to explore how existing techniques in the areas of numerical analysis and linear algebra can be refined and combined into a new approach of a new inexact Newton method iteration to be employed in interior point methods. We are interested in using the preconditioned conjugate gradient method to solve the augmented system (1.7) and studying the convergence behaviour of the resulting interior point algorithm.

In each iteration of interior point methods, one of the linear systems (1.7) or (1.8) has to be solved. The main computational effort of an interior point iteration is the solution of these linear systems. Accordingly, in recent years extensive research has been devoted to developing techniques for solving these systems. In chapter 2 we survey some of the popular solution methods for these linear systems.

Historically, the normal equations system (1.8) was solved directly, because this system is symmetric and positive definite and its dimension is smaller compared to the augmented system [52, 73]. In [24, 31, 32, 35], Cholesky factorisation is used to factorise the normal equations matrix into a lower triangular matrix multiplied with its transpose, then forward and backward substitutions are used to solve the normal equations. In order to speed up solving a linear system by Cholesky factorisation, a reordering for sparsity is required. There are two famous heuristic orderings, the minimum degree and the minimum local fill-in orderings, see [24, 31, 32, 35].

The size of optimization problems has been increasing dramatically. Solving the linear systems (1.7) and (1.8) with a direct method is often very difficult for large problems, even when ordering to exploit the sparsity is taken into consideration. This is due to three main reasons. Firstly, the normal equations (1.8) may easily get dense even though the constraint matrix A is not. Secondly, although the augmented system is usually sparse for a sparse constraint matrix, it is nevertheless, indefinite. Finally, the linear systems (1.7) and (1.8) become extremely ill-conditioned as the IPM approaches the solution, which leads to numerical instability. These difficulties make many researchers interested in finding alternative techniques for solving the linear systems (1.7) and (1.8). The idea was to use an iterative method to solve these linear systems. Iterative methods however, usually fail to solve these systems without preconditioning. The term preconditioning refers to transforming a linear system into another system with more favorable properties for iterative solution [75]. Therefore, there is an urgent need for designing good preconditioners, as a good preconditioner is the key ingredient for solving a linear system iteratively. That makes a significant number of researchers tackle this issue [10, 28, 44, 45].

For the same reasons as above the normal equations system is nominated again to be solved by using an iterative method. As the system is symmetric and positive definite, the preconditioned conjugate gradient (PCG) method [42] is an appropriate iterative method to solve this system. The PCG method is one of the most popular iterative methods, because it is a short recurrence method and it has strong convergence properties, see section 2.4. In [15, 42, 45, 54, 55], the PCG method is used to solve the normal equations. The preconditioners in [45, 54, 55] are the incomplete Cholesky factorisation of the normal equations matrix. The incomplete Cholesky factorisation was proposed by Meijerink and Van Der Vorst (1977) [56] to be used with symmetric Hermitian matrices. There are two strategies of identifying the position of the nonzero elements in this factorisation: the fixed fill-in strategy and the drop-tolerance strategy, see [12]. These types of preconditioner do not always work as well as expected. However, they are constructed by using fascinating techniques of linear algebra. These preconditioners are effective in the early stage of IPM, but they start to struggle in the final iterations. This is due to the extreme ill-conditioned nature of this system in the final iterations of IPM. Therefore, it is necessary to design a preconditioner after understanding the nature of the problem, in particular at the final iterations of IPM.

We notice that iterative methods struggle to solve the linear systems in the final iterations of an IPM, due to the extreme ill-conditioning of these systems. Therefore we are concerned with finding an iterative approach to solve these linear systems efficiently in the final iterations of an IPM. In this thesis we will convert the disadvantages of the final iterations of IPM into an advantage, and we will construct our preconditioner for the augmented system (1.7) by exploiting the issues that leads to the ill-conditioning of this system.

There are many important reasons why we choose to work with the augmented system. The first reason is that the augmented system is sparser compared with the normal equations. Factoring the augmented system (1.7) often produces significant savings in the number of nonzero entries over factoring the normal equations. The existence of a dense column in the constraint matrix A results in a straightforward dense normal equations matrix. For an example of such a situation, see [3, 24] and the references therein. Compared with Cholesky factorisation for the normal equations, the augmented system factorisation enjoys an additional degree of freedom resulting from the ability to interchange pivots between diagonal elements of Θ and diagonal elements of the already filled $(2, 2)$ block in (1.7). We aim to exploit these advantages when we construct our preconditioner for the augmented system.

The second reason is that the augmented system may have a better condition number compared to the normal equations, after suitable scaling as suggested in [4]. The ill-conditioning in these systems is due to the matrix Θ, since some of its elements move toward zero and the others move toward infinity. The position of Θ in the augmented system makes it easier to control the ill-conditioning of the augmented system when designing a preconditioner.

The final reason comes from the analysis by Oliveira and Sorensen [60] who propose a preconditioner for the augmented system (1.7), and then reduce the preconditioned system to positive definite normal equations, allowing them to use the conjugate gradients method to solve (1.8). They show in [60] that all preconditioners for the normal equations system have an equivalent for the augmented system, while the converse is not true. More precisely, they show that the whole classes of (different) preconditioners for the augmented system can result in the same preconditioner for the normal equations. We consider this to be a strong argument for constructing a preconditioner for the augmented system.

1.2 Contributions

The contributions of this research are as follows.

First, we design a block triangular preconditioner for the augmented system (1.7). To construct this preconditioner, we partition the constraint matrix A into two matrices. The first one is nonsingular matrix with size m, while the other one is the remaining matrix. The idea is to use the basic and nonbasic partition which is used in the simplex method, with one mean different; in the simplex method one has exactly m basic and $n-m$ nonbasic variables at each iteration, while in interior point method this is true in the optimal solution. So, such partition becomes clearer at final iterations of interior point method, where we suggest using our preconditioner. The nonsingular matrix in our partition represents an approximation of the basic part of the variables. After designing this preconditioner, we perform a spectral analysis of the preconditioned matrix. We also show that the preconditioned matrix has $n + p$ unit eigenvalues and the remaining eigenvalues are positive and greater or equal one, where p is the rank of the second matrix of the partition of A.

We propose preconditioner for the augmented system and go a step further than in [60]. Instead of reducing the augmented system to normal equations and then applying an iterative method, we use the preconditioned conjugate gradients method to solve the indefinite system (1.7). We are aware of the disadvantages associated with applying the PCG method to indefinite systems [26]. However, we are motivated by the recent analyses of Lukšan and Vlček [51] and Rozlozník and Simoncini [65] showing that short recurrence iterative methods such as conjugate gradients can be applied to indefinite systems in certain situations. We show in particular that the analysis of [65] may be applied to the preconditioner proposed in this thesis. We prove that the PCG method, when applied to the indefinite system (1.7) preconditioned with our proposed preconditioner, converges as in the case of a symmetric positive definite system. The convergence of the PCG method is proved by showing that the error term and the residual converge to zero. The error and the residual bounds are given by Theorem 3.3.4 and Theorem 3.3.5 respectively, which is related to symmetric positive definite matrices.

We have implemented this iterative approach in the final iterations of the interior point solver HOPDM when the normal equations system becomes ill conditioned. The implementation within HOPDM shows remarkable improvement on a series of problems, see the numerical results in Chapter 5.

A consequence of using an iterative method to solve the linear systems which arise in interior point methods is that the search direction is computed approximately. Hence, instead of the pure Newton iteration (1.4), we now have the following

$$F^{'}(t^k)\Delta t^k = -F(t^k) + r^k,$$

which is an inexact Newton iteration. This causes a major difference in an interior point algorithm, whose convergence is proved under the assumption that the search directions are calculated exactly. Our final contribution is the convergence analysis of an interior point algorithm with our specific inexact Newton direction.

We use the PCG method to solve the augmented system preconditioned with a block triangular matrix P. This yields a specific inexact interior point method. In this thesis we focus on the convergence analysis of one interior point algorithm for this inexact case. This algorithm is the infeasible path-following (IPF) algorithm. For the inexact case, we refer to this algorithm as the inexact infeasible path-following (IIPF) algorithm.

We prove global convergence and provide a complexity result for the IIPF algorithm. We design a suitable stopping criteria for the PCG method. This plays an important role in the convergence of the IIPF algorithm. This stopping criterion allows a low accuracy when the current iterate is far from the solution. We impose some conditions on the forcing term of the inexact Newton method in order to prove the convergence of the IIPF algorithm. Note that the same analysis can be used in the cases where the augmented system is solved iteratively, providing that the residual of this iterative method has a zero block in its second component corresponding to $(2, 2)$ block in (1.7) such that $r = [r_1, 0]$. Thus we can carry out this approach to cases like [65], for example.

The original results presented in this thesis have been the basis for two papers that have been accepted for publication, jointly with Jacek Gondzio and Julian Hall [2], and with Jacek Gondzio [1].

1.3 The structure of the thesis

This thesis is structured as follows. In Chapter 2, we introduce and formalise the primal-dual interior point method for linear programming. Also in this chapter we present some of the well known feasible and infeasible interior point algorithms. Moreover, Chapter 2 review the convergence behaviour of Newton and inexact Newton methods. Furthermore, in this chapter we discuss several well known methods to solve a linear system. We introduce briefly a few direct methods and discuss extensively several iterative methods. As in this thesis we are concerned with the use of an iterative method to solve the linear systems which arise from IPMs, we mainly focus on the Krylov subspace methods in this chapter.

In Chapter 3 firstly, we present preconditioners for the augmented system which have been constructed in the last few years. Secondly, we propose our new block triangular preconditioner and we perform a spectral analysis of this preconditioner. Moreover, in this chapter we take a closer look at the behaviour of conjugate gradients for the indefinite system: we follow [65] in the analysis of our preconditioner. Furthermore, we prove that the convergence of the PCG method applied to the indefinite system (1.7) preconditioned with the proposed preconditioner, is similar to the convergence of the PCG method applied to a positive definite system. Finally, we discuss the issues involved in the identification of a suitable subset of columns to produce a well-conditioned matrix.

In Chapter 4 we compute the residual of the inexact Newton method and choose suitable stopping criteria to the PCG method which makes sense for the convergence of the inexact Newton method. In addition in this chapter we perform the convergence analysis and provide the complexity result for the IIPF Algorithm.

We have implemented the conjugate gradients method with the indefinite preconditioner in the context of the HOPDM interior point solver and we have applied it to solve a number of medium and large-scale linear programming problems. In Chapter 5, we discuss our computational experience. In Chapter 6 we draw our conclusions and discuss possible future developments.

1.4 Notations

Throughout the thesis, we use the following notation. By $\mathcal{R}$ we denote the set of real number. For a natural number n, the symbol $\mathcal{R}^n$ denotes the set of vectors with n components in $\mathcal{R}$. Greek letters denote scalars, lower-case letters denote vectors and upper-case letters denote matrices. The ith row and jth column component of the matrix A is denoted by a_{ij}. The identity matrix will be denoted by I, a subscript will determine its dimension when it is not clear from context. The symbol $\|.\|$ represents the Euclidean norm ($\|x\| = \sqrt{x^T x}$). The symbol $\|.\|_G$ represents the G-norm for a symmetric positive definite matrix G ($\|x\|_G = \sqrt{x^T G x}$). The $\mathcal{F}$ and $\mathcal{F}^0$ denote the primal-dual feasible and strictly feasible sets respectively. The $\mathcal{N}_2()$ or $\mathcal{N}_{-\infty}()$

denote the interior point method neighbourhood, since most primal-dual algorithms take Newton step toward points in specific neighbourhood. The point $t^* = (x^*, y^*, s^*)$ denotes the optimal solution of interior point method. The sequence $\{t^k\} = \{(x^k, y^k, s^k)\}$ denotes the interior point iterations. The $\xi^k = (\xi_p^k, \xi_d^k, \xi_\mu^k)$ denotes the right hand side of the Newton method system (1.5) at iterate k. The $r^k = (r_p^k, r_d^k, r_\mu^k)$ denotes the inexact Newton method residual at iterate k. The r_{PCG}^k denotes the residual on the kth PCG iteration. The e^k denotes the error on the kth PCG iteration, unless otherwise stated. For any vector v is in (1.7), $v = [v_1, v_2]$ and $v_1 = [v_B, v_N]$, where $v_B \in \Re^m$. The PCG method residual $r_{PCG}^k = [r_1^k, r_2^k]$ and $r_1^k = [r_B^k, r_N^k]$.

Chapter 2

Fundamentals

2.1 The Interior Point Method

2.1.1 The IPM for linear programming

It is widely accepted that the primal-dual interior point method is the most efficient variant of interior point algorithms for linear programming [3, 77]. The usual transformation in interior point methods consists of replacing inequality constraints by the logarithmic barrier. The primal barrier problem becomes:

$$\begin{array}{ll} \min & c^T x - \mu \sum_{j=1}^{n} \ln x_j \\ \text{s.t.} & \qquad Ax \ = b, \end{array}$$

where $\mu > 0$ is a barrier parameter. The Lagrangian associated with this problem has the form:

$$L(x, y, \mu) = c^T x - y^T (Ax - b) - \mu \sum_{j=1}^{n} \ln x_j$$

and the conditions for a stationary point are

$$\begin{aligned}\nabla_x L(x,y,\mu) &= c - A^T y - \mu X^{-1} e &= 0\\ \nabla_y L(x,y,\mu) &= Ax - b &= 0,\end{aligned}$$

where $X^{-1} = \text{diag}\{x_1^{-1}, x_2^{-1}, \ldots, x_n^{-1}\}$. Denoting $s = \mu X^{-1} e$, i.e. $XSe = \mu e$, where $S = \text{diag}\{s_1, s_2, \ldots, s_n\}$ and $e = (1, 1, \ldots, 1)^T$, the first order optimality conditions (for the barrier problem) are:

$$\begin{aligned} Ax &= b,\\ A^T y + s &= c,\\ XSe &= \mu e\\ (x,s) &> 0. \end{aligned} \tag{2.1}$$

The interior point algorithm for linear programming applies Newton method to solve this system of nonlinear equations and gradually reduces the barrier parameter μ to guarantee convergence to the optimal solution of the original problem. The Newton direction is obtained by solving the system of linear equations:

$$\begin{bmatrix} A & 0 & 0\\ 0 & A^T & I\\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x\\ \Delta y\\ \Delta s \end{bmatrix} = \begin{bmatrix} b - Ax\\ c - A^T y - s\\ -XSe + \mu e \end{bmatrix}, \tag{2.2}$$

By eliminating

$$\Delta s = -X^{-1} S \Delta x + \mu X^{-1} e,$$

from the second equation we get the symmetric indefinite augmented system of linear equations

$$\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}.$$

where $\Theta = XS^{-1} \in \mathcal{R}^{n \times n}$ is a diagonal scaling matrix and the right-hand-side vectors satisfy $f = A^T y + s - c - X^{-1}(XSe - \mu e)$ and $g = Ax - b$.

2.1.2 The Primal-Dual Interior Point Algorithms

Primal-dual interior point algorithms are the most important, efficient and useful interior point algorithms for linear programming. That is because of the strong theoretical properties and the practical performance of these algorithms. Since 1994 researchers have understood well the properties and theoretical background of primal-dual interior point algorithms [3, 37, 53, 77, 79] and the references therein. In this section we briefly review several feasible primal-dual interior point algorithms and an infeasible primal-dual interior point algorithm.

Primal-dual interior point methods find primal-dual solutions (x^*, y^*, s^*) by applying Newton method to the optimality conditions in (2.1) and by modifying the search directions and step lengths so that the inequality $(x, s) > 0$ is satisfied strictly at every iteration [77]. Most primal-dual algorithms take Newton steps toward points in a specific neighbourhood. This neighbourhood guarantees to keep (x, s) strictly positive and to prevent $x_i s_i$ from becoming too small relatively for all $i = 1, ..., n$. In this section we introduce a few feasible primal-dual interior point algorithms and an infeasible primal-dual interior point algorithm. For a feasible algorithm the two most interesting neighbourhoods are $\mathcal{N}_2$ and $\mathcal{N}_{-\infty}$. The $\mathcal{N}_2$ neighbourhood is defined by

$$\mathcal{N}_2(\theta) = \{(x, y, s) \in \mathcal{F}^0 : \|XSe - \mu e\|_2 \leq \theta\mu\}$$

for some $\theta \in (0, 1)$. The $\mathcal{N}_{-\infty}$ neighbourhood is defined by

$$\mathcal{N}_{-\infty}(\gamma) = \{(x, y, s) \in \mathcal{F}^0 : x_i s_i \geq \gamma\mu, \forall i = 1, 2, ..., n\}$$

for some $\gamma \in (0, 1)$. By choosing γ close to zero, $\mathcal{N}_{-\infty}(\gamma)$ encompass most of the feasible region. However, $\mathcal{N}_2(\theta)$ is more restrictive, since certain points in $\mathcal{F}^0$ do not belong to $\mathcal{N}_2(\theta)$ no matter how close θ is chosen to its upper bound [77]. In other words, $\mathcal{N}_2(\theta)$ contains only a small fraction of the points in $\mathcal{F}^0$, while $\mathcal{N}_{-\infty}(\gamma)$ takes up almost all the entire of $\mathcal{F}^0$ for small γ, which makes $\mathcal{N}_{-\infty}(\gamma)$ much more expansive when γ is small. See [77].

For infeasible algorithms neighbourhoods should guarantee an extra condition; namely the residuals should decrease at each iteration. The extension of $\mathcal{N}_{-\infty}(\gamma)$ for infeasible algorithms is $\mathcal{N}_{-\infty}(\gamma, \beta)$, which is defined by

$$\begin{aligned}\mathcal{N}_{-\infty}(\gamma, \beta) = \{(x, y, s) : \|(\xi_p, \xi_d)\|/\mu \leq \beta\|(\xi_p^0, \xi_d^0)\|/\mu_0,\ (x, s) > 0,\\ x_i s_i \geq \gamma\mu,\ i = 1, 2, ..., n\},\end{aligned}$$

where the residuals $\xi_p = Ax - b$ and $\xi_d = A^T y + s - c$. See [77].

In primal-dual interior point methods, the initial solution (x^0, y^0, s^0) should belong to the neighbourhood. At each iteration, solution should also belong to this neighbourhood. The solution at iteration k is given by $(x^k, y^k, s^k) = (x^{k-1}, y^{k-1}, s^{k-1}) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k)$, where α_k is the step length and $(\Delta x^k, \Delta y^k, \Delta s^k)$

is the direction, which is given by:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} Ax^k - b \\ A^T y^k + s^k - c \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix}, \tag{2.3}$$

where $\sigma_k \in [0, 1]$ is centering parameter. Choosing σ_k plays a crucial role in primal-dual interior point algorithms. If $\sigma_k = 1$, the equation (2.3) gives a centering direction, which improves centrality (all $x_i s_i$ are close to μ) and makes little progress in reducing μ. If $\sigma_k = 0$ that gives the standard Newton step, which reduces μ. One can choose the centering parameter σ_k and the step length α_k to ensure that an iterate stays within the chosen neighbourhood. See [77].

For feasible algorithms, the iterations belong to $\mathcal{F}^0$, so for any iteration k we have $Ax^k = b$ and $A^T y^k + s^k = c$. That makes the first and the second rows of the right hand side of (2.3) equal to zero. So for feasible algorithms (2.3) replaced by:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix}. \tag{2.4}$$

The interior point algorithms which we mention in this section have a global linear convergence. An algorithm has a global convergence if the algorithm guarantees to converge to the solution from any approximation. The sequences $\{\mu_k\}$ converges linearly to zero if $\mu_{k+1} \leq \delta \mu_k$, where $\delta \in (0, 1)$. Knowing that an algorithm has global convergence and the rate of this convergence alone will not give the whole picture. It is necessary, to know the time requires an algorithm to solve a given instance of linear programming problem. Complexity theory has been concerned with the worst case behaviour of algorithms. Complexity result is an upper bound on the time required algorithm to solve a problem. For example, the short-step path-following algorithm has a polynomial complexity result in the order of $O(\sqrt{n}\log 1/\epsilon)$, where $\epsilon > 0$. This gives that there is an index K with $K = O(\sqrt{n}\log 1/\epsilon)$ such that $\mu_k \leq \epsilon$ for all $k \geq K$. See [77].

The Short-Step Path-Following Algorithm (SPF Algorithm):

- Given $\theta = 0.4, \sigma = 1 - 0.4/\sqrt{n}$, and $(x^0, y^0, s^0) \in \mathcal{N}_2(\theta)$.
- For $k = 0, 1, ...$

  set $\sigma_k = \sigma$ and solve (2.4) to obtain $(\Delta x^k, \Delta y^k, \Delta s^k)$;

  set $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + (\Delta x^k, \Delta y^k, \Delta s^k)$.

This algorithm has a global linear convergence and a polynomial complexity result in the order of $O(\sqrt{n}\log 1/\epsilon)$ [77].

The Predictor-Corrector Algorithm (PC Algorithm):

- Given $(x^0, y^0, s^0) \in \mathcal{N}_2(0.25)$.
- For $k = 0, 1, ...$

```
    if $k$ is even (predictor step)
        solve (2.4) with $\sigma_k = 0$ to obtain $(\Delta x^k, \Delta y^k, \Delta s^k)$; choose $\alpha_k$ as
        the largest value of $\alpha \in [0, 1]$ such that $(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in$
        $\mathcal{N}_2(0.5)$, where
        $(x^k(\alpha), y^k(\alpha), s^k(\alpha)) = (x^k, y^k, s^k) + \alpha(\Delta x^k, \Delta y^k, \Delta s^k)$;
        set $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k(\alpha), y^k(\alpha), s^k(\alpha))$;
```

```
else (corrector step)
    solve (2.4) with $\sigma_k = 1$ to obtain $(\Delta x^k, \Delta y^k, \Delta s^k)$;
    set $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + (\Delta x^k, \Delta y^k, \Delta s^k)$
```

The parameter σ_k is chosen to be either 0 or 1. This choice has the following meaning: improving centrality (corrector step) and reducing the duality measure μ (predictor step). Also this algorithm has a global linear convergence and a polynomial complexity result in the order of $O(\sqrt{n} \log 1/\epsilon)$. However, this algorithm is a definite improvement over the short-step algorithm because of the adaptivity that is built into the choice of predictor step length. See [77].

The Long-Step Path-Following Algorithm (LPF Algorithm):

```
• Given $\gamma, \sigma_{min}, \sigma_{max}$ with $\gamma \in (0,1)$, $0 < \sigma_{min} < \sigma_{max} < 1$, and
  $(x^0, y^0, s^0) \in \mathcal{N}_{-\infty}(\gamma)$.

• For $k = 0, 1, ...$
  set $\sigma_k \in [\sigma_{min}, \sigma_{max}]$;
  solve (2.4) to obtain $(\Delta x^k, \Delta y^k, \Delta s^k)$;
  choose $\alpha_k$ as the largest value of $\alpha \in [0,1]$ such that $(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma)$;
  set $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k(\alpha), y^k(\alpha), s^k(\alpha))$.
```

This algorithm has a global linear convergence and a polynomial complexity result in the order of $O(n \log 1/\epsilon)$ [77]. In [39] the authors show that the complexity result for long-step primal-dual algorithm is $O(\sqrt{n}L)$ iterations where L is the size of the input.

In most cases it is quite difficult to find a strictly feasible starting point (a point which belongs to $\mathcal{F}^0$). In this case one can use an infeasible interior point algorithm.

The Infeasible Path-Following Algorithm (IPF Algorithm):

1. Given $\gamma, \beta, \sigma_{min}, \sigma_{max}$ with $\gamma \in (0,1)$, $\beta \geq 1$, and $0 < \sigma_{min} < \sigma_{max} < 0.5$; choose (x^0, y^0, s^0) with $(x^0, s^0) > 0$;

2. For $k = 0, 1, 2, ...$

    - choose $\sigma_k \in [\sigma_{min}, \sigma_{max}]$; and solve

    $$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} \xi_p^k \\ \xi_d^k \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix},$$

    where $\xi_p^k = Ax^k - b$ and $\xi_d^k = A^T y^k + s^k - c$

    - choose α_k as the largest value of α in $[0,1]$ such that

    $$(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma, \beta)$$

    and the following Armijo condition holds:

    $$\mu_k(\alpha) \leq (1 - .01\alpha)\mu_k;$$

    - set $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k(\alpha_k), y^k(\alpha_k), s^k(\alpha_k))$;

    - stop when $\mu_k < \epsilon$, for a small positive constant ϵ.

β is chosen such that $\beta \geq 1$ to ensure that the initial point belongs to the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$. This algorithm has a global linear convergence and a polynomial complexity result in the order of $O(n^2 |\log \epsilon|)$ [77]. In [78] the author shows that the complexity result for the infeasible path-following algorithm is $O(\sqrt{n}L)$ iterations where L is the size of the input.

2.2 Newton method

In this section we give a closer look at Newton method, inexact Newton method and their convergence analysis. However, the convergence analysis of interior point methods follow a different path from the convergence analysis of Newton method, even though, interior point method takes Newton steps toward points on certain neighbourhood. Newton method is an iterative method which is used to solve a system of nonlinear equations. See [47, 61].

$$F(t) = 0. \tag{2.5}$$

Newton iterations are given by

$$t^{k+1} = t^k - F^{'}(t^k)^{-1}F(t^k), \tag{2.6}$$

where t^{k+1} is the new iterate and t^k is the current iterate.

Assume the problem (2.5) has the solution t^*. We can approximate the function with a polynomial by using Taylor expansion about t^k.

$$F(t) = F(t^k) + F^{'}(t^k)(t - t^k) + \frac{F^{''}(t^k)}{2}(t - t^k)^2 + ...$$

$$F(t) \approx M_k(t) = F(t^k) + F^{'}(t^k)(t - t^k)$$

Let t^{k+1} be the root of the $M_k(t)$ then

$$0 = M_k(t^{k+1}) = F(t^k) + F^{'}(t^k)(t^{k+1} - t^k)$$

which implies (2.6).

Let $\Delta t^k = t^{k+1} - t^k$ then (2.6) becomes

$$F^{'}(t^k)\Delta t^k = -F(t^k). \tag{2.7}$$

See [47] for more details.

2.2.1 The convergence of Newton method

The Newton method is attractive because it converges quadratically starting from any sufficiently good initial guess t^0. See [47].

Definition: $\beta(\delta)$ denote the ball of radius δ about the solution t^*

$$\beta(\delta) = \{t : \|e\| < \delta\},$$

where e is the error of the current iterate, $e = t - t^*$.

The standard assumptions:

1. Equation (2.5) has a solution t^*.
2. $F^{'}$ is Lipschitz continuous with Lipschitz constant γ.
3. $F^{'}(t^*)$ is nonsingular.

The following theorem shows that if the standard assumptions hold the function $F(t)$ satisfies the following properties, Kelley [47, Theorem 5.1.1].

Theorem 2.2.1. *Let the standard assumptions hold. If there are $K > 0$ and $\delta > 0$ such that $K\delta < 1$ and $t^k \in \beta(\delta)$, where the Newton iterate t^k given by (2.6), then*

$$\|e^{k+1}\| \leq K\|e^k\|^2. \tag{2.8}$$

This theorem shows that Newton method has a local convergence, since the initial solution t^0 is chosen to be near the solution t^*. Furthermore, Newton method converges quadratically, see (2.8).

2.2.2 Termination of the iteration

The iteration is terminated when the ratio $\|F(t)\|/\|F(t^0)\|$ is small [47]. More generally the termination conditioned can be written as

$$\|F(t)\| \leq \tau_r\|F(t^0)\| + \tau_a, \tag{2.9}$$

where τ_r is the relative error tolerance and τ_a is the absolute error tolerance [47].

2.2.3 Error in the function and derivative

Suppose that F and $F^{'}$ are computed inaccurately so that $F + \varepsilon$ and $F^{'} + \zeta$ are used instead of F and $F^{'}$ in iterations. Under this case Newton iterations should be

$$t^{k+1} = t^k - (F^{'}(t^k) + \zeta(t^k))^{-1}(F(t^k) + \varepsilon(t^k)). \tag{2.10}$$

Theorem 2.2.2. *Let the standard assumptions hold. Assume $F^{'}(t^k) + \zeta(t^k)$ is nonsingular. If there are $\bar{K} > 0$, $\delta > 0$, and $\delta_1 > 0$ such that $\|\zeta(t^k)\| < \delta_1$*

and $t^k \in \beta(\delta)$, where t^k is given by (2.10), then

$$\|e^{k+1}\| \leq \bar{K}(\|e^k\|^2 + \|\zeta(t^k)\|\|e^k\| + \|\varepsilon(t^k)\|). \tag{2.11}$$

Proof: Kelly [47, Theorem 5.4.1].

2.3 Inexact Newton method

Solving the linear equation (2.7) exactly can be very expensive. Therefore, this linear equation can be solved approximately by using an iterative method. So instead of (2.7) we get

$$F^{'}(t^k)\Delta t^k = -F(t^k) + r^k. \tag{2.12}$$

The process is stopped when the residual r^k satisfies

$$\|r^k\| \leq \eta_k \|F(t^k)\|. \tag{2.13}$$

We refer to the term η_k as the forcing term. See [20, 47].

2.3.1 The convergence of Inexact Newton Method

The following theorems illustrate the convergence of inexact Newton method. By comparing the error of Newton method (2.8) with the error of inexact Newton method (2.14), we note that the forcing term in the condition (2.13) plays an important role in the convergence of inexact Newton method. Therefore, choosing a stopping criterion for the residual of inexact Newton method affects directly on its convergence.

Theorem 2.3.1. *Let the standard assumptions hold. If there are δ and K_I such that $t^k \in \beta(\delta)$ and (2.13), where Δt^k satisfies (2.12), then*

$$\|e^{k+1}\| \leq K_I(\|e^k\| + \eta_k)\|e^k\|. \tag{2.14}$$

Proof: Kelly [47, Theorem 6.1.1].

However, in the Newton method the error term satisfies

$$\|e^{k+1}\| \leq K\|e^k\|^2.$$

Theorem 2.3.2. *Let the standard assumptions hold. If there are δ and $\overline{\eta}$ such that $t^0 \in \beta(\delta)$ and $\{\eta_k\} \subset [0, \overline{\eta}]$, then the inexact Newton iteration t^{k+1}, which satisfies (2.13), converges linearly to t^*. Moreover,*

- *if $\eta_k \to 0$ the convergence is superlinear.*
- *if $\eta_k \leq K_\eta \|F(t^k)\|^p$ for some $K_\eta > 0$ the convergence is superlinear with order $1 + p$.*

Proof: Kelly [47, Theorem 6.1.4].

The superliner convergence is defined as the following

$$\|e^{k+1}\| \leq \epsilon\|e^k\|, \text{ where } \epsilon \to 0.$$

The superliner convergence with order $1 + p$ is defined as the following

$$\|e^{k+1}\| \leq \epsilon\|e^k\|^p, \text{ where } \epsilon \in (0, 1).$$

2.4 Methods for solving a linear system

In this section, we discuss several methods to solve the following linear system $\mathcal{H}u = q$. This system represents either the augmented system (1.7) or the normal equations (1.8). $\mathcal{H} \in \mathcal{R}^{\ell\times\ell}$, where $\ell = n + m$ for the augmented system and $\ell = m$ for the normal equations, respectively.

For most problems, the linear system $\mathcal{H}u = q$ is sparse. Before introducing methods for solving this system, we first focus on the sparsity of linear system.

2.4.1 Sparse Matrices

A matrix is sparse if many of its coefficients are zero. It is very important to highlight sparsity for two main reasons. Firstly, many large scale problems, which occur in practice, have sparse matrices. Secondly, exploiting sparsity can lead to enormous computational saving. See [24]. To illustrate the potential saving from exploiting sparsity, we consider a small example. Suppose we want to solve a system with the following matrix

$$\mathcal{H} = \begin{bmatrix} \blacksquare & \blacksquare & \blacksquare & \blacksquare \\ \blacksquare & \blacksquare & & \\ \blacksquare & & \blacksquare & \\ \blacksquare & & & \end{bmatrix},$$

The term $\blacksquare$ represents a nonzero coefficient, while the coefficients are zero elsewhere.

Gaussian elimination can be used, for instance, to solve this system. It is used to reduce the matrix $\mathcal{H}$ to an equivalent upper triangular matrix U by applying rows operations. The first step of Gaussian elimination leads to the following matrix

$$\begin{bmatrix} \blacksquare & & & \\ \blacksquare & \blacksquare & f & f \\ \blacksquare & f & \blacksquare & f \\ \blacksquare & f & f & f \end{bmatrix},$$

where f represents a fill-in. The elimination operations change a zero coefficient into a nonzero one, (we refer to this by the term fill-in). A fill-in requires additional storage and operations. This elimination leads to full active submatrix (3×3 matrix; its columns: 2,3,4 and its rows: 2,3,4). Consequently, all Gaussian elimination steps from no one will be dense.

However, if we do rows/columns ordering to $\mathcal{H}$ we can control the amount of fill-in. For our example, swapping between row 1 and row 4 leads to the equivalent matrix

$$\begin{bmatrix} \blacksquare & & & \\ \blacksquare & \blacksquare & & \\ \blacksquare & & \blacksquare & \\ \blacksquare & \blacksquare & \blacksquare & \blacksquare \end{bmatrix}.$$

That leads to an upper triangular matrix without requiring any eliminations. This saves us extra storages and extra operations.

The problem of finding the ordering which minimizes fill-in is NP-complete [77]. However, there are good ordering heuristics which preform quite well in practices. There are two famous heuristic orderings, the minimum degree and the minimum local fill-in orderings, see [24, 31, 32, 35].

2.4.2 Direct Methods

The main focus of this thesis is the use of an iterative method to solve the linear system which arises from the IPMs. However, we will highlight briefly some direct methods.

2.4.2.1 Gaussian elimination

Gaussian elimination is one of the most well known direct methods. It is used to reduce the matrix $\mathcal{H}$ to an upper triangular matrix U by applying row operations. Diagonal elements are chosen to be the pivots. If a diagonal element is zero, a row interchange has to be carried out. The reduction for $\mathcal{H}$ is performed by using elementary row operations which can be written as

$$L_{\ell-1}...L_2L_1\mathcal{H} = U.$$

That can be written as

$$\mathcal{H} = LU,$$

where L is a unit lower triangular matrix, and its elements l_{ij} are precisely the multipliers which are used in the elimination to vanish the element at the (i, j) position in U. This decomposition of $\mathcal{H}$ is called LU factorisation of $\mathcal{H}$. See [57] for more details. The computation cost of this method can be expressed as $\frac{2}{3}\ell + O(\ell^2)$ flops, where each addition, subtraction, multiplication, division or square root counts as a flops [71].

2.4.2.2 Cholesky factorisation

Cholesky factorisation method is used to decompose symmetric positive definite matrices. This factorisation produces a lower triangular matrix L with positive diagonal elements such that

$$\mathcal{H} = LL^T.$$

Solving $\mathcal{H}u = q$ is equivalent to solving two systems one with a forward substitution and the other with a backward substitution,

$$Lv = r, L^T u = v.$$

We assume the constraint matrix A has a full row rank, so the matrix of the normal equations system (1.8) will be symmetric and positive definite. The use of Cholesky factorization to solve the normal equations is a common choice, see for example [35].

2.4.3 Iterative Methods

The standard approach uses the direct method to solve the normal equations (symmetric positive definite system) by sparse Cholesky factorisation. However, for large-scale problems, the computational effort of direct methods can become sometimes very expensive. Therefore, an iterative method is employed to solve the linear system which arises from IPMs.

Iterative method solves the problem approximately. It generates a sequence of iterations starting from an initial guess and terminates when the found solution is close enough to the exact solution or when the residual gets sufficiently small.

2.4.3.1 Stationary Iterative Methods

The first iterative methods which were used to solve large linear systems were based on relaxation of the coordinates. Starting from initial approximation solution, these methods modify the approximation solution until convergence is reach. Each of these modifications, called relaxation steps [66]. The iterations of these methods are based on splitting the matrix $\mathcal{H}$ into the form $\mathcal{H} = \mathcal{H}_1 + \mathcal{H}_2$, where $\mathcal{H}_1$ is a non-singular matrix. Then the system $\mathcal{H}u = q$ is converted to the fixed point problem $u = \mathcal{H}_1^{-1}(q - \mathcal{H}_2 u)$. By beginning with an initial solution u^0 the iterations of these methods is generated by

$$u^{j+1} = \mathcal{H}_1^{-1} q - \mathcal{H}_1^{-1} \mathcal{H}_2 u^j.$$

See [66, 74, 80]. Among different stationary methods we mention: Jacobi, Gauss-Seidel, sucessive overrelaxation (SOR), Arrow-Hurwicz and Uzawa methods. The stationary methods are now more commonly used as preconditioners for the Krylov subspace methods.

Jacobi Method

Jacobi method uses the splitting $\mathcal{H}_1 = D$ and $\mathcal{H}_2 = L + U$, as the matrix $\mathcal{H}$ is written as the following $\mathcal{H} = D+L+U$, where D is diagonal matrix , L is a nondiagonal lower triangular matrix and U is a nondiagonal upper triangular matrix, [47]. Jacobi method converges to solution for all right hand side q, if $0 < \sum_{j \neq i} |h_{ij}| < |h_{ii}|$ for all $1 \leq i \leq \ell$, see [47, Theorem 1.4.1].

Gauss-Seidel Method

The coefficient matrix for this method is also written as the following $\mathcal{H} = D + L + U$. The Gauss-Seidel method uses the splitting $\mathcal{H}_1 = D + U$ and

$\mathcal{H}_2 = L$, [47]. This method converges for the same conditions of convergence of the Jacobi method, [43].

Arrow-Hurwicz and Uzawa Methods

These iterative methods are used to solve saddle point problems, such as the augmented system (1.7). The idea of these stationary methods is to split the matrix $\mathcal{H}$ so that these methods become simultaneous iterations for both Δx and Δy [8].

The iterations of Uzawa's method is given as follow:

$$\begin{aligned}\Delta x^{j+1} &= \Theta(A^T\Delta y^j - f),\\ \Delta y^{j+1} &= \Delta y^j + \omega(A\Delta x^{j+1} - g),\end{aligned}$$

where $\omega > 0$ is relaxation parameter. Accordingly, the splitting matrices are given by

$$\mathcal{H}_1 = \begin{bmatrix} -\Theta^{-1} & 0 \\ A & -\frac{1}{\omega}I \end{bmatrix}, \quad \mathcal{H}_2 = \begin{bmatrix} 0 & A^T \\ 0 & \frac{1}{\omega}I \end{bmatrix}.$$

The iterations of Arrow-Hurwicz method is given as follow:

$$\begin{aligned}\Delta x^{j+1} &= \Delta x^j + \alpha(f + \Theta^{-1}\Delta x^j - A^T\Delta y^j),\\ \Delta y^{j+1} &= \Delta y^j + \omega(A\Delta x^{j+1} - g),\end{aligned}$$

where α and ω are relaxation parameters. The splitting matrices are given by

$$\mathcal{H}_1 = \begin{bmatrix} \frac{1}{\alpha}I & 0 \\ A & -\frac{1}{\omega}I \end{bmatrix}, \quad \mathcal{H}_2 = \begin{bmatrix} -\frac{1}{\alpha}I - \Theta^{-1} & A^T \\ 0 & \frac{1}{\omega}I \end{bmatrix}.$$

For more detail on Arrow-Hurwicz and Uzawa methods see [8] and the references therein.

2.4.3.2 Krylov Subspace Methods

Krylov subspace methods are a family of iterative methods to solve a linear system of the form

$$\mathcal{H}u = q. \tag{2.15}$$

Krylov subspace methods extract an approximate solution u^j from an affine subspace $u^0 + \mathcal{K}_j$ of dimension j, where u^0 is an arbitrary initial guess to the solution of (2.15). The Krylov subspace is defined by

$$\mathcal{K}_j(\mathcal{H}, r^0) = span\{r^0, \mathcal{H}r^0, \mathcal{H}^2 r^0, ..., \mathcal{H}^{j-1}r^0\}, \tag{2.16}$$

for $j \geq 1$. The residual r^0 is given by $r^0 = q - \mathcal{H}u^0$. See [47, 66].

The dimension of the subspace increases by one at each step of the approximation process. The Krylov subspace has the following properties. The first property is that $\mathcal{K}_j$ is the space of all vectors in the space which can be written as $u = p_{j-1}(\mathcal{H})r^0$, where p_{j-1} is a polynomial of degree not exceeding $j-1$. The other property is that the degree of the minimal polynomial of r^0 with respect to $\mathcal{H}$ (it is a polynomial such that $p_{j-1}(\mathcal{H})r^0 = 0$) does not exceed the size of the space dimension ℓ [66].

There exist many Krylov subspace methods, a few of the most important ones will be highlighted in the following discussion.

Conjugate Gradient Method (CG)

Conjugate gradient (CG) method is one of the most popular iterative methods. This method is used to solve symmetric and positive definite linear systems. Many studies analyse the CG method [33, 42, 47, 66, 68, 72], and many papers use it to solve the linear systems which arise from interior point methods [13, 15, 40, 45, 54, 55].

In [42, 68] the authors explain the idea of conjugacy. The idea is to pick up a set of $\mathcal{H}$-orthogonal search directions and to take exactly one step with the right length, in each one of them. Then the solution will be found after ℓ steps. Two vectors v and w are $\mathcal{H}$-orthogonal if $v^T\mathcal{H}w = 0$.

At each step the iterate will be

$$u^{j+1} = u^j + \alpha_j d^j,$$

where α_j is the step length and d^j is the direction. Let the error term be defined by $e^j = u^j - u^*$. The step length α_j is chosen such that the search direction is $\mathcal{H}$-orthogonal to the error e^{j+1}. Consequently, α_j is chosen as follows:

$$(d^j)^T\mathcal{H}e^{j+1} = 0,$$

which implies

$$(d^j)^T\mathcal{H}(e^j + \alpha_j d^j) = 0.$$

That leads to

$$\alpha_j = -\frac{(d^j)^T\mathcal{H}e^j}{(d^j)^T\mathcal{H}d^j}.$$

Unfortunately, we do not know the e^j. If we know e^j the problem would be solved. On the other hand, the residual is given as $r^j = q - \mathcal{H}u^j$, which can be written $\mathcal{H}u^j = q - r^j$ which is equivalent to $\mathcal{H}u^j - \mathcal{H}u^* = q - r^j - \mathcal{H}u^*$ that leads to $\mathcal{H}e^j = -r^j$. So the step length can be written as

$$\alpha_j = \frac{(d^j)^T r^j}{(d^j)^T \mathcal{H} d^j}.$$

All we need now is to find the set of $\mathcal{H}$-orthogonal search direction $\{d^j\}$. In order to find this set, we assume we have ℓ linearly independent columns z^0, ..., $z^{\ell-1}$. We choose $d^0 = z^0$ and for $j > 0$, set

$$d^j = z^j + \sum_{k=0}^{j-1} \beta_{k,j} d^k.$$

The $\beta_{j,i}$ is chosen such that $(d^j)^T \mathcal{H} d^i = 0$ for $j > i$. So for $j > i$

$$\beta_{j,i} = -\frac{(z^j)^T \mathcal{H} d^i}{(d^i)^T \mathcal{H} d^i}.$$

In the CG method the search directions are constructed by the conjugation of the residuals. So $z^j = r^j$. This choice makes sense because the residual is orthogonal to the previous search directions which guarantees producing a new linearly independent search direction unless the residual is zero. When the residual is zero, the solution is found. These properties guarantee that the CG method is a short recurrence (CG method does not require to save all previous search directions) Krylov subspace method.

In the CG method α_j and $\beta_{j,i}$ can be expressed as

$$\alpha_j = \frac{(r^j)^T r^j}{(d^j)^T \mathcal{H} d^j}, \quad \beta_{j,i} = -\frac{(r^j)^T \mathcal{H} d^i}{(d^i)^T \mathcal{H} d^i} \quad \text{for } j > i.$$

where the search direction is written as

$$d^j = r^j + \sum_{k=0}^{j-1} \beta_{j,k} d^k.$$

The residual can be rewritten as

$$r^{i+1} = r^i - \alpha_i \mathcal{H} d^i.$$

That because $r^{i+1} = q - \mathcal{H}u^{i+1} = q - \mathcal{H}(u^i + \alpha_i d^i) = r^i - \alpha_i \mathcal{H} d^i$.

So we have

$$(r^j)^T r^{i+1} = (r^j)^T r^i - \alpha_i (r^j)^T \mathcal{H} d^i \Rightarrow \alpha_i (r^j)^T \mathcal{H} d^i = (r^j)^T r^i - (r^j)^T r^{i+1}.$$

Since the residual is orthogonal to the previous residuals [68]. This leads to

$$(r^j)^T \mathcal{H} d^i = \begin{cases} -\frac{1}{\alpha_{j-1}} (r^j)^T r^j & j = i+1, \\ 0 & j > i+1. \end{cases}$$

That gives

$$\beta_{j,i} = \begin{cases} \frac{(r^j)^T r^j}{(d^{j-1})^T r^{j-1}} & j = i+1, \\ 0 & j > i+1. \end{cases}$$

Let $\beta_j = \beta_{j,j-1}$. So the search direction can be expressed as

$$d^j = r^j + \beta_j d^{j-1}.$$

Consequently, CG method is a short recurrence method, because it is required to save the immediate previous direction only.

The CG Algorithm:

- Given an initial solution u^0, $r^0 = q - \mathcal{H}u^0$ and $d^0 = r^0$.
- For $j = 0, 1, ...$

```
while $\|r^j\| > \epsilon$ do
$\alpha_j = (r^j)^T r^j / (d^j)^T \mathcal{H} d^j,$
$u^{j+1} = u^j + \alpha_j d^j,$
$r^{j+1} = r^j - \alpha_j \mathcal{H} d^j,$
$\beta_{j+1} = (r^{j+1})^T r^{j+1} / (r^j)^T r^j,$
$d^{j+1} = r^{j+1} + \beta_{j+1} d^j.$
```

Theorem 2.4.1. *Let $\mathcal{H}$ be symmetric positive definite. Then the CG algorithm will find the solution within ℓ iterations. [47, Theorem 2.2.1].*

This theorem shows that the CG method finds the solution after a maximum of ℓ iterations. In practice however, accumulated floating point roundoff error causes the residual to lose accuracy gradually. This causes search directions to lose $\mathcal{H}$-orthogonality [68]. So providing the convergence analysis of CG method is essential.

Theorem 2.4.2. *Let e^0 be the initial error of the CG. Then*

$$\|e^j\|_{\mathcal{H}}^2 \leq \min_{P_i, P_i(0)=1} \max_{\lambda \in \Lambda(\mathcal{H})} [P_i(\lambda)]^2 \|e^0\|_{\mathcal{H}}^2,$$

where P_i is a polynomials of degree i and $\Lambda(\mathcal{H})$ is the set of eigenvalues of $\mathcal{H}$. See [68].

Theorem 2.4.3. *Let e^0 be the initial error of the CG. Then*

$$\|e^j\|_{\mathcal{H}} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^j \|e^0\|_{\mathcal{H}},$$

where κ is the condition number of the matrix $\mathcal{H}$ and $\|.\|_{\mathcal{H}}$ is the $\mathcal{H}$-norm for the symmetric positive definite matrix $\mathcal{H}$. See [68].

The condition number of a matrix defines as $\kappa = \frac{\lambda_{\min}}{\lambda_{\max}}$, where $\lambda_{\min}$ and $\lambda_{\max}$ are the minimum and maximum eigenvalues of the matrix $\mathcal{H}$ respectively. The previous theorem is not precise. Since, the CG method converges in a few iterations for a matrix which has a few distinct eigenvalues, even if it has large condition number.

Theorem 2.4.4. *Let $\mathcal{H}$ be symmetric positive definite. Assume that there are exactly $k \leq \ell$ distinct eigenvalues of $\mathcal{H}$. Then the CG iteration terminates in at most k iterations. [47, Theorem 2.2.3].*

The previous theorems show that the convergence of the CG method depends on the eigenvalues of the matrix of the linear system. The idea of preconditioning appears to improve the characteristic of the original matrix. Let P be a preconditioner. P is an approximation to $\mathcal{H}$ but is easier to invert and it is a symmetric positive definite matrix. Instead of solving (2.15), the system $P^{-1}\mathcal{H}u = P^{-1}q$ is solved. The CG method is applied for a symmetric positive definite system. P is symmetric positive definite matrix, so it can be written as $P = LL^T$. Accordingly, we solve the following system $L^{-1}\mathcal{H}L^{-T}\hat{u} = L^{-1}q$, where $\hat{u} = L^T u$. Applying the CG method to solve the preconditioned system $L^{-1}\mathcal{H}L^{-T}\hat{u} = L^{-1}q$ leads to preconditioned conjugate gradient (PCG) method [47, 66, 68].

The PCG Algorithm:

- Given an initial solution u^0, $r^0 = q - \mathcal{H}u^0$ and $d^0 = P^{-1}r^0$.

- For $j = 0, 1, ...$

```
while $\|r^j\| > \epsilon$ do
  $\alpha_j = (r^j)^T P^{-1} r^j / (d^j)^T \mathcal{H} d^j,$
  $u^{j+1} = u^j + \alpha_j d^j,$
  $r^{j+1} = r^j - \alpha_j \mathcal{H} d^j,$
  $\beta_{j+1} = (r^{j+1})^T P^{-1} r^{j+1} / (r^j)^T P^{-1} r^j,$
  $d^{j+1} = P^{-1} r^{j+1} + \beta_{j+1} d^j.$
```

Generalized Minimal Residual Method (GMRES)

CG method is used to solve a symmetric positive definite system. In 1986 the GMRES was proposed as a Krylov subspace method for solving a non-symmetric system [67]. GMRES method minimizes the residuals norm over all vectors in $u^0 + \mathcal{K}_k$. Suppose there is an orthogonal projector V_k onto $\mathcal{K}_k$, then any vector $u^k \in u^0 + \mathcal{K}_k$ can be written as

$$u^k = u^0 + V_k y,$$

where $y \in \mathcal{R}^k$. The GMRES generates iterations such that the residual r^k is minimized, which can be written as

$$\text{Min}_{u^k \in u^0 + \mathcal{K}_k} \|r^k\|.$$

On the other hand

$$\|r^k\| = \|q - \mathcal{H}u^k\| = \|q - \mathcal{H}(u^0 + V_k y)\| = \|r^0 - \mathcal{H}V_k y\|.$$

The columns of V_k are generated by using Arnoldi algorithm [47, Algorithm 3.4.1]. The starting vector is given as $v^1 = r^0/\|r^0\|$ and the following vectors are generated by

$$v^{j+1} = \frac{\mathcal{H}v^j - \sum_{i=1}^{j}((\mathcal{H}v^j)^T v^i)v^i}{\|\mathcal{H}v^j - \sum_{i=1}^{j}((\mathcal{H}v^j)^T v^i)v^i\|},$$

for $j \geq 0$.

Let H_k be constructed such that $h_{ji} = (\mathcal{H}v^j)^T v^i$ and $h_{ij} = 0$ for $i > j+1$. Then Arnoldi algorithm produces matrices V_k such that

$$\mathcal{H}V_k = V_{k+1}H_k.$$

Consequently, the residual norm becomes

$$\|r^k\| = \|r^0 - \mathcal{H}V_k y\| = \|r^0 - V_{k+1}H_k y\| = \|V_{k+1}(\beta e_1 - H_k y)\|.$$

That is because $v^1 = r^0/\|r^0\|$ and $\beta = \|r^0\|$, where $e_1 = [1, 0, ..., 0]$ and $e_1 \in \mathcal{R}^{k+1}$. See [47, 66].

The GMRES Algorithm:

- Given an initial solution u^0, $r^0 = q - \mathcal{H}u^0$, $v^1 = r^0/\|r^0\|$, $\rho_0 = \|r^0\|$, $\beta_0 = \rho_0$ and j=0.
- While $\rho_j > \epsilon\|q\|$ and $j < j_{max}$ do

  (a) $j = j + 1$.

  (b) For $i = 1, ..., j$

$$
\begin{aligned}
h_{ij} &= (\mathcal{H}v^j)^T v^i \\
v^{j+1} &= \mathcal{H}v^j - \textstyle\sum_{i=0}^{j} h_{ij} v^i, \\
h_{j+1,j} &= \|v^{j+1}\|, \\
v^{j+1} &= v^{j+1}/\|v^{j+1}\|, \\
e_1 &= (1, 0, ..., 0)^T \in \mathcal{R}^{j+1}, \\
\text{Minimize} & \quad \|\beta_j e_1 - H_j d^j\| \text{ over } \mathcal{R}^j \text{ to obtain } d^j, \\
\rho_{j+1} &= \|\beta_j e_1 - H_j d^j\|, \\
u^{j+1} &= u^j + V_j d^j.
\end{aligned}
$$

The GMRES method breaks down when $\|\mathcal{H}v^j - \sum_{i=1}^{j}((\mathcal{H}v^j)^T v^i)v^i\|$ is zero. This quantity is zero when the residual is zero. This is not a problem, since if the residual is zero the solution will be found. See [47, 66].

BiConjugate Gradient Method (BiCG)

Among all methods which do not require the matrix to be symmetric the GMRES method is the most successful Krylov subspace method in terms of minimization property. However, the operations and the storage requirement for this method increase linearly with the iteration number (GMRES is long recurrence method). The BiConjugate Gradient method is a short recurrence method and is used to solve nonsymmetric problem. It takes another approach: instead of minimizing the residual, the residual is required to satisfy the bi-orthogonality condition

$$(r^j)^T w = 0, \quad \forall w \in \bar{\mathcal{K}}_j; \quad \bar{\mathcal{K}}_j = span\{\hat{r}^0, \mathcal{H}^T\hat{r}^0, ..., (\mathcal{H}^T)^{j-1}\hat{r}^0\},$$

where $\bar{\mathcal{K}}_j$ is Krylov subspace for $\mathcal{H}^T$ and usually $\hat{r}^0$ is chosen such that $\hat{r}^0 = r^0$ [47].

The BiCG Algorithm:

- Given an initial solution u^0, $r^0 = q - \mathcal{H}u^0$, Choose $\hat{r}^0$ such that $\hat{r}^0 \neq 0$, $d^0 = r^0$ and $\hat{d}^0 = \hat{r}^0$.
- For $j = 0, 1, ...$

```
while $\|r^j\| > \epsilon$ do
 $\alpha_j = (r^j)^T \hat{r}^j / (d^j)^T \mathcal{H}^T \hat{d}^j$,
 $u^{j+1} = u^j + \alpha_j d^j$,
 $r^{j+1} = r^j - \alpha_j \mathcal{H} d^j$,   $\hat{r}^{j+1} = \hat{r}^j - \alpha_j \mathcal{H}^T \hat{d}^j$,
 $\beta_j = (r^{j+1})^T \hat{r}^{j+1} / (r^j)^T \hat{r}^j$,
 $d^{j+1} = r^{j+1} + \beta_j d^j$,   $\hat{d}^{j+1} = \hat{r}^{j+1} + \beta_j \hat{d}^j$.
```

The BiCG method breaks down when either $(\hat{r}^j)^T r^j = 0$ or $(d^j)^T \mathcal{H}^T \hat{d}^j = 0$. If these quantities are very small this method becomes unstable [47, 70].

MINRES and SYMMLQ Methods

MINRES and SYMMLQ methods are used to solve symmetric indefinite equation systems. The MINRES method minimizes the 2-norm of the residual, while the SYMMLQ method solves the projected system, but it does not minimize anything. It maintains the residual orthogonal to the previous residuals. See [62] for more detail. As the MINRES and the SYMMLQ methods are used to solve symmetric indefinite system, these methods should be preconditioned by a symmetric preconditioner, see [25, 62].

2.4.4 Null Space Methods

Null space methods can be used for solving saddle point problems like the augmented system (1.7).

Solving (1.7) is equivalent to solving the following two equations:

$$\begin{aligned} -\Theta^{-1}\Delta x + A^T\Delta y &= f, \\ A\Delta x \qquad &= g. \end{aligned} \tag{2.17}$$

Let us introduce Z the null space matrix. Z is a matrix belong to $\mathfrak{R}^{n\times(n-m)}$ and satisfies $AZ = 0$.

Null space method is described as follows

1. Find $\Delta\tilde{x}$ such that

$$A\Delta\tilde{x} = g.$$

2. Solve the system

$$Z^T\Theta^{-1}Zp = -Z^T(\Theta^{-1}\Delta\tilde{x} + f), \tag{2.18}$$

where Z is the null space matrix of the constraint matrix A.

3. Set the solution $(\Delta x^*, \Delta y^*)$ as the following:

$\Delta x^* = \Delta\tilde{x} + Zp$.

Δy^* is the solution of the system

$$AA^T\Delta y = A(f + \Theta^{-1}\Delta x^*).$$

See [8].

Let us explain this method. First, we multiply the first equation of (2.17) with Z^T, which gives

$$-Z^T\Theta^{-1}\Delta x + Z^TA^T\Delta y = Z^Tf.$$

This is equivalent to

$$-Z^T\Theta^{-1}\Delta x = Z^T f$$

because of $Z^T A^T = 0$.

Let us denote $\Delta x = \Delta\tilde{x} + Zp$, where $\Delta\tilde{x}$ is chosen such that $A\Delta\tilde{x} = g$, then the previous equation becomes

$$Z^T\Theta^{-1}Zp = -Z^T(\Theta^{-1}\Delta\tilde{x} + f),$$

which is equivalent to (2.18).

In order to find Δy^*, we substitute Δx^* is the first equation of (2.17) and then multiply it with A which gives $AA^T\Delta y = A(f + \Theta^{-1}\Delta x^*)$.

The null space method is an attractive approach when $n-m$ is small. The null space system (2.18) can be solved either directly or iteratively (see Subsection 1.2.1 and 1.2.2 above). In [19] the PCG method is used to solve the null space system (which is similar to (2.18) but for quadratic minimization problem).

In order to use the null space method we first have to compute the null space matrix Z. Let us assume A has full row rank. The matrix Z is given by

$$Z = \begin{bmatrix} -A_1^{-1}A_2 \\ I_{n-m} \end{bmatrix},$$

where the constraint matrix A is partitioned as $A = [A_1, A_2]$, where A_1 is a $m \times m$ nonsingular matrix. There are plenty of choices to construct the $m \times m$ nonsingular matrix A_1 see [8]. In order to save on computation time and storage, one should choose the sparsest null basis matrix A_1. The problem of finding the sparsest null basis is called the null space problem. This problem is NP hard [17], and there are many papers which propose (heuristic) approaches to solve it [8, 11, 17, 18, 63].

Chapter 3

The PCG Method for the Augmented System

We are dealing with large and sparse problems and we are looking for an iterative method from the Krylov-subspace family which can solve the augmented system (1.7) efficiently. As we have discussed in the previous chapter, there exists a wide range of iterative methods which can be used in this context The family of Krylov-subspace methods [47, 66, 72] enjoys a particularly good reputation among different iterative methods. Since we plan to solve large systems of equations, we prefer to use a short recurrence method rather than a long recurrence one. The full recurrence methods such as GMRES [67] occasionally do not manage to converge fast enough and become unacceptably expensive. Among the short recurrence methods we considered MINRES [62] and PCG [42, 66, 72]. Bearing in mind that, whichever method is used, preconditioning is necessary, we decided not to use MINRES because this method requires a symmetric positive definite preconditioner, a restriction we would like to avoid. Summing up, encouraged by recent analyses [51, 65] we will apply the preconditioned conjugate gradients (PCG) method directly to the indefinite system (1.7).

In the introduction section we explained fully why we chose to work with the augmented system (1.7). To summarise, the augmented system has better conditioning and has additional flexibility in exploiting sparsity compared to the normal equations. In addition, all preconditioners for the normal equations system have an equivalent for the augmented system, while the opposite is not true.

The results presented in this chapter have been the subject of joint work with Jacek Gondzio and Julian Hall [2].

3.1 Preconditioner

Choosing the preconditioner for a linear system plays a critical role in the convergence of the iterative solver. The issue of finding a preconditioner for the augmented system was investigated in many papers [9, 10, 16, 21, 22, 23, 34, 46, 60] to mention a few. Let $\mathcal{H}$ be the matrix of the augmented system which arises from IPMs for the linear, quadratic or nonlinear programming problems.

$$\mathcal{H} = \begin{bmatrix} H & A^T \\ A & 0 \end{bmatrix}, \tag{3.1}$$

where H is a $n \times n$ matrix.

Before presenting a list of preconditioners for augmented systems, we should mention first the characteristics of good preconditioner. The preconditioner is considered to be good if it satisfies the following features. The first one is that the preconditioner should be a good approximation to the original matrix $\mathcal{H}$. If preconditioner is approximately equal to the original matrix, then the preconditioned matrix $P^{-1}\mathcal{H}$ will be approximately equal to identity matrix. That makes it easy to solve the preconditioned system. The second feature is that the preconditioner should be relatively easy to compute. Since for most iterative methods, the preconditioner is computed at each iteration of interior point method. The third feature is that it should be relatively easy to solve an equation with preconditioner, namely it should be easy to solve $Pd = r$. Since, this system is required to be solved at each iterations of the iterative solver. The final feature is that the eigenvalues of the preconditioned matrix should be clustered (and the distinct eigenvalues of the preconditioned matrix should be as less as possible) and bounded away from zero. Because, the convergence of iterative solvers usually relates to the eigenvalues of the preconditioned system. For the PCG method, for instance, see Theorem 2.4.4.

It is very difficult to design a preconditioner satisfies the previous four features in the same time. Consequently one needs to make a balance among these features to design a high-quality preconditioner. That why there are huge number of studies tackle this issue. Below we discuss a few of recently developed preconditioners for (3.1). We also report theorems which show the behaviours of the eigenvalues of preconditioned matrices for some of these preconditioners, see Theorems 3.1.1, 3.1.2, 3.1.3 and 3.1.4. This information is important because it give an idea about the convergence of the preconditioned system.

In [9] Bergamaschi, Gondzio, Venturin and Zilli propose a preconditioner for the augmented system for the linear, quadratic or nonlinear programming problems. This preconditioner is defined as follow:

$$P = \begin{bmatrix} G & \tilde{A}^T \\ \tilde{A} & 0 \end{bmatrix},$$

where G is an invertible approximation of H, and $\tilde{A}$ is a sparse approximation of the Jacobian of constraints that is of matrix A. Let the error matrix $E = A - \tilde{A}$ have rank p, where $0 \leq p \leq m$. Let $\tilde{\sigma}$ be the smallest singular value of $\tilde{A}D^{-1/2}$ and e_Q and e_A be errors terms given as

$$e_Q = \|D^{-1/2}QD^{-1/2} - I\|, \quad e_A = \frac{\|ED^{-1/2}\|}{\tilde{\sigma}}.$$

The eigenvalues of the preconditioned matrix $P^{-1}\mathcal{H}$ are characterized by the following theorem [9, Theorem 2.1].

Theorem 3.1.1. *Assume A and $\tilde{A}$ have maximum rank. If the eigenvector is of the form $(0, y)^T$ then the eigenvalues of $P^{-1}\mathcal{H}$ are either one (with multiplicity at least $m - p$) or possibly complex and bounded by $|\epsilon| \leq e_A$. Corresponding to eigenvectors of the form $(x, y)^T$ with $x \neq 0$ the eigenvalues are*

1. *equal to one (with multiplicity at least $m - p$), or*

2. *real positive and bounded by*

$$\lambda_{\min}(D^{-1/2}QD^{-1/2}) \leq \lambda \leq \lambda_{\max}(D^{-1/2}QD^{-1/2}), \text{ or}$$

3. *complex, satisfying*

$$|\epsilon_R| \leq e_Q + e_A,$$
$$|\epsilon_I| \leq e_Q + e_A,$$

where $\epsilon = \epsilon_R + i\epsilon_I$.

In [21] the constraint matrix is partitioned into two matrices, such that $A = [A_1, A_2]$, where A_1 is nonsingular. Accordingly, the matrix H is partitioned as follows

$$H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}.$$

The preconditioner P is constructed by replacing H by G. Similarly G is partitioned into

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix}.$$

The following theorem describes the eigenvalues of the preconditioned matrix $P^{-1}\mathcal{H}$ [21, Theorem 2.1].

Theorem 3.1.2. *Suppose that Z is the null space matrix of A. Then $P^{-1}\mathcal{H}$ has $2m$ unit eigenvalues, and the remaining $n-m$ eigenvalues are those of the generalized eigenproblem*

$$Z^THZv = \lambda Z^TGZv.$$

Different choices of the matrices G_{11}, G_{12}, G_{21} and G_{22} give different preconditioners. For the symmetric case $H_{21} = H_{12}^T$, the authors proposed different choices of the matrix G, which improve the eigenvalues of the preconditioned matrix $P^{-1}\mathcal{H}$. Here we will mention a few of these preconditioners. By choosing

$$G_{22} = H_{22},\ G_{11} = 0 \text{ and } G_{21} = 0.$$

The eigenvalues of the preconditioned matrix are given in the following theorem [21, Theorem 2.3].

Theorem 3.1.3. *Suppose that the matrix G is chosen as mentioned before. Suppose that H_{22} is positive definite, and let*

$$\begin{aligned}\rho = \quad & \min\{rank(A_2), rank(H_{21})\} + \min\{rank(A_2), rank(H_{21}) \\ & + \min[rank(A_2), rank(H_{11})]\}.\end{aligned}$$

Then $P^{-1}\mathcal{H}$ has at most

$$\begin{aligned}rank(R^TH_{21}^T + H_{21}R + R^TH_{11}R) + 1 \quad & \leq \min(\rho, n-m) + 1 \\ & \leq \min(2m, n-m) + 1,\end{aligned}$$

distinct eigenvalues, where $R = -A_1^{-1}A_2$.

For $G_{22} = H_{22}$, $G_{11} = H_{11}$ and $G_{21} = 0$. The eigenvalues of the preconditioned matrix satisfy the following theorem [21, Theorem 2.4].

Theorem 3.1.4. *Suppose that the matrix G is chosen as mentioned before. Suppose that $H_{22} + R^TH_{11}^TR$ is positive definite, and that*

$$\nu = 2\min\{rank(A_2), rank(H_{21})\}.$$

Then $P^{-1}\mathcal{H}$ has at most $\nu + 1$ distinct eigenvalues, where

$$rank(R^TH_{11}R) + 1 \leq \nu + 1 \leq \min(2m, n-m) + 1.$$

In [34] the authors propose four different symmetric positive definite preconditioners for the augmented system for the linear programs. In order to construct these preconditioners the matrices H and A are partitioned as has been mentioned earlier. However, A_2 is chosen to be the nonsingular matrix instead of A_1.

The first preconditioner is a diagonal matrix. This preconditioner is given by

$$P = C_1 C_1^T = \begin{bmatrix} H_{11} & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}.$$

The preconditioned matrix is given by

$$C_1^{-1} \mathcal{H} C_1^{-T} = \begin{bmatrix} I & 0 & H_{11}^{-1/2} A_1^T \\ 0 & H_{22} & A_2^T \\ A_1 H_{11}^{-1/2} & A_2 & 0 \end{bmatrix}.$$

The second preconditioner is a block diagonal matrix. It is presented as follows

$$P = C_2 C_2^T = \begin{bmatrix} H_{11} & 0 & 0 \\ 0 & A_2^T A_2 & 0 \\ 0 & 0 & I \end{bmatrix}.$$

The preconditioned matrix is given by

$$C_2^{-1} \mathcal{H} C_2^{-T} = \begin{bmatrix} I & 0 & H_{11}^{-1/2} A_1^T \\ 0 & A_2^{-T} H_{22} A_2^{-1} & I \\ A_1 H_{11}^{-1/2} & I & 0 \end{bmatrix}.$$

The third preconditioner is designed to eliminate the submatrix $A_2^{-T} H_{22} A_2^{-1}$

in the previous preconditioned matrix. This preconditioner is given by

$$P = C_3C_3^T, \quad C_3 = \begin{bmatrix} H_{11}^{1/2} & 0 & 0 \\ 0 & A_2^T & \frac{1}{2}H_{22}A_2^{-1} \\ 0 & 0 & I \end{bmatrix}.$$

The preconditioned matrix is given by

$$C_3^{-1}\mathcal{H}C_3^{-T} = \begin{bmatrix} I & -\frac{1}{2}H_{11}^{-1/2}A_1^TA_2^{-T}H_{22}A_2^{-1} & H_{11}^{-1/2}A_1^T \\ -\frac{1}{2}A_2^{-T}H_{22}A_2^{-1}A_1H_{11}^{-1/2} & 0 & I \\ A_1H_{11}^{-1/2} & I & 0 \end{bmatrix}.$$

The fourth preconditioner also eliminates the submatrix $A_2^{-T}H_{22}A_2^{-1}$, using the factorization $A_2^T = LU$. This preconditioner is given by

$$P = C_4C_4^T, \quad C_4 = \begin{bmatrix} H_{11}^{1/2} & 0 & 0 \\ 0 & L & \frac{1}{2}H_{22}L^{-T} \\ 0 & 0 & U^T \end{bmatrix}.$$

The preconditioned matrix is given by

$$C_4^{-1}\mathcal{H}C_4^{-T} = \begin{bmatrix} I & -\frac{1}{2}H_{11}^{-1/2}A_1^TU^{-1}L^{-1}H_{22}L^{-T} & H_{11}^{-1/2}A_1^TU^{-1} \\ -\frac{1}{2}L^{-1}H_{22}L^{-T}U^{-T}A_1H_{11}^{-1/2} & 0 & I \\ U^{-T}A_1H_{11}^{-1/2} & I & 0 \end{bmatrix}.$$

The preconditioner in [60] is given in the form $P = CC^T$ and is applied from the left and from the right to the augmented system, which arises from the IPMs for LP. To construct this preconditioner the matrices A and H are partitioned as mentioned before, where A_1 is nonsingular. The inverse of C is given by

$$C^{-1} = \begin{bmatrix} -H^{-1/2} & M \\ T & 0 \end{bmatrix},$$

$T = [I \quad 0]Q$, where Q is a permutation matrix, and $M = T^T H_{11}^{1/2} A_1^{-1}$.

The preconditioned matrix is given by:

$$C^{-1}\mathcal{H}C^{-T} = Q \begin{bmatrix} -I & -W & 0 \\ -W^T & I & 0 \\ 0 & 0 & H_{11} \end{bmatrix} Q^T,$$

where $W = H_{11}^{1/2} A_1^{-1} A_2 H_{22}^{-1/2}$.

Assume $\Delta x = [\Delta x_1, \Delta x_2]$ is partitioned accordingly to the partition of A.

Eventually in the approach of [60] the preconditioned system is reduced to the following normal equations

$$(I + WW^T)\Delta x_1 = \tilde{g}.$$

In this section we construct a new preconditioner for the augmented system (1.7). And before we do so we will rearrange the augmented system such that

$$\begin{bmatrix} \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} -\Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}, \tag{3.2}$$

where in this chapter we redefine g as follows $g = Ax - b$.

To design the preconditioner for the augmented system, we first observe that the ill-conditioning in linear systems (1.7) and (1.8) is a consequence of the properties of the diagonal scaling matrix Θ. From the complementarity condition for linear programs we know that, at the optimum, $\hat{x}_j \hat{s}_j = 0, \forall j \in \{1, 2, \ldots, n\}$. The condition $\hat{x}_j \hat{s}_j = 0$ is satisfied if at least one of the variables $\hat{x}_j$ and $\hat{s}_j$ is zero. Primal-dual interior point methods identify a strong optimal partition [77], that is, they produce an optimal solution with the property $\hat{x}_j + \hat{s}_j > 0, \forall j$. In other words, only one of $\hat{x}_j$ and $\hat{s}_j$ is zero. The set of indices $\{1, 2, \ldots, n\}$ can therefore be partitioned into two disjoint subsets:

$$\mathcal{B} = \{j \in \{1, 2, ..., n\} : \hat{x}_j > 0\} \quad \text{and} \quad \mathcal{N} = \{j \in \{1, 2, ..., n\} : \hat{s}_j > 0\}.$$

In fact, the optimal partition is closely related (but not equivalent to) the basic-nonbasic partition in the simplex method. That is due to that simplex method iterations move from vertex to vertex until the optimal solution is found. So the simplex method has exactly m basic variables (variables belong to $\mathcal{B}$) and $n-m$ nonbasic variables (variables belong to $\mathcal{N}$). However, interior point methods approach the optimal solution by moving through the interior of the feasible region. Consequently, interior point methods have m basic variable and $n-m$ nonbasic variables in the limit only. That is why we refer to this partition in interior point methods by optimal partition.

Unlike the simplex method which satisfies the complementarity condition at each iteration, the interior point method satisfies this condition only in the limit. The primal-dual interior point method identifies a strong optimal partition near the optimal solution. Below we will summarise its asymptotic behaviour and use the arrow to denote “converges to”. If at the optimal solution $j \in \mathcal{B}$, then $x_j \to \hat{x}_j > 0$ and $s_j \to 0$, hence the corresponding element $\theta_j \to \infty$. If at the optimal solution $j \in \mathcal{N}$, then $x_j \to 0$ and

$s_j \rightarrow \hat{s}_j > 0$ and $\theta_j \rightarrow 0$. Summing up,

$$\theta_j \rightarrow \begin{cases} \infty, & \text{if } j \in \mathcal{B} \\ 0, & \text{if } j \in \mathcal{N}, \end{cases} \quad \text{and} \quad \theta_j^{-1} \rightarrow \begin{cases} 0, & \text{if } j \in \mathcal{B} \\ \infty, & \text{if } j \in \mathcal{N}. \end{cases} \tag{3.3}$$

This property of interior point methods is responsible for a number of numerical difficulties. In particular, it causes both linear systems (1.7) and (1.8) to become very ill-conditioned when an interior point method approaches the optimal solution [3]. However, it may be used to advantage when constructing a preconditioner for the iterative method.

We partition the matrices and vectors:

$$A = [A_{\mathcal{B}}, A_{\mathcal{N}}], \quad \Theta = \begin{bmatrix} \Theta_{\mathcal{B}} & 0 \\ 0 & \Theta_{\mathcal{N}} \end{bmatrix}, \quad x = [x_{\mathcal{B}}, x_{\mathcal{N}}], \quad \text{and} \quad s = [s_{\mathcal{B}}, s_{\mathcal{N}}]$$

according to the partition of $\{1, 2, \ldots, n\}$ into sets $\mathcal{B}$ and $\mathcal{N}$. With this notation, from (3.3) we conclude that $\Theta_{\mathcal{N}} \approx 0$ and $\Theta_{\mathcal{B}}^{-1} \approx 0$. Consequently, the matrix in the augmented system (3.2) can be approximated as follows:

$$\begin{bmatrix} \Theta_{\mathcal{B}}^{-1} & & A_{\mathcal{B}}^T \\ & \Theta_{\mathcal{N}}^{-1} & A_{\mathcal{N}}^T \\ A_{\mathcal{B}} & A_{\mathcal{N}} & \end{bmatrix} \approx \begin{bmatrix} & & A_{\mathcal{B}}^T \\ & \Theta_{\mathcal{N}}^{-1} & A_{\mathcal{N}}^T \\ A_{\mathcal{B}} & A_{\mathcal{N}} & \end{bmatrix}, \tag{3.4}$$

and the matrix in the normal equations system (1.8) can be approximated as follows:

$$A\Theta A^T = A_{\mathcal{B}}\Theta_{\mathcal{B}}A_{\mathcal{B}}^T + A_{\mathcal{N}}\Theta_{\mathcal{N}}A_{\mathcal{N}}^T \approx A_{\mathcal{B}}\Theta_{\mathcal{B}}A_{\mathcal{B}}^T. \tag{3.5}$$

If the matrix $A_{\mathcal{B}}$ was square and nonsingular then equations (3.4) and (3.5) would suggest obvious preconditioners for the augmented system and normal equations, respectively. However, there is no guarantee that this is the case. On the contrary, in practical applications it is very unlikely that the matrix $A_{\mathcal{B}}$ corresponding to the optimal partition is square and nonsingular. Moreover, the optimal partition is known only when an IPM approaches the optimal solution of the linear program.

To construct a preconditioner to (3.2) with a structure similar to the approximation (3.4) we need to guess an optimal partition and, additionally, guarantee that the matrix B which approximates $A_{\mathcal{B}}$ is nonsingular. We exploit the difference in magnitude of elements in Θ to design a preconditioner. We sort the elements of Θ in non-increasing order: $\theta_1 \geq \theta_2 \geq \theta_3 \geq \cdots \geq \theta_n$. Hence the elements of Θ^{-1} satisfy $\theta_1^{-1} \leq \theta_2^{-1} \leq \theta_3^{-1} \leq \cdots \leq \theta_n^{-1}$. If the primal-dual iterate is sufficiently close to an optimal solution, then the first elements θ_j^{-1} in this list correspond to variables x_j which are most likely to be nonzero at the optimum, and the last elements in the list correspond to variables which are likely to be zero at the optimum. We select the first m linearly independent columns of the matrix A, when permuted according to the order of θ_j^{-1}, and we construct a nonsingular matrix B from these columns. The submatrix of A corresponding to all the remaining columns is denoted by N. Therefore we assume that a partition $A = [B, N]$ is known such that B is nonsingular and the entries θ_j^{-1} corresponding to columns of B are chosen from the smallest elements of Θ^{-1}. According to this partitioning of A and Θ (and after a symmetric row and column permutation) the indefinite matrix in (3.2) can be rewritten in the following form

$$K = \begin{bmatrix} \Theta_B^{-1} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix}. \tag{3.6}$$

By construction, the elements of Θ_B^{-1} are supposed to be among the smallest elements of Θ^{-1}, hence we may assume that $\Theta_B^{-1} \approx 0$. The following easily invertible block-triangular matrix

$$P = \begin{bmatrix} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \tag{3.7}$$

is a good approximation to K. Hence P is an attractive preconditioner for K. We should mention that Oliveira and Sorensen [60] use a similar partitioning process to derive their preconditioner for the normal equations. They order the columns of the matrix $A\Theta^{-1}$ from the smallest to the largest with respect to the 1-norm and then scan the columns of A in this order to select the first m that are linearly independent.

Since the matrix B was constructed from columns corresponding to the smallest possible elements of Θ^{-1} we may expect that $\|\Theta_B^{-1}\|_F \ll \|\Theta_N^{-1}\|_F$, where $\|.\|_F$ denotes the Frobenius norm of the matrix. Using (3.6) and (3.7) we derive the following bound on the square of the Frobenius norm of the difference of matrices K and P:

$$\|K - P\|_F^2 = \|\Theta_B^{-1}\|_F^2 \ll \|P\|_F^2 < \|K\|_F^2. \tag{3.8}$$

Summing up, P is a good approximation to K (since the approximation error is small in relation to $\|P\|_F^2$ and $\|K\|_F^2$) and we may consider it as a possible preconditioner of K. Secondly, it is easy to compute P, we order the elements of Θ in non-increase order then we pick the first m linearly independent columns of A in this order to construct the nonsingular matrix B, see Subsection 3.4. In addition, it is easy to solve an equation with P

because it is block-triangular with nonsingular diagonal blocks B, Θ_N^{-1} and B^T. We conclude this section by giving explicit formulae for the solution of equations with the preconditioner (3.7) and leave the analysis of spectral properties of the preconditioned matrix $P^{-1}K$ to Subsection 3.2.

3.1.1 Solving equations with P

The matrix (3.7) is block triangular and its diagonal blocks B, Θ_N^{-1} and B^T are invertible. Let $d = [d_B,\, d_N,\, d_y]$ and $r = [r_B,\, r_N,\, r_y]$ and consider the system of equations

$$\begin{bmatrix} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \begin{bmatrix} d_B \\ d_N \\ d_y \end{bmatrix} = \begin{bmatrix} r_B \\ r_N \\ r_y \end{bmatrix}. \tag{3.9}$$

The solution of (3.9) can easily be computed by exploiting the block-triangular structure of the matrix:

$$\begin{aligned} B^T d_y = r_B \quad &\Rightarrow \quad d_y = B^{-T} r_B \\ \Theta_N^{-1} d_N + N^T d_y = r_N \quad &\Rightarrow \quad d_N = \Theta_N r_N - \Theta_N N^T d_y \\ B d_B + N d_N = r_y \quad &\Rightarrow \quad d_B = B^{-1}(r_y - N d_N). \end{aligned} \tag{3.10}$$

The operation $d = P^{-1}r$ involves solving two equations (one with B and one with B^T) and a couple of matrix-vector multiplications. These operations will be performed at every iteration of the conjugate gradients procedure hence they should be implemented in the most efficient way. The issues of choosing a well-conditioned basis matrix B with sparse factored inverse are addressed in Subsection 3.4.

3.2 Spectral analysis

We have observed earlier that if Θ_B is chosen carefully and $\|\Theta_B^{-1}\|_F \ll \|\Theta_N^{-1}\|_F$ then the preconditioner (3.7) is a good approximation to K in (3.6). To assess the quality of the preconditioner we need a better understanding of the relation between P and K.

We will therefore analyse the spectral properties of the preconditioned matrix $P^{-1}K$. Let us use the notation $Kt = q$ to denote the system (3.2), where $t = [-\Delta x, \Delta y]$ and $q = [f, g]$. Given a starting approximation $t^{(0)}$ and the associated residual $r^{(0)} = q - Kt^{(0)}$ the indefinite preconditioner may be applied either from the right, yielding the system

$$KP^{-1}\hat{t} = q, \qquad t = P^{-1}\hat{t}, \tag{3.11}$$

or from the left, so that the system to be solved becomes

$$P^{-1}Kt = P^{-1}q. \tag{3.12}$$

The right and the left preconditioned matrices KP^{-1} and $P^{-1}K$ have the same eigenvalues so general spectral results can be given in terms of either of the two formulations. The following theorem shows that the eigenvalues of the $P^{-1}K$ matrix are real and positive. Moreover they are bounded away from zero.

Theorem 3.2.1. *Let λ be an eigenvalue of $P^{-1}K$. Then λ is real and $\lambda \geq 1$.*

Proof. Let v be an eigenvector of $P^{-1}K$ corresponding to the eigenvalue λ, that is, $P^{-1}Kv = \lambda v$. Let $\lambda = 1 + \tau$ and, applying the usual partitioning

$v = [v_B, v_N, v_y]$, the eigensystem can be written as $Kv = (1+\tau)Pv$:

$$\begin{bmatrix} \Theta_B^{-1} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \begin{bmatrix} v_B \\ v_N \\ v_y \end{bmatrix} = (1+\tau) \begin{bmatrix} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \begin{bmatrix} v_B \\ v_N \\ v_y \end{bmatrix}$$

which yields

$$\begin{aligned} \Theta_B^{-1} v_B &= \tau B^T v_y \\ \tau(\Theta_N^{-1} v_N + N^T v_y) &= 0 \\ \tau(B v_B + N v_N) &= 0. \end{aligned}$$

We consider two cases. When $\tau = 0$ clearly $\lambda = 1$ so the claim is true. Otherwise, when $\tau \neq 0$, the equation system can be simplified:

$$\begin{aligned} \Theta_B^{-1} v_B &= \tau B^T v_y \\ \Theta_N^{-1} v_N + N^T v_y &= 0 \\ B v_B + N v_N &= 0, \end{aligned}$$

and solved for τ. From the third equation we get $v_B = -B^{-1} N v_N$ and, substituting this in the first equation, yields $N v_N = -\tau B \Theta_B B^T v_y$. Next, we use the second equation to substitute for $v_N = -\Theta_N N^T v_y$ giving

$$(N \Theta_N N^T) v_y = \tau (B \Theta_B B^T) v_y.$$

If $v_y = 0$ then (using $\tau \neq 0$) we deduce that $v_B = 0$ and $v_N = 0$, that is the eigenvector is zero. We can exclude such a situation and safely assume that $v_y \neq 0$. In this case, we multiply both sides of the equation by v_y^T to get

$$v_y^T(N\Theta_N N^T)v_y = \tau v_y^T(B\Theta_B B^T)v_y.$$

Since all the elements of Θ are positive and B is nonsingular, the matrix $B\Theta_B B^T$ is symmetric positive definite and the matrix $N\Theta_N N^T$ is symmetric positive semidefinite. Hence we conclude that

$$\tau = \frac{v_y^T(N\Theta_N N^T)v_y}{v_y^T(B\Theta_B B^T)v_y} \geq 0, \tag{3.13}$$

which is real and positive number, which completes the proof. □

The proof reveals the importance of the correct partitioning of $A = [B, N]$. Indeed, this partition should have a number of desirable features:

- B should be nonsingular and well-conditioned since we should operate accurately with the preconditioner;
- All elements in Θ_B^{-1} should be small in comparison with those in Θ_N^{-1}.

The condition $\|\Theta_B^{-1}\|_F \ll \|\Theta_N^{-1}\|_F$ is relatively easy to satisfy. However, (3.13) indicates that we need a stronger property: we would like to bound τ from above and, in that way, cluster all eigenvalues of $P^{-1}K$ in an interval $[1, \lambda_{max}]$, with λ_{max} kept as small as possible. This opens questions regarding the necessary concessions to be made when the matrix B and the corresponding Θ_B are chosen. The ability to identify a well-conditioned matrix B consisting of columns for which the θ_j are "large" is crucial for the good/efficient behaviour of our approach. We discuss these issues in detail in Section 3.4.

In the previous theorem we show that the eigenvalues of the preconditioned matrix KP^{-1} are real and greater than one. In the following theorem we show that the preconditioned matrix KP^{-1} has at least $n+p$ unit eigenvalues, where p is the rank of the matrix N.

Theorem 3.2.2. *The preconditioned matrix KP^{-1} has at least $n+p$ unit eigenvalues.*

Proof. The inverse of the preconditioner P is given by

$$P^{-1} = \begin{bmatrix} B^{-1}N\Theta_N N^T B^{-T} & -B^{-1}N\Theta_N & B^{-1} \\ -\Theta_N N^T B^{-T} & \Theta_N & 0 \\ B^{-T} & 0 & 0 \end{bmatrix}.$$

Therefore, the preconditioned matrix KP^{-1} is given by

$$KP^{-1} = \begin{bmatrix} I+\Theta_B^{-1}B^{-1}N\Theta_N N^T B^{-T} & -\Theta_B^{-1}B^{-1}N\Theta_N & \Theta_B^{-1}B^{-1} \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}.$$

Let v be an eigenvector of KP^{-1} corresponding to the eigenvalue λ, that is, $KP^{-1}v = \lambda v$, which can be rewritten as

$$\begin{bmatrix} I+\Theta_B^{-1}B^{-1}N\Theta_N N^T B^{-T} & -\Theta_B^{-1}B^{-1}N\Theta_N & \Theta_B^{-1}B^{-1} \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} v_B \\ v_N \\ v_y \end{bmatrix} = \lambda \begin{bmatrix} v_B \\ v_N \\ v_y \end{bmatrix}$$

which yields

$$(I+\Theta_B^{-1}B^{-1}N\Theta_N N^T B^{-T})v_B - \Theta_B^{-1}B^{-1}N\Theta_N v_N + \Theta_B^{-1}B^{-1}v_y = \lambda v_B, \quad (3.14)$$

$$v_N = \lambda v_N, \quad (3.15)$$

$$v_y = \lambda v_y. \quad (3.16)$$

We now analyse a number of cases depending on v_B, v_N and v_y.

1. $v_B = 0$. Substituting this in (3.14) gives $v_y = N\Theta_N v_N$. That gives the eigenvector $[0,\, v_N,\, N\Theta_N v_N]$ which is associated with the unit eigenvalue with multiplicity $n-m$, because we can find $n-m$ linearly independent vectors v_N.

2. $v_B \neq 0$. Then there is a nonzero vector z such that $v_B = B^T z$. $z \neq 0$ because $v_B \neq 0$ and B is nonsingular. By substituting this in (3.14) we get the following equality

$$B\Theta_B B^T z + N\Theta_N N^T z - N\Theta_N v_N + v_y = \lambda B\Theta_B B^T z. \tag{3.17}$$

We have the following cases:

(a) $v_y = 0$ and $v_N \neq 0$. If we choose v_N such that $v_N = N^T z$, we will get the eigenvector $[B^T z,\, N^T z,\, 0]$ which is associated with the unit eigenvalue with multiplicity m, because we can find m linearly independent vector v_B.

(b) $v_N = 0$ and $v_y \neq 0$. We can choose $v_y = -N\Theta_N N^T z$. That gives the eigenvector $[B^T z,\, 0,\, -N\Theta_N N^T z]$ which is associated with the unit eigenvalue with multiplicity p, where N has rank p. For the reason that we can find p linearly independent vector $-N\Theta_N N^T z$, because the vector z is nonzero and N has rank p.

(c) $v_N \neq 0$ and $v_y \neq 0$. Let us choose v_N such that

$$v_N = (\epsilon I + N\Theta_N)^{-1} v_y, \tag{3.18}$$

where $\epsilon > 0$ and it is chosen such that the matrix $(\epsilon I + N\Theta_N)$ is nonsingular. $(\epsilon I + N\Theta_N)$ is nonsingular for any $\epsilon \in (0, \delta)$, where

$$\delta = \min_i \{|\lambda_i| : \lambda_i \neq 0\}, \text{ where } \lambda_i \text{ is all eigenvalues of } N\Theta_N.$$

By substituting (3.18) in (3.17), we get

$$B\Theta_B B^T z + N\Theta_N N^T z + \epsilon v_N = \lambda B\Theta_B B^T z.$$

We can choose z any nonzero vector such that $z^T v_N \geq 0$. Since $z \neq 0$, $B\Theta_B B^T$ is symmetric positive definite and $N\Theta_N N^T$ is symmetric positive semidefinite, we can write

$$\lambda = 1 + \frac{z^T N\Theta_N N^T z + \epsilon z^T v_N}{z^T B\Theta_B B^T z}. \tag{3.19}$$

The remaining $m - p$ eigenvectors are $[v_B,\ v_N,\ v_y]$ which is associated with the eigenvalues (3.19), because we can find $m - p$ linearly independent vectors v_y, which are linearly independent from v_y in case (b).

We conclude from the previous cases that the preconditioned matrix KP^{-1} has at least $n + p$ unit eigenvalues. □

3.3 The PCG method for nonsymmetric indefinite system

Rozlozník and Simoncini [65] used the BiCG method to solve an indefinite system such as (3.2) preconditioned from the right. They show that the right preconditioned BiCG method reduces to the standard preconditioned

CG method if the following two properties hold. The first property is that the preconditioned matrix $H = KP^{-1}$ is J-symmetric, where $J = P^{-1}$, and the second is that $g = 0$. The reason behind this is that when $g = 0$ the residual of PCG has a zero block and can be expressed as $r^j = [r_1^j,\, 0]$. Although in our case $g \neq 0$, the initial iterate t^0 can be chosen so that the corresponding residual has the form $r^0 = [r_1^0,\, 0]$. Furthermore, the preconditioned matrix $H = KP^{-1}$ is J-symmetric, since $H^TJ = JH$. See [65].

Let us consider the following starting point for CG:

$$t^0 = \begin{bmatrix} -\Delta x_B^0 \\ -\Delta x_N^0 \\ \Delta y^0 \end{bmatrix} = \begin{bmatrix} B^{-1}g \\ 0 \\ 0 \end{bmatrix}, \tag{3.20}$$

where $\Delta x = \begin{bmatrix} \Delta x_B \\ \Delta x_N \end{bmatrix}$. The initial residual $r^0 = q - Kt^0$ may then be written as

$$r^0 = \begin{bmatrix} f_B \\ f_N \\ g \end{bmatrix} - \begin{bmatrix} \Theta_B^{-1} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \begin{bmatrix} B^{-1}g \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} f_B - \Theta_B^{-1}B^{-1}g \\ f_N \\ 0 \end{bmatrix}.$$

Note two interesting properties of the preconditioned matrix KP^{-1} stated as two Lemmas below. Multiplying by the preconditioned matrix KP^{-1} preserves a zero block in the third component of the vector.

Lemma 3.3.1. *Let* $t = \begin{bmatrix} v_B \\ v_N \\ 0 \end{bmatrix}$. *Then* $KP^{-1}t = \begin{bmatrix} z_B \\ z_N \\ 0 \end{bmatrix}$.

Proof. We note first that, by using (3.9)-(3.10), we may write $u = P^{-1}t$ as

$$u = \begin{bmatrix} B^{-1}N\Theta_N N^T B^{-T} v_B - B^{-1}N\Theta_N v_N \\ -\Theta_N N^T B^{-T} v_B + \Theta_N v_N \\ B^{-T} v_B \end{bmatrix}.$$

Hence

$$\begin{aligned} KP^{-1}t &= Ku = \begin{bmatrix} \Theta_B^{-1} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \begin{bmatrix} B^{-1}N\Theta_N N^T B^{-T} v_B - B^{-1}N\Theta_N v_N \\ -\Theta_N N^T B^{-T} v_B + \Theta_N v_N \\ B^{-T} v_B \end{bmatrix} \\ &= \begin{bmatrix} (I + \Theta_B^{-1} B^{-1} N\Theta_N N^T B^{-T}) v_B - \Theta_B^{-1} B^{-1} N\Theta_N v_N \\ v_N \\ 0 \end{bmatrix}, \end{aligned}$$

which completes the proof. □

Furthermore, using the initial approximate solution

$$t^0 = \begin{bmatrix} -\Delta x_B^0 \\ -\Delta x_N^0 \\ \Delta y^0 \end{bmatrix} = \begin{bmatrix} B^{-1}(g - N\Theta_N f_N) \\ \Theta_N f_N \\ 0 \end{bmatrix}, \qquad (3.21)$$

the residuals will have two zero blocks, $r = \begin{bmatrix} r_B \\ 0 \\ 0 \end{bmatrix}$.

The initial residual $r^0 = q - Kt^0$ may be written:

$$r^0 = \begin{bmatrix} f_B \\ f_N \\ g \end{bmatrix} - \begin{bmatrix} \Theta_B^{-1} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \begin{bmatrix} B^{-1}(g - N\Theta_N f_N) \\ \Theta_N f_N \\ 0 \end{bmatrix},$$

which gives

$$r^0 = \begin{bmatrix} f_B - \Theta_B^{-1}B^{-1}g + \Theta_B^{-1}B^{-1}N\Theta_N f_N \\ 0 \\ 0 \end{bmatrix}.$$

We observe an important property of the preconditioned matrix: multiplying with the matrix KP^{-1} preserves the zero blocks in the second and third components of the vector.

Lemma 3.3.2. *Let* $t = \begin{bmatrix} v_B \\ 0 \\ 0 \end{bmatrix}$. *Then* $KP^{-1}t = \begin{bmatrix} z_B \\ 0 \\ 0 \end{bmatrix}$.

Proof. We note first that, by using (3.9)-(3.10), we may write $u = P^{-1}t$ as

$$u = \begin{bmatrix} B^{-1}N\Theta_N N^T B^{-T} v_B \\ -\Theta_N N^T B^{-T} v_B \\ B^{-T} v_B \end{bmatrix}$$

hence

$$KP^{-1}t = Ku = \begin{bmatrix} \Theta_B^{-1} & & B^T \\ & \Theta_N^{-1} & N^T \\ B & N & \end{bmatrix} \begin{bmatrix} B^{-1}N\Theta_N N^T B^{-T} v_B \\ -\Theta_N N^T B^{-T} v_B \\ B^{-T} v_B \end{bmatrix},$$

we obtain

$$KP^{-1}t = \begin{bmatrix} (I + \Theta_B^{-1} B^{-1} N \Theta_N N^T B^{-T}) v_B \\ 0 \\ 0 \end{bmatrix},$$

which completes the proof. □

From the PCG algorithm, we have $d^0 = P^{-1}r^0$, $d^j = P^{-1}r^j + \beta_j d^{j-1}$ and $r^{j+1} = r^j - \alpha_j K d^j$. So the residual r^1 is computed as linear combination of r^0 and $KP^{-1}r^0$. For $j > 1$, the residual r^{j+1} is computed as a linear combination of r^{j-1}, r^j and $KP^{-1}r^j$ (That is because $r^{j+1} = \alpha_j\beta_j/\alpha_{j-1} r^{j-1} + (1 - \alpha_j\beta_j/\alpha_{j-1})r^j - \alpha_j K P^{-1} r^j$). This implies that $r^j = [r_1^j, 0]$ for $j = 0,1, \ldots$ Consequently, we can use the standard PCG method along with (3.7) to solve (3.2).

3.3.1 The convergence of the PCG method

In this section, we analyse the behaviour of the PCG method for the indefinite system (3.2) and give explicit formulae describing the convergence of the method. The convergence analysis of the PCG method is important because both K and P are indefinite matrices. In [65] the authors prove that both the error and the residual of PCG method converge to zero. In here we prove that too. We analyse the method working in our specific setup with a particular starting point guaranteeing that the initial residual has the form $r^0 = [r_B^0, 0, 0]$.

The PCG algorithm (see Chapter 2) generates iterates $t^j, j = 0, 1, \ldots$ with residuals $r^j = q - Kt^j$. The error corresponding to each PCG iteration has the form $e^j = t^j - t^*$, where t^* is the solution of (3.2), and the residual can be written as $r^j = -Ke^j$ since $Ke^j = Kt^j - Kt^* = -r^j$. In Lemma 3.3.3 we prove that the indefinite K-inner product of the error e^j in the PCG algorithm is always non-negative so we can write $\|e^j\|_K = \sqrt{< e^j, Ke^j >}$, even though K is not positive definite. In Theorem 3.3.4 we show that the K-norm of the error e^j is minimized over the eigenvalues of the symmetric positive definite matrices. Similarly, in Theorem 3.3.5 we show that the Euclidean norm of the residual r^j is also minimized over the eigenvalues of the symmetric positive definite matrices. In other words, the error and residual terms display asymptotic convergence similar to that observed when PCG is applied to symmetric positive definite systems.

Lemma 3.3.3. *Assume we use (3.20) or (3.21) as initial solution of PCG method. Then the indefinite K-inner product $< e^j, Ke^j >$ is non-negative for the error e^j hence it defines a seminorm*

$$\|e^j\|_K = \sqrt{< e^j, Ke^j >} = \|e_1^j\|_{\Theta^{-1}}. \tag{3.22}$$

Proof. We have shown in Lemmas 3.3.1 and 3.3.2 that, for a suitable initial solution, the residual has the form $r^j = [r_1^j, 0]$. Hence

$$r^j = -Ke^j = -\begin{bmatrix} \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} e_1^j \\ e_2^j \end{bmatrix} = \begin{bmatrix} -\Theta^{-1}e_1^j - A^T e_2^j \\ -Ae_1^j \end{bmatrix},$$

implies $Ae_1^j = 0$. Simple calculations give the following result

$$\begin{aligned}
< e^j, Ke^j > &= (e^j)^T K e^j = \begin{bmatrix} (e_1^j)^T & (e_2^j)^T \end{bmatrix} \begin{bmatrix} \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} e_1^j \\ e_2^j \end{bmatrix} \\
&= (e_1^j)^T \Theta^{-1} e_1^j + (e_1^j)^T A^T e_2^j + (e_2^j)^T A e_1^j \\
&= (e_1^j)^T \Theta^{-1} e_1^j \\
&= (e_B^j)^T \Theta_B^{-1} e_B^j + (e_N^j)^T \Theta_N^{-1} e_N^j \geq 0 \qquad (3.23)
\end{aligned}$$

because Θ^{-1} is positive definite. This gives $\|e^j\|_K = \|e_1^j\|_{\Theta^{-1}}$, which completes the proof. □

Let D_j be the Krylov subspace $D_j = span\{d^0, d^1, ..., d^{j-1}\}$. Then $D_1 = span\{d^0\} = span\{P^{-1}r^0\}$. $D_2 = span\{d^0, d^1\}$, where the direction d^1 is a linear combination of the previous direction and $P^{-1}r^1$, while r^1 is a linear combination of the previous residual and Kd^0. This implies that d^1 is a linear combination of d^0 and $P^{-1}KP^{-1}r^0$, which gives $D_2 = span\{P^{-1}r^0, P^{-1}KP^{-1}r^0\}$. By the same argument d^{j-1} is a linear combination of d^{j-2} and $(P^{-1}K)^{j-1}P^{-1}r^0$, giving $D_j = span\{P^{-1}r^0, P^{-1}KP^{-1}r^0, ..., (P^{-1}K)^{j-1}P^{-1}r^0\}$. Moreover, $r^0 = -Ke^0$, so $D_j = span\{P^{-1}Ke^0, (P^{-1}K)^2e^0, \ldots, (P^{-1}K)^j e^0\}$.

The error can be written as $e^j = e^{j-1} + \alpha_{j-1}d^{j-1}$, hence $e^j = e^0 + \sum_{k=0}^{j-1} \alpha_k d^k$. Since $d^j \in D_{j+1}$ the error can be written as $e^j = (I + \sum_{k=1}^{j} \psi_k (P^{-1}K)^k)e^0$, where the coefficient ψ_k is related to α_k and β_k. Hence the error term can be expressed as

$$e^j = \phi_j(P^{-1}K)e^0, \qquad (3.24)$$

where ϕ_j is a polynomial of degree j and we require that $\phi_j(0) = 1$.

Theorem 3.3.4. *Let $e^{(0)}$ be the initial error of PCG. Then*

$$\|e^j\|_K^2 \leq \min_{\phi\in P_j,\phi(0)=1} \max_{\lambda\in\Lambda(I_m+WW^T)} [\phi(\lambda)]^2 \|e_B^0\|^2_{\Theta_B^{-1}} + \min_{\phi\in P_j,\phi(0)=1} \max_{\lambda\in\Lambda(I_{n-m}+W^TW)} [\phi(\lambda)]^2 \|e_N^0\|^2_{\Theta_N^{-1}}, \tag{3.25}$$

where P_j is the set of polynomials of degree j, $\Lambda(G)$ is the set of eigenvalues of the matrix G and $W = \Theta_B^{-1/2}B^{-1}N\Theta_N^{1/2}$. $I_m + WW^T$ and $I_{n-m} + W^TW$ are symmetric positive definite matrices.

Proof. First, we observe that $Ae_1^0 = 0$, that is $Be_B^0 + Ne_N^0 = 0$, and hence we write

$$Ke^0 = \begin{bmatrix} \Theta_B^{-1}e_B^0 + B^Te_2^0 \\ \Theta_N^{-1}e_N^0 + N^Te_2^0 \\ 0 \end{bmatrix}$$

and, using (3.10), we get

$$P^{-1}Ke^0 = \begin{bmatrix} B^{-1}N\Theta_NN^TB^{-T}\Theta_B^{-1}e_B^0 - B^{-1}Ne_N^0 \\ -\Theta_NN^TB^{-T}\Theta_B^{-1}e_B^0 + e_N^0 \\ B^{-T}\Theta_B^{-1}e_B^0 + e_2^0 \end{bmatrix}.$$

Since $Be_B^0 + Ne_N^0 = 0$, that is $e_B^0 = -B^{-1}Ne_N^0$ and $Ne_N^0 = -Be_B^0$, we obtain

$$\begin{aligned} P^{-1}Ke^0 &= \begin{bmatrix} B^{-1}N\Theta_NN^TB^{-T}\Theta_B^{-1}e_B^0 - B^{-1}(-Be_B^0) \\ -\Theta_NN^TB^{-T}\Theta_B^{-1}(-B^{-1}Ne_N^0) + e_N^0 \\ B^{-T}\Theta_B^{-1}e_B^0 + e_2^0 \end{bmatrix} \\ &= \begin{bmatrix} \Theta_B(\Theta_B^{-1} + \Theta_B^{-1}B^{-1}N\Theta_NN^TB^{-T}\Theta_B^{-1})e_B^0 \\ \Theta_N(\Theta_N^{-1} + N^TB^{-T}\Theta_B^{-1}B^{-1}N)e_N^0 \\ B^{-T}\Theta_B^{-1}e_B^0 + e_2^0 \end{bmatrix}. \end{aligned} \tag{3.26}$$

Let us define

$$C_1 = \Theta_B^{-1} + \Theta_B^{-1} B^{-1} N \Theta_N N^T B^{-T} \Theta_B^{-1} \text{ and } C_2 = \Theta_N^{-1} + N^T B^{-T} \Theta_B^{-1} B^{-1} N.$$

It is easy to prove that C_1 and C_2 are symmetric and positive definite matrices. By repeating a similar argument to the one used to derive (3.26) we obtain

$$\phi(P^{-1}K)e^0 = \begin{bmatrix} \phi(\Theta_B C_1)e_B^0 \\ \phi(\Theta_N C_2)e_N^0 \\ * \end{bmatrix}. \tag{3.27}$$

We observe that it is not necessary to compute the last component of the vector $P^{-1}Ke^0$ because Lemma 3.3.3 guarantees that this component does not contribute to $\|e^j\|_K^2$.

Using (3.27) to compute the K-norm of the error (3.23) we obtain

$$\|\phi_j(P^{-1}K)e^0\|_K^2 \;\; = \;\; \|\phi_j(\Theta_B C_1)e_B^0\|_{\Theta_B^{-1}}^2 + \|\phi_j(\Theta_N C_2)e_N^0\|_{\Theta_N^{-1}}^2. \tag{3.28}$$

Let us observe that

$$(\Theta_B C_1)^k = \Theta_B^{1/2}(\Theta_B^{1/2} C_1 \Theta_B^{1/2})^k \Theta_B^{-1/2} = \Theta_B^{1/2}(I_m + WW^T)^k \Theta_B^{-1/2},$$

where $(I_m + WW^T)$ is a symmetric and positive definite matrix.

Analogously, we observe that $(\Theta_N C_2)^k = \Theta_N^{1/2}(I_{n-m} + W^T W)^k \Theta_N^{-1/2}$, also $(I_{n-m} + W^T W)$ is a symmetric and positive definite matrix. Using these facts, the two terms on the right-hand-side of (3.28) can be simplified as follows

$$\begin{aligned}
\|\phi_j(\Theta_B C_1)e_B^0\|^2_{\Theta_B^{-1}} &= \|\Theta_B^{1/2}\phi_j(I_m + WW^T)\Theta_B^{-1/2}e_B^0\|^2_{\Theta_B^{-1}} \\
&= \|\phi_j(I_m + WW^T)\Theta_B^{-1/2}e_B^0\|^2, \\
\|\phi_j(\Theta_N C_2)e_N^0\|^2_{\Theta_N^{-1}} &= \|\Theta_N^{1/2}\phi_j(I_{n-m} + W^TW)\Theta_N^{-1/2}e_N^0\|^2_{\Theta_N^{-1}} \\
&= \|\phi_j(I_{n-m} + W^TW)\Theta_N^{-1/2}e_N^0\|^2,
\end{aligned}$$

From (3.24) we have $\|e^j\|_K^2 = \|\phi_j(P^{-1}K)e^0\|_K^2$, where ϕ_j is a polynomial of degree j and $\phi_j(0) = 1$. So the K-norm error in (3.28) becomes

$$\|e^j\|_K^2 = \|\phi_j(I_m + WW^T)\Theta_B^{-1/2}e_B^0\|^2 + \|\phi_j(I_{n-m} + W^TW)\Theta_N^{-1/2}e_N^0\|^2. \tag{3.29}$$

That is for every polynomial ϕ_j over the set of eigenvalues of $I_m + WW^T$ and $I_{n-m} + W^TW$. Consequently, we can write

$$\begin{aligned}
\|e^j\|_K^2 \leq &\min_{\phi\in P_j,\phi(0)=1} \max_{\lambda\in\Lambda(I_m+WW^T)} [\phi(\lambda)]^2 \|\Theta_B^{-1/2}e_B^0\|^2 \\
+ &\min_{\phi\in P_j,\phi(0)=1} \max_{\lambda\in\Lambda(I_{n-m}+W^TW)} [\phi(\lambda)]^2 \|\Theta_N^{-1/2}e_N^0\|^2,
\end{aligned}$$

and the claim is proved after substituting $\|\Theta_B^{-1/2}e_B^0\|^2 = \|e_B^0\|^2_{\Theta_B^{-1}}$ and $\|\Theta_N^{-1/2}e_N^0\|^2 = \|e_N^0\|^2_{\Theta_N^{-1}}$.

□

The K-norm of the error $e^j = \phi_j(P^{-1}K)e^0$ is minimized over the eigenvalues of the symmetric positive definite matrices $(I_m + WW^T)$ and $(I_{n-m} + W^TW)$ so the error term behaves similar to the symmetric positive definite case.

The Euclidean norm of the residual is minimized over the eigenvalues of the symmetric positive definite matrix $I_m + WW^T$. The following Theorem shows that the residual term displays asymptotic convergence similar to that observed when PCG is applied to positive definite system.

Theorem 3.3.5. *The residual of the PCG method which is used to solve the augmented system (1.7) preconditioned by P satisfies*

$$\|r^j\| \leq \min_{\phi \in P_j, \phi(0)=1} \max_{\lambda \in \Lambda(I_m + WW^T)} |\phi(\lambda)| \|r_B^0\|. \tag{3.30}$$

Proof. The residual satisfies

$$r^j = -Ke^j,$$

and the error can be written as

$$e^j = \phi_j(P^{-1}K)e^0.$$

So we can write the residual as

$$r^j = -K\phi_j(P^{-1}K)e^0 = -\phi_j(KP^{-1})Ke^0 = \phi_j(KP^{-1})r^0.$$

Furthermore,

$$KP^{-1}r^0 = \begin{bmatrix} (I + \Theta_B^{-1}B^{-1}N\Theta_N N^T B^{-T})r_B^0 - \Theta_B^{-1}B^{-1}N\Theta_N r_N^0 + \Theta_B^{-1}B^{-1}r_2^0 \\ r_N^0 \\ r_2^0 \end{bmatrix},$$

where $r^j = [r_B^j, r_N^j, r_2^j]$. The initial residual has the form $r^0 = [r_B^0, 0, 0]$

because of using the starting point (3.21), so the previous equation becomes

$$KP^{-1}r^0 = \begin{bmatrix} \Theta_B^{-1}(\Theta_B + B^{-1}N\Theta_N N^T B^{-T})r_B^0 \\ 0 \\ 0 \end{bmatrix}. \tag{3.31}$$

Let us define $C = \Theta_B + B^{-1}N\Theta_N N^T B^{-T}$. It is easy to prove that C is a symmetric positive definite matrix. By repeating a similar argument to one used to derive (3.31) we obtain

$$r^j = \phi_j(KP^{-1})r^0 = \begin{bmatrix} \phi_j(\Theta_B^{-1}C)r_B^0 \\ 0 \\ 0 \end{bmatrix}, \tag{3.32}$$

and so

$$\|r^j\| = \|\phi_j(\Theta_B^{-1}C)r_B^0\|. \tag{3.33}$$

Let us observe that $(\Theta_B^{-1}C)^k = \Theta_B^{-1/2}(\Theta_B^{-1/2}C\Theta_B^{-1/2})^k\Theta_B^{1/2} = \Theta_B^{-1/2}(I_m + WW^T)^k\Theta_B^{1/2}$, where
$I_m + WW^T$ is a symmetric positive definite matrix.

Using these definitions, (3.33) can be written as

$$\|r^j\| = \|\Theta_B^{-1/2}\phi_j(I_m + WW^T)\Theta_B^{1/2}r_B^0\| = \|\phi_j(I_m + WW^T)\Theta_B^{1/2}r_B^0\|_{\Theta_B^{-1}}.$$

Therefore,

$$\|r^j\| \le \min_{\phi\in P_j,\phi(0)=1}\ \max_{\lambda\in\Lambda(I_m+WW^T)} |\phi(\lambda)|\|\Theta_B^{1/2}r_B^0\|_{\Theta_B^{-1}},$$

and the claim is proved after substituting $\|\Theta_B^{1/2} r_B^0\|_{\Theta_B^{-1}} = \|r_B^0\|$. □

3.4 Identifying and factorising the matrix B

The preconditioner P was derived on the assumption that it should be significantly cheaper to compute sparse factors of just the matrix B than computing a Cholesky factorisation of the coefficient matrix of the normal equations. Assuming that A has full row rank, we can find an m by m non-singular sub-matrix B.

The matrix B is given by the first m linearly independent columns of the matrix $\widetilde{A}$, where the columns of $\widetilde{A}$ are those of the constraint matrix A, ordered by increasing value of θ_j^{-1}. The set of columns forming B is identified by applying Gaussian elimination to the matrix $\widetilde{A}$, as described below. Although this yields an LU factorisation of B, the factorisation is not efficient with respect to sparsity and its use in subsequent PCG iterations would be costly. This potential cost is reduced significantly by using the Tomlin matrix inversion procedure [69] to determine the factorisation of B for use in PCG iterations. The Tomlin procedure is a relatively simple method of triangularisation and factorisation that underpins the highly efficient implementation of the revised simplex method described by Hall and McKinnon [41]. Since the matrix B is analogous to a simplex basis matrix, the use of the Tomlin procedure in this thesis is expected to be similarly advantageous.

3.4.1 Identifying the columns of B via Gaussian elimination

When applying Gaussian elimination to the matrix $\widetilde{A}$ in order to identify the set of columns forming B, it is important to stress that the matrix $\widetilde{A}$ is not updated when elimination operations are identified. The linear independence of a particular column of $\widetilde{A}$, with respect to columns already in B, is determined as follows.

Suppose that k columns of B have been determined and let L_k be the current lower triangular matrix of elimination multipliers. Let a_q be the first column of $\widetilde{A}$ that has not yet been considered for inclusion in B. The system $L_k \hat{a}_q = a_q$ is solved and the entries of the pivotal column $\hat{a}_q$ are scanned for a good pivotal value. At each step of Gaussian elimination, one requires to divide the indices of the pivotal column by the pivot. So it is necessary to choose the pivot with large magnitude. Usually the pivot is chosen to be the coefficient which has the maximum magnitude among the coefficients of the pivotal column. On the other hand, chosen the pivot plays an important role in term of sparsity. So, we consider the pivot to be good if it has an acceptable large magnitude and has relatively small row count.

If there are no acceptable pivots, indicating that a_q is linearly dependent on the columns already in B, then a_q is discarded. Otherwise, a pivot is chosen and a_q is added to the set of columns forming B.

At least m systems of the form $L_k \hat{a}_q = a_q$ must be solved in order to identify all the columns of B. For some problems, a comparable number of linearly dependent columns of $\widetilde{A}$ are encountered before a complete basis is formed. Thus the efficiency with which $L_k \hat{a}_q = a_q$ is solved is crucial. Additionally, the ill-conditioning of B may lead to PCG being prohibitively expensive. This issue of efficiency is addressed in the following two ways.

Firstly, in order to reduce the number of nonzeros in the matrices L_k, the pivotal entry in $\hat{a}_q$ is selected from the set of acceptable pivots on grounds of sparsity. If the matrix $\widetilde{A}$ were updated with respect to elimination operations, then the acceptable pivot of minimum row count could be chosen. Since this is not known, a set of approximate row counts is maintained and used to discriminate between acceptable pivots. This set of approximate row counts is initialised to be correct and then, as elimination operations are identified, updated according to the maximum fill-in that could occur were $\widetilde{A}$ to be updated. (The row counts are initially the number of nonzero indices in each row of $\widetilde{A}$. Then at each step of Gaussian elimination the row counts are approximately updated. Row counts are updated if fill-in occurs, while they are not updated if cancellations occur. Consequently, the same indices may include more than once if it is removed and then it is created again. In practice however, there is no much advantage of checking for cancellations and keeping the list of cancel indices.)

Secondly, since a_q is sparse, consideration is given to the likelihood that $\hat{a}_q$ is also sparse. This is trivially the case when $k = 0$ since $\hat{a}_q = a_q$. Since the columns of L_k are subsets of the entries in pivotal columns, it follows that for small values of k, $\hat{a}_q$ will remain sparse. For some important classes of LP problems, this property holds for all k and is analogous to what Hall and McKinnon term hyper-sparsity [41]. Techniques for exploiting hyper-sparsity when forming $\hat{a}_q$ analogous to those described in [41] have been used when computing the preconditioner and have led to significant improvements in computational performance.

Tomlin invert

We apply the Tomlin matrix inversion procedure to the matrix B to determine a sparser LU factorisation for B.

The active sub-matrix at any time in the Tomlin procedure consists of those rows and columns in which a pivot has not been found. Initially it is the whole matrix B. The Tomlin procedure has the following steps:

1. Find any identity columns of the matrix B and then eliminate these columns and their corresponding rows from the active sub-matrix.

2. Find any singleton row in the active sub-matrix and eliminate it together with the corresponding column. Store the column of singleton row in the matrix L. Repeat this step to find all singleton rows in the active sub-matrix.

3. Find any singleton column in the active sub-matrix and eliminate it together with the corresponding row from the active sub-matrix. Store the singleton column in the matrix U. Repeat this to find all singleton columns in the active sub-matrix.

4. Repeat 2 and 3 until there are no more singleton rows or columns.

5. If the active sub-matrix is empty then stop. Otherwise, move to next step.

6. Apply Gaussian elimination to the remaining active sub-matrix.

Chapter 4

Inexact Interior Point Method

The consequence of using an iterative method to solve the linear system which arises from IPMs, is solving the KKT system approximately. In this case, the Newton method (1.4) is solved approximately. So instead of (1.4) we have the following system.

$$F^{'}(t^k)\Delta t^k = -F(t^k) + r^k, \tag{4.1}$$

where r^k is the residual of the inexact Newton method. Any approximate step is accepted provided that the residual r^k is small such as

$$\|r^k\| \leq \eta_k \|F(t^k)\|, \tag{4.2}$$

as required by the theory [20, 47]. We refer to the term η_k as the forcing term.

The original content of this chapter has already appeared in [1], co-authored with Jacek Gondzio.

The idea behind inexact interior point algorithms is to derive a stopping criterion of the iterative linear system solvers that minimizes the computational effort involved in computing the search directions and guarantee global convergence [5].

We use the PCG method to solve the augmented system (1.7) preconditioned by a block triangular matrix P (3.7). As a result of this the search directions are computed approximately. That makes it necessary to rethink the convergence of the interior point algorithms, whose convergence are proved under the assumption that the search directions are calculated exactly. In this chapter we focus on one interior point algorithm which is the infeasible path-following algorithm. In order to prove the convergence of the inexact infeasible path-following algorithm (IIPF algorithm) we should prove first the convergence of the PCG method applied to the indefinite system (1.7) then we prove the convergence of the IIPF algorithm.

In the previous chapter we proved that the PCG method applied to the indefinite system (1.7) preconditioned with (3.7) and initialized with an appropriate starting point (3.21), converges in a similar way to the case of applying PCG to a positive definite system. In this chapter we show that applying PCG to solve (1.7) with the preconditioner (3.7) can be analysed using the classical framework of the inexact Newton method (4.1).

The use of inexact Newton methods in interior point methods for LP was investigated in [5, 6, 16, 29, 58, 59]. In [5] the convergence of the infeasible interior point algorithm of Kojima, Megiddo, and Mizuno is proved under the assumption that the iterates are bounded. Monteiro and O'Neal [59] propose the convergence analysis of inexact infeasible long-step primal-dual algorithm and give complexity results for this method. In [59] the PCG method is used to solve the normal equations preconditioned with a sparse preconditioner. The proposed preconditioner was inspired by the Maximum Weight Basis

Algorithm developed in [64]. In [7] an inexact interior point method for semidefinite programming is presented. It allows the linear system to be solved to a low accuracy when the current iterate is far from the solution. In [50] the convergence analysis of inexact infeasible primal-dual path-following algorithm for convex quadratic programming is presented. In these papers the search directions are inexact as the PCG method is used to solve the normal equations. Korzak [49] proves the convergence of the inexact infeasible interior point algorithm of Kojima, Megiddo and Mizuno for LP. This is for search directions which are computed approximately for any iterative solver. This convergence is proven under the assumption that the iterates are bounded. Furthermore, in [82] Zhou and Toh show that the primal-dual inexact infeasible interior point algorithm can find the ϵ-approximate solution of a semidefinite programm in $O(n^2 \ln(1/\epsilon))$ iterations. That is also for search directions which are computed approximately for any iterative solver without the need of assuming the boundedness of the iterations. That is because residuals satisfy specific conditions. One of these conditions is dependent on the smallest singular value of the constraint matrix.

In order to provide the complexity result for the inexact infeasible interior point methods, one should find an upper bound on $|\Delta x^T \Delta s|$ at each iteration of IPM. In [50] the authors change the neighbourhood of the interior point algorithm for QP. The same approach is used to find a bound on $|\Delta x^T \Delta s|$ in [59]. However, that does not work for LP case. The authors assume that there is a point $(\bar{x}, \bar{y}, \bar{s})$ such that the residual of the infeasible primal-dual algorithm is zero (the point $(\bar{x}, \bar{y}, \bar{s})$ is primal-dual feasible) and there is a strictly positive point (x^0, y^0, s^0) such that $(x^k, y^k, s^k) = \rho(x^0, y^0, s^0)$, where $\rho \in [0, 1]$ and also $(x^0, s^0) \geq (\bar{x}, \bar{s})$. These conditions are restrictive and do not always hold. In [6, 7] the inexactness comes from solving the normal equation system iteratively. In order to find a bound on $|\Delta x^T \Delta s|$, the authors find a bound on the normal equations matrix. However, in [82] the authors force residual to satisfy specific conditions, one of which depends on the singular value on the constraint matrix.

In our case we do not require the residual of the inexact Newton method to satisfy a sophisticated condition. The condition on the residual is defined by $\|r^k\| \leq \eta_k \mu_k$. This condition allows a low accuracy when the current iterate is far from the solution and high accuracy as the interior point method approaches optimality, because the term μ_k decreases as the iterations move toward the solution. Furthermore, we use shifting residual strategy, which makes the proof of the convergence and the complexity result of the inexact infeasible path-following algorithm follow the exact case.

In this chapter we study the convergence analysis of inexact infeasible path following algorithm for linear programming as the PCG method is used to solve the augmented system preconditioned with block triangular sparse preconditioner. We prove the global convergence and the complexity result for this method without having to assume the boundedness of the iterates. We design a suitable stopping criteria for the PCG method. This plays an important role in the whole convergence of IIPF algorithm. This stopping criteria allows a low accuracy when the current iterate is far from the solution. We state conditions on the forcing term of inexact Newton method in order to prove the convergence of IIPF algorithm.

The inexact approach in this thesis can be used in the cases where the augmented system is solved iteratively, provided that the residual of this iterative method has a zero block $r = [r_1, 0]$. So we can carry out this approach to cases like [65] for example.

4.1 The residual of inexact Newton method

Using the PCG method to solve the augmented system (1.7) produces a specific value of the residual of the inexact Newton method (4.1). So we shall find the value of the residual r in (4.1) in order to prove the convergence of inexact infeasible path following algorithm and provide a complexity result.

Solving (1.7) approximately gives

$$\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} + \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}, \tag{4.3}$$

where $r_1 = [r_B, r_N]$.

That gives the following equations:

$$-X^{-1}S\Delta x + A^T\Delta y \;=\; f + r_1 \;=\; c - A^T y - \sigma\mu X^{-1}e + r_1, \tag{4.4}$$

$$A\Delta x \;=\; g + r_2 \;=\; b - Ax + r_2. \tag{4.5}$$

Then we find Δs by substituting Δx in (1.6). However, we can shift the residual from (4.4) to (1.6) by assuming there is a residual h while computing Δs. Then (1.6) is replaced by

$$\Delta s = -X^{-1}S\Delta x - s + \sigma\mu X^{-1}e + h,$$

which we can rewrite as

$$-X^{-1}S\Delta x = \Delta s + s - \sigma\mu X^{-1}e - h.$$

Substituting it in (4.4) gives

$$A^T \Delta y + \Delta s = c - A^T y - s + h + r_1.$$

To satisfy the second equation of (1.5) we choose $h = -r_1$. This gives

$$A^T \Delta y + \Delta s = c - A^T y - s, \tag{4.6}$$

and

$$\Delta s = -X^{-1} S \Delta x - s + \sigma \mu X^{-1} e - r_1,$$

which implies

$$S \Delta x + X \Delta s = -XSe + \sigma \mu e - X r_1. \tag{4.7}$$

Equations (4.5), (4.6) and (4.7) give

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{bmatrix} + \begin{bmatrix} r_2 \\ 0 \\ -X r_1 \end{bmatrix},$$

where $\xi_p = b - Ax$, $\xi_d = c - A^T y - s$, $\xi_\mu = -XSe + \sigma \mu e$ and $\sigma \in [0, 1]$.

In the setting in which we apply the PCG method to solve (1.7) preconditioned with (3.7) we have $r_2 = 0$ and $r_1 = [r_B, 0]$, see equation (3.32) in the proof of Theorem 3.3.5. Therefore, the inexact Newton method residual

r is

$$r = \begin{bmatrix} 0 \\ 0 \\ -Xr_1 \end{bmatrix}$$

with $Xr_1 = \begin{bmatrix} X_B r_B \\ X_N r_N \end{bmatrix} = \begin{bmatrix} X_B r_B \\ 0 \end{bmatrix}$.

Shifting the residual from (4.4) to (1.6) is an essential step to prove the convergence of the IIPF algorithm. It results in moving the residual from the second row to the last row of the inexact Newton system, which makes the proof of the convergence of the IIPF Algorithm much easier, as we will see in Section 4.2.

The issue of choosing the stopping criteria of inexact Newton method to satisfy the condition (4.2) has been discussed in many papers. See for example [5, 6, 7, 49, 82]. In [5, 6] the residual of inexact Newton method is chosen such that

$$\|r^k\| \leq \eta_k \mu_k,$$

while in [7] the choice satisfies

$$\|r^k\| \leq \eta_k (n\mu_k).$$

Let the residual be $r = [r_p, r_d, r_\mu]$. According to Korzak [49], the residual is chosen such that

$$\begin{aligned}\|r_p^k\|_2 &\leq (1-\tau_1)\|Ax^k - b\|_2,\\ \|r_d^k\|_2 &\leq (1-\tau_2)\|A^Ty^k + s^k - c\|_2,\\ \|r_\mu^k\|_\infty &\leq \tau_3\mu_k.\end{aligned}$$

where $\tau_1, \tau_2 \in (0,1]$ and $\tau_3 \in [0,1)$ are some appropriately chosen constants.

In our case $r_p = r_d = 0$, we will stop the PCG algorithm when

$$\|r_\mu^k\|_\infty \leq \eta_k\mu_k.$$

As $r_\mu^k = -X^k r_1^k$ and $r_1 = [r_B, 0]$, the stopping criteria becomes

$$\|X_B^k r_B^k\|_\infty \leq \eta_k\mu_k. \tag{4.8}$$

We terminate the PCG algorithm when the stopping criteria (4.8) is satisfied. This stopping criteria allows a low accuracy when the current iterate is far from the solution. In the later iterations the accuracy increases because the average complementarity gap μ reduces from one iteration to another.

4.2 Convergence of the IIPF Algorithm

In this section we carry out the proof of the convergence of the IIPF algorithm and derive a complexity result. In the previous section we used the shifting residual strategy, which makes the proof of the convergence of this inexact algorithm similar to that of the exact case.

This section is organised as follows. First we describe the IIPF algorithm. Then in Lemmas 4.2.1, 4.2.2 and 4.2.3 we derive useful bounds on the iterates. In Theorems 4.2.4 and 4.2.5 we prove that there is a step length α such that the new iteration generated by IIPF algorithm belongs to the neighbourhood $\mathcal{N}_{-\infty}(\gamma,\beta)$ and the average complementarily gap decreases. In order to prove that we supply conditions on the forcing term η_k. In Theorem 4.2.6 we show that the sequence $\{\mu_k\}$ converges Q-linearly to zero and the normal residual sequence $\{\|(\xi_p^k,\xi_d^k)\|\}$ converges R-linearly to zero. Finally in Theorem 4.2.7, we provide the complexity result for this algorithm.

Definition: The central path neighbourhood $\mathcal{N}_{-\infty}(\gamma,\beta)$ is defined by

$$\begin{aligned}\mathcal{N}_{-\infty}(\gamma,\beta)=\{(x,y,s):\|(\xi_p,\xi_d)\|/\mu\le\beta\|(\xi_p^0,\xi_d^0)\|/\mu_0,\ (x,s)>0,\\ x_is_i\ge\gamma\mu,\ i=1,2,...,n\},\end{aligned}\tag{4.9}$$

where $\gamma\in(0,1)$ and $\beta\ge1$ [77].

4.2.1 Inexact Infeasible Path-Following Algorithm

1. Given $\gamma,\beta,\sigma_{min},\sigma_{max}$ with $\gamma\in(0,1)$, $\beta\ge1$, $0<\sigma_{min}<\sigma_{max}<0.5$, and

   $0<\eta_{min}<\eta_{max}<1$; choose (x^0,y^0,s^0) with $(x^0,s^0)>0$;

2. For $k=0,1,2,...$

   - choose $\sigma_k\in[\sigma_{min},\sigma_{max}]$ and $\eta_k\in[\eta_{min},\eta_{max}]$ such that $\eta_k<\frac{\sigma_k(1-\gamma)}{(1+\gamma)}$ and $\eta_k+\sigma_k<0.99$; and solve

$$\begin{bmatrix}A&0&0\\0&A^T&I\\S^k&0&X^k\end{bmatrix}\begin{bmatrix}\Delta x^k\\\Delta y^k\\\Delta s^k\end{bmatrix}=\begin{bmatrix}\xi_p^k\\\xi_d^k\\\sigma_k\mu_ke-X^kS^ke\end{bmatrix}-\begin{bmatrix}0\\0\\X^kr_1^k\end{bmatrix}\tag{4.10}$$

Such that $r_N^k = 0$ and

$$\|X_B^k r_B^k\|_\infty \leq \eta_k \mu_k, \tag{4.11}$$

- choose α_k as the largest value of α in $[0, 1]$ such that

$$(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma, \beta) \tag{4.12}$$

and the following Armijo condition holds:

$$\mu_k(\alpha) \leq (1 - .01\alpha)\mu_k; \tag{4.13}$$

- set $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k(\alpha_k), y^k(\alpha_k), s^k(\alpha_k))$;
- stop when $\mu_k < \epsilon$, for a small positive constant ϵ.

In this section we will follow the convergence analysis of the infeasible path-following algorithm proposed originally by Zhang [81]. However, we will follow the proof techniques proposed in Wright's book [77].

Firstly, let us introduce the quantity

$$\nu_k = \prod_{j=0}^{k-1}(1 - \alpha_j),\ \nu_0 = 1$$

Note that $\xi_p^{k+1} = b - Ax^{k+1} = b - A(x^k + \alpha_k \Delta x^k) = b - Ax^k - \alpha_k A \Delta x^k = \xi_p^k - \alpha_k A \Delta x^k$, from the first row of (4.10) we get

$$\xi_p^{k+1} = (1 - \alpha_k)\xi_p^k, \tag{4.14}$$

which implies

$$\xi_p^k = \nu_k \xi_p^0.$$

Note also $\xi_d^{k+1} = c - A^T y^{k+1} - s^{k+1} = c - A^T(y^k + \alpha_k \Delta y^k) - (s^k + \alpha_k \Delta s^k) = (c - A^T y^k - s^k) - \alpha_k(A^T \Delta y^k + \Delta s^k) = \xi_d^k - \alpha_k(A^T \Delta y^k + \Delta s^k)$. From the second row of (4.10) we get

$$\xi_d^{k+1} = (1 - \alpha_k)\xi_d^k, \tag{4.15}$$

which implies

$$\xi_d^k = \nu_k \xi_d^0,$$

Consequently, the quantity ν_k satisfies

$$\nu_k \leq \beta \frac{\mu_k}{\mu_0}.$$

More details can be found in [77].

Let (x^*, y^*, s^*) be any primal-dual solution.

Lemma 4.2.1. *Assume that $(x^k, y^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$, $(\Delta x^k, \Delta y^k, \Delta s^k)$ satisfies (4.10) and (4.11) for all $k \geq 0$, and $\mu_k \leq (1 - .01\alpha_{k-1})\mu_{k-1}$ for all $k \geq 1$. Then there is a positive constant C_1 such that for all $k \geq 0$*

$$\nu_k \|(x^k, s^k)\| \leq C_1 \mu_k, \tag{4.16}$$

where C_1 is given as

$$C_1 = \zeta^{-1}(n\beta + n + \beta\|(x^0, s^0)\|_\infty \|(x^*, s^*)\|_1 / \mu_0),$$

where

$$\zeta = \min_{i=1,\ldots,n} \min(x_i^0, s_i^0).$$

The proof of this Lemma is similar to the proof of Lemma 6.3 in [77]. Moreover, we follow the same logic as in [77] to prove the following lemma.

Lemma 4.2.2. *Assume that $(x^k, y^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$, $(\Delta x^k, \Delta y^k, \Delta s^k)$ satisfies (4.10) and (4.11) for all $k \geq 0$, and $\mu_k \leq (1 - .01\alpha_{k-1})\mu_{k-1}$ for all $k \geq 1$. Then there is a positive constant C_2 such that*

$$\|D^{-1}\Delta x^k\| \leq C_2 \mu_k^{1/2}, \tag{4.17}$$

$$\|D\Delta s^k\| \leq C_2 \mu_k^{1/2}, \tag{4.18}$$

where $D = X^{1/2}S^{-1/2}$. For all $k \geq 0$.

Proof. For simplicity we omit the iteration index k in the proof.

Let

$$(\bar{x}, \bar{y}, \bar{s}) = (\Delta x, \Delta y, \Delta s) + \nu_k(x^0, y^0, s^0) - \nu_k(x^*, y^*, s^*).$$

Then $A\bar{x} = 0$ and $A^T\bar{y} + \bar{s} = 0$, which implies $\bar{x}^T\bar{s} = 0$.

$A\bar{x} = 0$ because

$$A\bar{x} = A\Delta x + \nu_k Ax^0 - \nu_k Ax^* = \xi_p + \nu_k Ax^0 - \nu_k b = \xi_p - \nu_k \xi_0 = 0.$$

Similarly one can show that $A^T\bar{y} + \bar{s} = 0$. Hence

$$0 = \bar{x}^T\bar{s} = (\Delta x + \nu_k x^0 - \nu_k x^*)^T(\Delta s + \nu_k s^0 - \nu_k s^*). \tag{4.19}$$

Using the last row of (4.10) implies

$$\begin{aligned}&S(\Delta x+\nu_k x^0-\nu_k x^*)+X(\Delta s+\nu_k s^0-\nu_k s^*)\\&=S\Delta x+X\Delta s+\nu_k S(x^0-x^*)+\nu_k X(s^0-s^*)\\&=-XSe+\sigma\mu e-Xr_1+\nu_k S(x^0-x^*)+\nu_k X(s^0-s^*).\end{aligned}$$

By multiplying this system by $(XS)^{-1/2}$, we get

$$\begin{aligned}&D^{-1}(\Delta x+\nu_k x^0-\nu_k x^*)+D(\Delta s+\nu_k s^0-\nu_k s^*)\\&=(XS)^{-1/2}(-XSe+\sigma\mu e-Xr_1)+\nu_k D^{-1}(x^0-x^*)+\nu_k D(s^0-s^*).\end{aligned}$$

The equality (4.19) gives

$$\begin{aligned}&\|D^{-1}(\Delta x+\nu_k x^0-\nu_k x^*)+D(\Delta s+\nu_k s^0-\nu_k s^*)\|^2=\\&\|D^{-1}(\Delta x+\nu_k x^0-\nu_k x^*)\|^2+\|D(\Delta s+\nu_k s^0-\nu_k s^*)\|^2.\end{aligned}$$

Consequently,

$$\begin{aligned}&\|D^{-1}(\Delta x+\nu_k x^0-\nu_k x^*)\|^2+\|D(\Delta s+\nu_k s^0-\nu_k s^*)\|^2\\&=\|(XS)^{-1/2}(-XSe+\sigma\mu e-Xr_1)+\nu_k D^{-1}(x^0-x^*)+\nu_k D(s^0-s^*)\|^2,\end{aligned}\tag{4.20}$$

which leads to

$$\begin{aligned}&\|D^{-1}(\Delta x+\nu_k x^0-\nu_k x^*)\|\leq\|(XS)^{-1/2}(-XSe+\sigma\mu e-Xr_1)\\&+\nu_k D^{-1}(x^0-x^*)+\nu_k D(s^0-s^*)\|\leq\|(XS)^{-1/2}(-XSe+\sigma\mu e-Xr_1)\|\\&+\nu_k\|D^{-1}(x^0-x^*)\|+\nu_k\|D(s^0-s^*)\|.\end{aligned}$$

The triangle inequality and addition of an extra term $\nu_k\|D(s^0-s^*)\|$ to the right hand side give

$$\begin{aligned}\|D^{-1}\Delta x\| \leq \ & \|(XS)^{-1/2}[-XSe+\sigma\mu e - Xr_1]\| + 2\nu_k\|D^{-1}(x^0-x^*)\| \\ & +2\nu_k\|D(s^0-s^*)\|.\end{aligned} \tag{4.21}$$

(4.20) leads to

$$\begin{aligned}&\|D(\Delta s+\nu_k s^0-\nu_k s^*)\| \leq \|(XS)^{-1/2}(-XSe+\sigma\mu e-Xr_1)+\nu_k D^{-1}(x^0-x^*) \\ &+\nu_k D(s^0-s^*)\| \leq \|(XS)^{-1/2}(-XSe+\sigma\mu e-Xr_1)\|+\nu_k\|D^{-1}(x^0-x^*)\| \\ &+\nu_k\|D(s^0-s^*)\|.\end{aligned}$$

The triangle inequality and addition of an extra term $\nu_k\|D^{-1}(x^0-x^*)\|$ to the right hand side give

$$\begin{aligned}\|D\Delta s\| \leq \ & \|(XS)^{-1/2}[-XSe+\sigma\mu e - Xr_1]\| + 2\nu_k\|D^{-1}(x^0-x^*)\| \\ & +2\nu_k\|D(s^0-s^*)\|.\end{aligned} \tag{4.22}$$

We can write

$$\begin{aligned}\|(XS)^{-1/2}(-XSe+\sigma\mu e-Xr_1)\|^2 &= \sum_{i=1}^{n}\frac{(-x_is_i+\sigma\mu-x_ir_{1,i})^2}{x_is_i} \\ \leq \frac{\|-XSe+\sigma\mu e-Xr_1\|^2}{\min_i x_is_i} &\leq \frac{1}{\gamma\mu}\|-XSe+\sigma\mu e-Xr_1\|^2.\end{aligned}$$

because $(x,y,s)\in\mathcal{N}_{-\infty}(\gamma,\beta)$ which implies $x_is_i\geq\gamma\mu$ for $i=1,...,n$.

On the other hand,

$$\begin{aligned}\|-XSe+\sigma\mu e\|^2 &= \|XSe\|^2+\|\sigma\mu e\|^2-2\sigma\mu e^TXSe = \|XSe\|^2+n\sigma^2\mu^2-2n\sigma\mu^2 \\ &\leq \|XSe\|_1^2+n\sigma^2\mu^2-2n\sigma\mu^2 = (x^Ts)^2+n\sigma^2\mu^2-2n\sigma\mu^2 \\ &\leq n^2\mu^2+n\sigma^2\mu^2-2n\sigma\mu^2 \leq n^2\mu^2,\end{aligned}$$

as $\sigma \in (0,1)$. This leads to

$$\begin{aligned} \|-XSe+\sigma\mu e - Xr_1\| &\leq \|-XSe+\sigma\mu e\| + \|Xr_1\| \\ &\leq n\mu + \sqrt{n}\|X_B r_B\|_\infty \leq n\mu + \sqrt{n}\eta\mu \\ &\leq n\mu + \sqrt{n}\eta_{max}\mu, \end{aligned}$$

which implies the following

$$\|(XS)^{-1/2}(-XSe+\sigma\mu e - Xr_1)\| \leq \gamma^{-1/2}(n+\sqrt{n}\eta_{max})\mu^{1/2}. \tag{4.23}$$

On the other hand

$$\begin{aligned} &\nu_k\|D^{-1}(x^0-x^*)\| + \nu_k\|D(s^0-s^*)\| \\ &\leq \nu_k(\|D^{-1}\|+\|D\|)\max(\|x^0-x^*\|, \|s^0-s^*\|). \end{aligned} \tag{4.24}$$

For the matrix norm $\|D^{-1}\|$, we have

$$\|D^{-1}\| \leq \max_i \|D_{ii}^{-1}\| = \|D^{-1}e\|_\infty = \|(XS)^{-1/2}Se\|_\infty \leq \|(XS)^{-1/2}\|\|s\|_1,$$

and similarly

$$\|D\| \leq \|(XS)^{-1/2}\|\|x\|_1.$$

Using Lemma 4.2.1 and (4.24) we get

$$\nu_k\|D^{-1}(x^0-x^*)\| + \nu_k\|D(s^0-s^*)\| \leq \nu_k\|(x,s)\|_1\|(XS)^{-1/2}\|\max(\|x^0-x^*\|,$$
$$\|s^0-s^*\|) \leq C_1\gamma^{-1/2}\mu^{1/2}\max(\|x^0-x^*\|, \|s^0-s^*\|).$$

By substituting the previous inequality and (4.23) in (4.21) and (4.22) we get

$$\|D^{-1}\Delta x\| \leq (\gamma^{-1/2}(n + \sqrt{n}\eta_{max}) + 2C_1\gamma^{-1/2}\max(\|x^0 - x^*\|, \|s^0 - s^*\|))\mu^{1/2}$$

and

$$\|D\Delta s\| \leq (\gamma^{-1/2}(n + \sqrt{n}\eta_{max}) + 2C_1\gamma^{-1/2}\max(\|x^0 - x^*\|, \|s^0 - s^*\|))\mu^{1/2}.$$

Let us define C_2 as

$$C_2 = \gamma^{-1/2}(n + \sqrt{n}\eta_{max}) + 2C_1\gamma^{-1/2}\max(\|x^0 - x^*\|, \|s^0 - s^*\|).$$

which completes the proof. □

Lemma 4.2.3. *Assume that $(x^k, y^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$, $(\Delta x^k, \Delta y^k, \Delta s^k)$ satisfies (4.10) and (4.11) for all $k \geq 0$, and $\mu_k \leq (1 - .01\alpha_{k-1})\mu_{k-1}$ for all $k \geq 1$. Then there is a positive constant C_3 such that*

$$|(\Delta x^k)^T \Delta s^k| \leq C_3\mu_k, \tag{4.25}$$

$$|\Delta x_i^k \Delta s_i^k| \leq C_3\mu_k \tag{4.26}$$

for all $k \geq 0$.

Proof. For simplicity we omit the iteration index k in the proof. From Lemma 4.2.2 we have

$$|\Delta x^T \Delta s| = |(D^{-1}\Delta x)^T(D\Delta s)| \leq \|D^{-1}\Delta x\|\|D\Delta s\| \leq C_2^2\mu.$$

Moreover, using Lemma 4.2.2 again we obtain

$$|\Delta x_i \Delta s_i| = |D_{ii}^{-1}\Delta x_i D_{ii}\Delta s_i| = |D_{ii}^{-1}\Delta x_i||D_{ii}\Delta s_i| \leq \|D^{-1}\Delta x\|\|D\Delta s\| \leq C_2^2\mu.$$

Let us denote $C_3 = C_2^2$, and the proof is complete. □

Theorem 4.2.4. *Assume that $(x^k, y^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$, $(\Delta x^k, \Delta y^k, \Delta s^k)$ satisfies (4.10) and (4.11) for all $k \geq 0$, and $\mu_k \leq (1 - .01\alpha_{k-1})\mu_{k-1}$ for all $k \geq 1$. Then there is a value $\bar{\alpha} \in (0,1)$ such that the following three conditions are satisfied for all $\alpha \in [0, \bar{\alpha}]$ for all $k \geq 0$*

$$(x^k + \alpha\Delta x^k)^T(s^k + \alpha\Delta s^k) \geq (1-\alpha)(x^k)^T s^k \tag{4.27}$$

$$(x_i^k + \alpha\Delta x_i^k)(s_i^k + \alpha\Delta s_i^k) \geq \frac{\gamma}{n}(x^k + \alpha\Delta x^k)^T(s^k + \alpha\Delta s^k) \tag{4.28}$$

$$(x^k + \alpha\Delta x^k)^T(s^k + \alpha\Delta s^k) \leq (1 - .01\alpha)(x^k)^T s^k. \tag{4.29}$$

Proof. For simplicity we omit the iteration index k in the proof.

The last row of the system (4.10) implies

$$s^T\Delta x + x^T\Delta s = -x^T s + n\sigma\mu - x_B^T r_B,$$

and

$$s_i\Delta x_i + x_i\Delta s_i = -x_i s_i + \sigma\mu - x_i r_{1,i}$$

which leads to

$$\begin{aligned}(x+\alpha\Delta x)^T(s+\alpha\Delta s) &= x^Ts+\alpha(x^T\Delta s+s^T\Delta x)+\alpha^2(\Delta x)^T\Delta s\\ &= x^Ts+\alpha(-x^Ts+n\sigma\mu-x_B^Tr_B)+\alpha^2(\Delta x)^T\Delta s\\ &= (1-\alpha)x^Ts+n\alpha\sigma\mu-\alpha x_B^Tr_B+\alpha^2(\Delta x)^T\Delta s.\end{aligned}$$

Similarly

$$\begin{aligned}(x_i+\alpha\Delta x_i)(s_i+\alpha\Delta s_i) &= x_is_i+\alpha(s_i\Delta x_i+x_i\Delta s_i)+\alpha^2\Delta x_i\Delta s_i\\ &= x_is_i+\alpha(-x_is_i+\sigma\mu-x_ir_{1,i})+\alpha^2\Delta x_i\Delta s_i\\ &= (1-\alpha)x_is_i+\alpha\sigma\mu-\alpha x_ir_{1,i}+\alpha^2\Delta x_i\Delta s_i.\end{aligned}$$

For (4.27) we have

$$\begin{aligned}(x+\alpha\Delta x)^T(s+\alpha\Delta s)-(1-\alpha)x^Ts &= (1-\alpha)x^Ts+n\alpha\sigma\mu-\alpha x_B^Tr_B\\ &\quad +\alpha^2(\Delta x)^T\Delta s-(1-\alpha)x^Ts\\ &= n\alpha\sigma\mu-\alpha x_B^Tr_B+\alpha^2(\Delta x)^T\Delta s\\ &\geq n\alpha\sigma\mu-\alpha|x_B^Tr_B|-\alpha^2|(\Delta x)^T\Delta s|\\ &\geq n\alpha\sigma\mu-n\alpha\eta\mu-\alpha^2C_3\mu\end{aligned}$$

where we used the fact that from (4.11) we have

$$|x_B^Tr_B|\leq n\|X_Br_B\|_\infty\leq n\eta\mu.$$

Therefore, the condition (4.27) holds for all $\alpha\in[0,\alpha_1]$, where α_1 is given by

$$\alpha_1=\frac{n(\sigma-\eta)}{C_3}, \tag{4.30}$$

and we choose $\eta<\sigma-\varepsilon_1$ to guarantee α_1 to be strictly positive, where ε_1 is a constant strictly greater than zero.

Let us consider (4.28)

$$(x_i+\alpha\Delta x_i)(s_i+\alpha\Delta s_i)-\tfrac{\gamma}{n}(x+\alpha\Delta x)^T(s+\alpha\Delta s)=(1-\alpha)x_is_i+\alpha\sigma\mu$$
$$-\alpha x_ir_{1,i}+\alpha^2\Delta x_i\Delta s_i-\tfrac{\gamma}{n}((1-\alpha)x^Ts+n\alpha\sigma\mu-\alpha x_B^Tr_B+\alpha^2(\Delta x)^T\Delta s)$$

because $(x,y,s)\in N_{-\infty}(\gamma,\beta)$, so $x_is_i\geq\gamma\mu$, $\forall i=1,...,n$, that gives

$$(x_i+\alpha\Delta x_i)(s_i+\alpha\Delta s_i)-\tfrac{\gamma}{n}(x+\alpha\Delta x)^T(s+\alpha\Delta s)\geq(1-\alpha)\gamma\mu+\alpha\sigma\mu$$
$$-\alpha\max_i x_ir_{1,i}-\alpha^2|\Delta x_i\Delta s_i|-\gamma(1-\alpha)\mu-\gamma\alpha\sigma\mu+\tfrac{\gamma}{n}\alpha x_B^Tr_B-\tfrac{\gamma}{n}\alpha^2(\Delta x)^T\Delta s$$
$$\geq\alpha\sigma\mu-\alpha\|X_Br_B\|_\infty-\alpha^2C_3\mu-\alpha\sigma\gamma\mu-\tfrac{\gamma}{n}\alpha|x_B^Tr_B|-\tfrac{\gamma}{n}\alpha^2C_3\mu\geq\alpha\sigma\mu-\alpha\eta\mu$$
$$-\alpha^2C_3\mu-\alpha\sigma\gamma\mu-\gamma\alpha\eta\mu-\tfrac{\gamma}{n}\alpha^2C_3\mu\geq\alpha((1-\gamma)\sigma-\eta(1+\gamma))\mu-2\alpha^2C_3\mu$$

Condition (4.28) holds for all $\alpha\in[0,\alpha_2]$, where α_2 is given by:

$$\alpha_2=\frac{\sigma(1-\gamma)-(1+\gamma)\eta}{2C_3}. \tag{4.31}$$

We choose $\eta<\frac{\sigma(1-\gamma)}{(1+\gamma)}-\varepsilon_2$ to guarantee α_2 to be strictly positive, where ε_2 is a constant strictly greater than zero.

Finally, let us consider condition (4.29)

$$\tfrac{1}{n}[(x+\alpha\Delta x)^T(s+\alpha\Delta s)-(1-.01\alpha)x^Ts]=$$
$$=\tfrac{1}{n}[(1-\alpha)x^Ts+n\alpha\sigma\mu-\alpha x_B^Tr_B+\alpha^2(\Delta x)^T\Delta s-(1-.01\alpha)x^Ts]$$
$$=\tfrac{1}{n}[-.99\alpha x^Ts+n\alpha\sigma\mu-\alpha x_B^Tr_B+\alpha^2(\Delta x)^T\Delta s]$$
$$\leq-.99\alpha\mu+\alpha\sigma\mu+\tfrac{\alpha}{n}|x_B^Tr_B|+\tfrac{\alpha^2}{n}C_3\mu\leq-.99\alpha\mu+\alpha\sigma\mu+\alpha\eta\mu+\tfrac{\alpha^2}{n}C_3\mu.$$

We can conclude that condition (4.29) holds for all $\alpha\in[0,\alpha_3]$, where α_3 is given by:

$$\alpha_3=\frac{n(0.99-\sigma-\eta)}{C_3}. \tag{4.32}$$

We choose η and σ such that $\eta + \sigma < 0.99 - \varepsilon_3$ to guarantee α_3 to be strictly positive, where ε_3 is a constant strictly greater than zero.

Combining the bounds (4.30), (4.31) and (4.32), we conclude that conditions (4.27), (4.28) and (4.29) hold for $\alpha \in [0, \bar{\alpha}]$, where

$$\bar{\alpha} = \min\left\{1, \frac{n(\sigma - \eta)}{C_3}, \frac{\sigma(1-\gamma) - (1+\gamma)\eta}{2C_3}, \frac{n(0.99 - \sigma - \eta)}{C_3}\right\}. \tag{4.33}$$

□

We introduce the constants ε_1, ε_2 and ε_3 to guarantee that the limit of the step length $\bar{\alpha}$ is strictly greater than zero and to make it flexible to choose the parameters η_k and σ_k.

Note that if $\eta < \frac{\sigma(1-\gamma)}{(1+\gamma)}$ then $\eta < \sigma$ because $\frac{(1-\gamma)}{(1+\gamma)} < 1$ for any $\gamma \in (0, 1)$.

From this theorem we observe that the forcing term η_k should be chosen such that the following two conditions $\eta_k < \frac{\sigma_k(1-\gamma)}{(1+\gamma)} - \varepsilon_2$ and $\eta_k + \sigma_k < 0.99 - \varepsilon_3$ are satisfied. Under these assumption the following theorem guarantees that there is a step length α such that the new point belongs to the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$ and its average complementarity gap decreases according to condition (4.13).

Below we prove two theorems using standard techniques which follow from Wright [77].

Theorem 4.2.5. *Assume that $\eta_k < \frac{\sigma_k(1-\gamma)}{(1+\gamma)} - \varepsilon_2$, $\eta_k + \sigma_k < 0.99 - \varepsilon_3$ for $\varepsilon_2, \varepsilon_3 > 0$, $(x^k, y^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$ and $(\Delta x^k, \Delta y^k, \Delta s^k)$ satisfies (4.10) and (4.11) for all $k \geq 0$, $\mu_k \leq (1 - .01\alpha_{k-1})\mu_{k-1}$ for all $k \geq 1$. Then $(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma, \beta)$ and $\mu_k(\alpha) \leq (1 - .01\alpha)\mu_k$ for all $\alpha \in [0, \bar{\alpha}]$, where $\bar{\alpha}$ is given by (4.33).*

Proof. Theorem 4.2.4 ensures that the conditions (4.27), (4.28) and (4.29) are satisfied. Note that (4.29) implies that the condition $\mu_k(\alpha) \leq (1 - .01\alpha)\mu_k$

is satisfied, while (4.28) guarantees that $x_i^k(\alpha)s_i^k(\alpha) \geq \gamma\mu_k(\alpha)$.

To prove that $(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma, \beta)$, we have to prove that $\|(\xi_p^k(\alpha), \xi_d^k(\alpha))\|/\mu_k(\alpha) \leq \beta\|(\xi_p^0, \xi_d^0)\|/\mu_0$. From (4.14), (4.15) and (4.27) we have

$$\begin{aligned}\tfrac{\|(\xi_p^k(\alpha),\xi_d^k(\alpha))\|}{\mu_k(\alpha)} &= \tfrac{(1-\alpha)\|(\xi_p^k,\xi_d^k)\|}{\mu_k(\alpha)} \leq \tfrac{(1-\alpha)\|(\xi_p^k,\xi_d^k)\|}{(1-\alpha)\mu_k} \leq \tfrac{\|(\xi_p^k,\xi_d^k)\|}{\mu_k} \\ &\leq \beta\tfrac{\|(\xi_p^0,\xi_d^0)\|}{\mu_0},\end{aligned}$$

since $(x^k, y^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$. □

Theorem 4.2.6. *The sequence $\{\mu_k\}$ generated by the IIPF Algorithm converges Q-linearly to zero, and the sequence of residual norms $\{\|(\xi_p^k, \xi_d^k)\|\}$ converges R-linearly to zero.*

Proof. Q-linear convergence of $\{\mu_k\}$ follows directly from condition (4.13) and Theorem 4.2.4. There exists a constant $\bar{\alpha} > 0$ such that $\alpha_k \geq \bar{\alpha}$ for every k such that

$$\mu_{k+1} \leq (1 - .01\alpha_k)\mu_k \leq (1 - .01\bar{\alpha})\mu_k, \text{ for all } k \geq 0.$$

From (4.14) and (4.15) we also have

$$\|(\xi_p^{k+1}, \xi_d^{k+1})\| \leq (1 - \alpha_k)\|(\xi_p^k, \xi_d^k)\|.$$

Therefore,

$$\|(\xi_p^{k+1}, \xi_d^{k+1})\| \leq (1 - \bar{\alpha})\|(\xi_p^k, \xi_d^k)\|.$$

Also from Theorem 4.2.5 we know that

$$\|(\xi_p^{k+1}, \xi_d^{k+1})\| \leq \mu_k\beta\frac{\|(\xi_p^0, \xi_d^0)\|}{\mu_0}.$$

Therefore, the sequence of residual norms is bounded above by another sequence that converges Q-linearly, so $\{\|(\xi_p^k, \xi_d^k)\|\}$ converges R-linearly. □

Theorem 4.2.7. *Let $\epsilon > 0$ and the starting point $(x^0, y^0, s^0) \in \mathcal{N}_{-\infty}(\gamma, \beta)$ in the Algorithm IIPF be given. Then there is an index K with*

$$K = O(n^2 |log\epsilon|)$$

such that the iterates $\{(x^k, y^k, s^k)\}$ generated by IIPF Algorithm satisfy

$$\mu_k \leq \epsilon, \textit{ for all } k \geq K.$$

Proof. If the conditions of Theorem 4.2.5 are satisfied, then the conditions (4.12) and (4.13) are satisfied for all $\alpha \in [0, \bar{\alpha}]$ for all $k \geq 0$. By Theorem 4.2.4, the quantity $\bar{\alpha}$ satisfies

$$\bar{\alpha} \geq \min \left\{ 1, \frac{n(\sigma - \eta)}{C_3}, \frac{\sigma(1-\gamma) - (1+\gamma)\eta}{2C_3}, \frac{n(0.99 - \sigma - \eta)}{C_3} \right\}.$$

Furthermore, from Lemmas 4.2.1, 4.2.2 and 4.2.3 we have $C_3 = O(n^2)$, therefore

$$\bar{\alpha} \geq \frac{\delta}{n^2}$$

for some positive scalar δ independent of n. That implies

$$\mu_{k+1} \leq (1 - .01\bar{\alpha})\mu_k \leq (1 - \frac{.01\delta}{n^2})\mu_k, \text{ for } k \geq 0.$$

The complexity result is an immediate consequence of Theorem 3.2 of [77]. □

Chapter 5

Numerical Results

The numerical results, which are demonstrated in this chapter, have been presented in the paper [2]. The method discussed in this thesis has been implemented in the context of HOPDM [36]. We have implemented the preconditioned conjugate gradients method for the augmented system given a specific starting point. In the implementation, the starting point (3.21) with two zero blocks in its residual is used. We consider a subset of the linear programming problems from the Netlib [30], Kennington [14] and other public test sets used in [60]. In this chapter we indicate that the new approach can be very effective in some cases, and that the new approach is an important option for some classes of problems.

In the initial iterations of the interior point method the normal equations are solved using the direct approach by forming the Cholesky factorisation LDL^T for the normal equations matrix. As the interior point method approaches optimality, the normal equation matrix becomes extremely ill-conditioned due to a very different magnitude of the entries in Θ. At this point, we switch to the iterative solver. In practice, we switch to PCG when two conditions are satisfied: firstly, there are enough small elements in Θ^{-1}

(we have at least $3m/4$ small entries θ_j^{-1}, where $\theta_j^{-1} \leq 10^{-2}$) . Secondly, the relative duality gap is less than or equal to 10^{-2}.

In our implementation, the termination criterion for the PCG method is set as $\|r_k\|/\|r^{(0)}\| < \varepsilon$. Initially, we chose $\varepsilon = 10^{-2}$. When the relative duality gap becomes less than or equal to 10^{-3} the value of ε is changed to 10^{-3} and, finally, when the relative duality gap falls below 10^{-4} the value of ε becomes 10^{-4}.

Through out our study, we assume A has full row rank. This assumption does not effect on the robustness of this approach. Since, if A does not have full rank, we add artificial variables to the constraints to construct full rank constraints matrix A. We also add these variables to the objective function after multiply them with big constant M.

The numerical results, which are shown in this chapter, are calculated for the following case. The matrix B is rebuilt at each iteration of interior point method, where the iterative solver is used. On the other hand, we can used the old information to update B for the next iteration. This will save a lot of factorisation time. However, in this case we will have larger θ_j, and consequently the number of the PCG iterations will increase. The idea of updating B is very interesting, but it requires a lot of work to grab hold of the best total running time (especially, to make a balance between the time of the PCG solver and the LU factorisation) for most of problems. This will be one of our future works.

In Table 5.1, we report the problem sizes: m, n and $nz(A)$ denote the number of rows, columns and nonzeros in the constraint matrix A. In the next two columns, $nz(B)$ denotes the number of nonzeros in the LU factorisation of the basis matrix B and $nz(L)$ denotes the number of nonzero elements in the Cholesky factor of the normal equations matrix. In this chapter, we report results for problems which benefit from the use of the iterative approach presented. As shown in the last column of Table 5.1, the iterative method is storage-efficient, requiring one or two orders of magnitude less memory than the Cholesky factorisation. These results show that in most cases we save more than 90% of the memory by using the LU factorisation compared with Cholesky factorisation. In pds20 problem for instance, the Cholesky factorisation has 1626987 nonzeros, while LU factorisation only has 37123, which makes the memory saving reach 97.7%. If the PCG approach were used for all IPM iterations, this memory advantage would allow certain problems to be solved for which the memory requirement of Cholesky would be prohibitive. In addition, it is essential that the LU factors are smaller by a significant factor since they will have to be applied twice for each PCG iteration when solving for the Newton direction, whereas the direct method using Cholesky factors requires the L factor to be used just twice to compute the Newton direction. The relative memory requirement can also be viewed as a measure of the maximum number of PCG iterations that can be performed while remaining competitive with the direct method using Cholesky factors.

The results of comparing our mixed approach against the pure direct approach are given in Table 5.2. In all reported runs we have asked for eight digits of accuracy in the solution. For each test problem we report the number of interior point iterations and the total CPU time in seconds needed to solve the problem. Additionally, for the mixed approach we also report the number of interior point iterations in which preconditioned conjugate gradients method was used (IPM-pcg). For the problem `fit2p`, for example, 12 of the 25 interior point iterations used the iterative solution method: the remaining 13 iterations used the direct method. In the last column of Table 5.2 we report the saving in the total CPU time, when the mixed approach is used instead of the pure direct approach. For the problem fit2p, for example, the mixed approach is 64% faster than the pure direct approach.

As we report in the column headed "Mixed approach" of Table 5.2, we use the PCG method only in the final iterations of the interior point method, while the rest of the interior point iterations are made using the direct method. For most problems, the numbers of IPM iterations required when using the pure direct and mixed approaches to solve a given problem are the same or differ only slightly. However, for chr15a, pds-10 and pds-20, the mixed approach requires more iterations, significantly so in the case of the latter two problems. In the case of chr15a this accounts for the only negative time saving in Table 5.2. For one problem, chr22b, using the mixed approach leads to significantly fewer IPM iterations being required.

In order to give an insight into the behaviour of the preconditioned conjugate gradients, in Table 5.3 we report the number of PCG iterations needed to solve a particular linear system. First, we report separately this number for the last interior point iteration when our preconditioner is supposed to behave best. The following three columns correspond to the minimum, the average, and the maximum number of PCG iterations encountered throughout all iterative solves.

Finally, in Table 5.4 we report results for the problems solved with the pure iterative method. In these runs we have ignored the spread of elements in the diagonal matrix Θ and the distance to optimality, and we have forced the use of the PCG method in all interior point iterations. Such an approach comes with a risk of failure of the PCG method because the preconditioner does not have all its attractive properties in the earlier IPM iterations. Indeed, we would not advise its use in the general context. However, for several problems in our collection such an approach has been very successful. In this table the term unsolved denotes to that the solver is excess iteration limit.

So far, we have reported some problems, which are benefit of our approach. In Table 5.5 and Table 5.6 we show problems, which do not benefit of our approach. The consequences of using an iterative solver to solve the linear systems which arise from IPM, may lead to increase the number of IPM iterations. The total running time does not improve in the following problems because of this reason: shell, nw14, pds-02 and storm8. In the most of the problems in tables 5.5 and 5.6, the iterative approach works fine. Since, the PCG method converges to the solution in reasonable number of iterations. The slowness of the running time is due to that the solving time of iterative approach increases comparing with the direct approach. In agg and gfrd-pnc for instance there is no much saving in term of nonzero in the factorization, which causes increasing of the solving time.

Problem	Dimensions			Nonzeros in Factors		Memory
	m	n	nz(A)	nz(B)	nz(L)	saving
aircraft	3754	7517	24034	9754	1417131	99.3 %
chr12a	947	1662	5820	5801	78822	92.6 %
chr12b	947	1662	5820	4311	85155	94.9 %
chr12c	947	1662	5820	6187	80318	92.3 %
chr15b	1814	3270	11460	9574	218023	95.6 %
chr15c	1814	3270	11460	9979	219901	95.5 %
chr18a	3095	5679	19908	19559	531166	95.5 %
chr18b	3095	5679	19908	9139	527294	96.3 %
chr20a	4219	7810	27380	38477	885955	95.7 %
chr20b	4219	7810	27380	63243	893674	92.9 %
chr20c	4219	7810	27380	23802	926034	94.7 %
chr22a	5587	10417	36520	33685	1392239	97.5 %
chr22b	5587	10417	36520	38489	1382161	97.2 %
chr25a	8148	15325	53725	49605	2555662	98.1 %
fit1p	628	1677	10894	5002	196251	97.5 %
fit2p	3001	13525	60784	34303	4498500	99.2 %
fome10	6071	12230	35632	114338	1610864	92.2 %
fome11	14695	24460	71264	237844	3221728	92.6 %
fome12	24285	48920	167492	445156	6443456	93.1 %
pds-06	9882	28655	82269	22020	580116	96.2 %
pds-10	16559	48763	140063	37123	1626987	97.7 %
pds-20	33875	105728	304153	77352	6960089	97.7 %
route	20894	23923	187686	14876	3078015	99.5 %
scr10	689	1540	5940	13653	124559	89.0 %
scr12	1151	2784	10716	20437	330483	93.8 %
scr15	2234	6210	24060	77680	125514	38.1 %
scr20	5079	15980	61780	446686	6561431	93.2 %

Table 5.1: Comparing the number of nonzero elements in the LU factorisation of the basis B and in the Cholesky factorisation of the normal equations matrix $A\Theta A^T$.

Problem	Direct approach		Mixed approach			Time
	Time	IPM-iters	Time	IPM-iters	IPM-pcg	saving
aircraft	33.15	17	24.94	17	5	24.8 %
chr12a	0.304	14	0.290	14	2	4.61 %
chr12b	0.402	16	0.354	16	3	11.9 %
chr12c	0.256	11	0.254	11	1	0.78 %
chr15b	1.263	17	1.196	17	2	5.80 %
chr15c	1.231	17	1.194	17	2	3.03 %
chr18a	6.480	29	5.747	30	5	11.3 %
chr18b	3.520	16	3.213	16	3	8.72 %
chr20a	13.69	28	9.292	28	14	23.1 %
chr20b	11.31	27	9.895	27	8	12.5 %
chr20c	11.91	23	11.76	23	4	1.26 %
chr22a	25.59	28	24.73	28	2	3.36 %
chr22b	48.78	52	27.09	33	2	44.5 %
chr25a	81.04	39	71.92	39	5	11.3 %
fit1p	3.49	20	2.01	20	9	42.2 %
fit2p	583.33	25	211.93	25	12	63.7 %
fome10	281.96	45	124.01	43	17	56.0 %
fome11	827.85	48	288.44	44	17	65.2 %
fome12	1646.29	48	604.98	44	17	63.3 %
pds-06	60.81	44	28.12	43	21	57.8 %
pds-10	198.08	38	103.34	53	29	47.8 %
pds-20	2004.87	47	770.83	66	38	61.6 %
route	53.98	25	48.99	24	4	9.20 %
scr10	0.839	19	0.685	19	8	18.4 %
scr12	3.092	14	2.951	14	2	18.8 %
scr15	50.79	26	41.22	26	7	18.8 %
scr20	614.56	25	517.62	26	4	15.8 %

Table 5.2: Solution statistics.

Problem	PCG Iterations			
	lastIPM	min	average	max
aircraft	10	8	9	10
chr12a	19	18	20	23
chr12b	29	28	29	29
chr12c	26	26	26	26
chr15b	33	31	38	36
chr15c	32	31	32	32
chr18a	37	35	37	38
chr18b	57	53	56	57
chr20a	39	38	56	82
chr20b	32	32	63	104
chr20c	45	42	44	45
chr22a	48	46	49	53
chr22b	45	39	42	46
chr25a	51	46	50	55
fit1p	2	2	3	6
fit2p	4	3	15	43
fome10	142	129	243	519
fome11	169	123	205	494
fome12	111	111	210	500
pds-06	60	36	53	71
pds-10	66	45	60	86
pds-20	111	44	78	145
route	85	30	60	92
scr10	19	16	19	23
scr12	44	44	45	45
scr15	43	43	61	78
scr20	200	141	181	291

Table 5.3: The number of PCG iterations during the interior point method iterations.

Problem	Direct approach		Iterative approach		Time
	Time	IPM-iters	Time	IPM-iters	saving
aircraft	33.15	17	2.87	15	91.3 %
chr12a	0.304	14	0.449	14	-47.7 %
chr12b	0.402	16	0.306	14	23.9%
chr12c	0.256	11	0.254	11	1.01%
chr15b	1.263	17	0.944	16	25.3 %
chr15c	1.231	17	0.959	18	22.1 %
chr18a	6.480	29	3.119	29	51.9 %
chr18b	3.520	16	2.255	18	35.9 %
chr20a	13.69	28	5.721	34	58.2 %
chr20b	11.31	27	5.721	30	49.4 %
chr20c	11.91	23	4.800	22	59.7 %
chr22a	25.59	28	6.725	31	73.7 %
chr22b	48.78	52	8.232	36	83.1 %
chr25a	81.04	39	17.54	41	78.4 %
fit1p	3.49	20	0.38	19	89.1 %
fit2p	583.33	25	19.09	26	96.7 %
fome10	281.96	45	126.72	47	19.6 %
fome11	827.85	48	437.93	51	74.02 %
fome12	1646.29	48	-	-	Unsolved
pds-06	60.81	44	98.80	44	-31.23%
pds-10	198.08	38	122.42	46	33.15%
pds-20	2004.87	47	-	-	Unsolved
scr10	0.839	19	0.633	19	24.6 %
scr12	3.092	14	1.701	15	96.7 %
scr15	50.79	26	16.55	26	67.4 %
scr20	614.56	25	-	-	Unsolved

Table 5.4: Efficiency of the pure iterative method.

Problem	Dimensions			Direct approach		Mixed approach		
	m	n	nz(A)	Time	IPM-iters	Time	IPM-iters	IPM-pcg
80bau3b	2235	14269	24883	2.209	50	5.172	50	14
agg	3754	7517	24034	0.179	20	0.277	26	12
bore3d	233	567	1679	0.064	23	0.059	23	2
chr15a	1814	3270	11460	1.274	17	1.316	22	9
dbir2	18879	64729	1177011	310.7	38	225.8	39	11
gfrd-pnc	616	1776	3061	0.100	18	0.123	18	13
pds-02	2953	10488	19424	1.476	31	3.213	34	15
qap8	912	2544	8208	2.183	10	2.380	10	1
nw14	73	123482	904983	24.12	45	46.04	50	27
scorpion	388	854	1922	0.056	16	0.053	16	1
shell	536	2313	3594	0.150	21	0.407	43	21
ship04l	360	2526	6740	0.123	16	0.142	16	3
ship04s	360	1866	4760	0.099	16	0.117	16	5
stocfor1	117	282	618	0.024	20	0.057	20	11
stocfor2	2157	5202	11514	0.582	36	1.829	36	10
storm8	4393	15715	32946	4.541	52	8.691	54	18

Table 5.5: Solution statistics for problems, which do not benefit of iterative approach.

Problem	Nonzeros in Factors		PCG Iterations		
	nz(B)	nz(L)	min	average	max
80bau3b	5800	42709	29	64	226
agg	1589	16629	3	24	45
bore3d	821	2941	17	17	17
chr15a	10533	218060	37	38	41
dbir2	51609	2869915	50	74	93
gfrd-pnc	1240	1798	11	13	15
pds-02	6422	40288	38	48	58
qap8	60553	193032	175	175	175
nw14	443	1968	6	??	15
scorpion	1559	2102	38	38	38
shell	1075	4096	3	25	45
ship04l	941	4428	10	12	14
ship04s	938	3252	10	11	13
stocfor1	302	903	8	29	46
stocfor2	6585	33207	32	96	325
storm8	9805	136922	42	64	85

Table 5.6: Comparing the number of nonzero elements in the factorisations and the number of PCG iterations during IPM.

Chapter 6

Conclusions

In this thesis we have discussed interior point method for linear programming problems. At each iteration of the IPM at least one linear system has to be solved. The main computational effort of interior point algorithms consists in the computation of these linear systems. Every day optimization problems become larger. Solving the corresponding linear systems with a direct method becomes sometimes very expensive for large problems. In this thesis, we have been concerned with using an iterative method to solve these linear systems. In Chapter 2 we have reviewed some of the popular solution methods of these linear systems (direct methods and iterative method).

In this thesis we have used the PCG method to solve the (indefinite) augmented system (1.7), which arises from interior point algorithms for linear programming. We have proposed in Chapter 3 a new sparse preconditioner for the augmented system. This preconditioner takes advantage of the fact that a subset of elements in the matrix Θ^{-1} converge to zero as the solution of the linear program is approached. We replace these elements with zeros in the preconditioner. As a result, we have obtained a sparse and easily invertible block-triangular matrix. The constraint matrix A has been partitioned into

$[B, N]$, where B is an m by m nonsingular matrix. The matrix B is obtained from m linearly independent columns of A which correspond to small θ_j^{-1}. By following the analysis of Rozlozník and Simoncini [65] closely, we have shown that the PCG method can be applied to a non-symmetric indefinite matrix for a specific starting point. In addition, we have analysed the behaviour of the error and residual terms. This analysis reveals that, although we work with the indefinite system preconditioned with the indefinite matrix, the error and residual converge to zero and, asymptotically, behave in a similar way to the classical case when PCG is applied to a positive definite system.

The use of an iterative method in this context makes an essential difference in the implementation of the interior point algorithm. This requires a better understanding of IPM convergence properties in a situation when directions are inexact. In Chapter 4 we have considered the convergence analysis of the inexact infeasible path-following algorithm, where the augmented system is solved iteratively, according to what have been mentioned earlier. We have used a trick which consisted in shifting the residual from the dual constraint to the perturbed complementarity constraint. This has allowed us to modify the analysis of the (exact) infeasible IPM [77, 81] and generalize it to the *inexact* case. We have chosen a suitable stopping criteria of the PCG method used in this context and have provided a condition on the forcing term. Furthermore, we have proved the global convergence of the IIPF algorithm and have provided a complexity result for this method.

Finally, in Chapter 5 we have illustrated the feasibility of our approach on a set of medium to large-scale linear problems. Based on these results we conclude that it is advantageous to apply the preconditioned conjugate gradient method to indefinite KKT systems arising in interior point algorithms for linear programming.

There are many research possibilities of interest still to explore in this area. The approach proposed in this thesis has proved to work well. However, in its current form it is limited to the linear programming case. One of the possible developments is to extend this approach to the quadratic and nonlinear programming problems.

Bibliography

[1] G. Al-Jeiroudi and J. Gondzio, *Convergence analysis of inexact infeasible interior point method for linear optimization*, (accepted for publication in Journal on Optimization Theory and Applications), (2007).

[2] G. Al-Jeiroudi, J. Gondzio, and J. Hall, *Preconditioning indefinite systems in interior point methods for large scale linear optimization*, Optimization Methods and Software, 23 (2008), pp. 345–363.

[3] 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, T. Terlaky, ed., Kluwer Academic Publishers, 1996, pp. 189–252.

[4] M. Arioli, I. S. Duff, and P. P. M. de Rijk, *On the augmented system approach to sparse least-squares problems*, Numerische Mathematik, 55 (1989), pp. 667–684.

[5] V. Baryamureeba and T. Steihaug, *On the convergence of an inexact primal-dual interior point method for linear programming*, in Lecture Notes in Computer Science, Springer Berlin/Heidelberg, 2006.

[6] S. Bellavia, *An inexact interior point method*, Journal of Optimization Theory and Applications, 96 (1998), pp. 109–121.

[7] S. Bellavia and S. Pieraccini, *Convergence analysis of an inexact infeasible interior point method for semidefinite programming*, Computational Optimization and Applications, 29 (2004), pp. 289–313.

[8] M. Benzi, G. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numerica, 14 (2005), pp. 1–137.

[9] L. Bergamaschi, J. Gondzio, M. Venturin, and G. Zilli, *Inexact constraint preconditioners for linear systems arising in interior point methods*, Computational Optimization and Applications, 36 (2007), pp. 137–147.

[10] L. Bergamaschi, J. Gondzio, and G. Zilli, *Preconditioning indefinite systems in interior point methods for optimization*, Computational Optimization and Applications, 28 (2004), pp. 149–171.

[11] M. W. Berry, M. T. Heath, I. Kaneko, M. Lawo, and R. J. Plemmon, *An algorithm to compute a sparse basis of the null space*, Numerische Mathematik, 47 (1985), pp. 483–504.

[12] A. Bjorck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

[13] S. Bocanegra, F. Campos, and A. Oliveira, *Using a hybrid preconditioner for solving large-scale linear systems arising from interior point methods*, Computational Optimization and Applications, 36 (2007), pp. 149–164.

[14] W. J. Carolan, J. E. Hill, J. L. Kennington, S. Niemi, and S. J. Wichmann, *An empirical evaluation of the KORBX algorithms for military airlift applications*, Operations Research, 38 (1990), pp. 240–248.

[15] T. Carpenter and D. Shanno, *An interior point method for quadratic programs based on conjugate projected gradients*, Computational Optimization and Applications, 2 (1993), pp. 5–28.

[16] J. S. Chai and K. C. Toh, *Preconditioning and iterative solution of symmetric indefinite linear systems arising from interior point methods for linear programming*, Computational Optimization and Applications, 36 (2007), pp. 221–247.

[17] T. F. Coleman and A. Pothen, *The null space problem I. complexity*, SIAM Journal on Algebraic and Discrete Methods, 7 (1986), pp. 527–537.

[18] ——, *The null space problem II. algorithms*, SIAM Journal on Algebraic and Discrete Methods, 7 (1986), pp. 544–562.

[19] T. F. Coleman and A. Verma, *A preconditioned conjugate gradient approach to linear equality constrained minimization*, Computational Optimization and Applications, 20 (2001), pp. 61–72.

[20] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM Journal on Numerical Analysis, 19 (1982), pp. 400–408.

[21] H. Dollar, N. Gould, and A. Wathen, *On implicit-factorization constraint preconditioners*, in Large-Scale Nonlinear Optimization, G. D. Pillo, ed., Springer Netherlands, 2006.

[22] H. Dollar and A. Wathen, *Approximate factorization constraint preconditioners for saddle-point matrices*, SIAM Journal on Scientific Computing, 27 (2005), pp. 1555–1572.

[23] H. S. Dollar, N. I. M. Gould, W. H. A. Schilders, and A. J. Wathen, *Implicit-factorization preconditioning and iterative solvers for regularized saddle-point systems*, SIAM Journal on Matrix and Applications, 28 (2006), pp. 170–189.

[24] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct methods for sparse matrices*, Oxford University Press, New York, 1987.

[25] B. Fischer, *Polynomial Based Iteration Methods for Symmetric Linear Systems*, Wiley-Teubner, Chichester and Stuttgart, 1996.

[26] R. Fletcher, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis Dundee 1975, G. Watson, ed., Springer-Verlag, Berlin, New York, 1976, pp. 73–89.

[27] A. Forsgren, P. E.Gill, and M. H.Wright, *Interior point methods for nonlinear optimization*, SIAM Review, 44 (2002), pp. 525–597.

[28] R. W. Freund and F. Jarre, *A QMR-based interior-point algorithm for solving linear programs*, Mathematical Programming, 76 (1997), pp. 183–210.

[29] R. W. Freund, F. Jarre, and S. Mizuno, *Convergence of a class of inexact interior-point algorithms for linear programs*, Mathematics of Operations Research, 24 (1999), pp. 105–122.

[30] D. M. Gay, *Electronic mail distribution of linear programming test problems*, Mathematical Programming Society COAL Newsletter, 13 (1985), pp. 10–12.

[31] A. George and J. W. H. Liu, *The evolution of the minimum degree ordering algorithm*, SIAM Review, 31 (1989), pp. 1–19.

[32] ——, *Computing solution of large sparse positive definite systems*, Prentice-Hall, Englewood Cliffs, (NJ, 1981).

[33] J. C. Gilbert and J. Nocedal, *Global convergence properties of conjugate gradient methods for optimization*, SIAM Journal on Optimization, 2 (1992), pp. 21–42.

[34] P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders, *Preconditioners for indefinite systems arising in optimization*, SIAM Journal on Matrix Analysis and Applications, 13 (1992), pp. 292–311.

[35] J. Gondzio, *Implementing Cholesky factorization for interior point methods of linear programming*, Optimization, 27 (1993), pp. 121–140.

[36] ——, *HOPDM (version 2.12) – a fast LP solver based on a primal-dual interior point method*, European Journal of Operational Research, 85 (1995), pp. 221–225.

[37] J. Gondzio and T. Terlaky, *A computational view of interior point methods for linear programming*, In J. E. Beasley, editor, Advances in Linear and Integer Programming, chapter 3, Oxford University Press, Oxford, England, (1994), pp. 103–144.

[38] C. Gonzaga, *Path-following methods in linear programming*, SIAM Review, 34 (1992), pp. 167–224.

[39] C. Gonzaga and M. J. Todd, *An $O(\sqrt{n}L)$-iteration large-step primal-dual affine algorithm for linear programming*, SIAM Journal on Optimization, 2 (1992), pp. 349–359.

[40] N. I. M. Gould, M. E. Hribar, and J. Nocedal, *On the solution of equality constrained quadratic problems arising in optimization*, SIAM Journal on Scientific Computing, 23 (2001), pp. 1375–1394.

[41] J. A. J. Hall and K. I. M. Mckinnon, *Hyper-sparsity in the revised simplex method and how to exploit it*, Computational Optimization and Applications, 32 (2005), pp. 259–283.

[42] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, Journal of Research of Natlional Bureau of Standards, 49 (1952), pp. 409–436.

[43] K. R. James and W. Riha, *Convergence criteria for successive overrelaxation*, SIAM Journal on Numerical Analysis, 12 (1975), pp. 137–143.

[44] J. J. Júdice, J. Patricio, L. F. Portugal, M. G. C. Resende, and G. Veiga, *A study of preconditioners for network interior point methods*, Computational Optimization and Applications, 24 (2003), pp. 5–35.

[45] N. Karmarkar and K. Ramakrishnan, *Computational results of an interior point algorithm for large scale linear programming*, Mathematical Programming, 52 (1991), pp. 555–586.

[46] C. Keller, N. I. M. Gould, and A. J. Wathen, *Constraint preconditioning for indefinite linear systems*, SIAM Journal on Matrix Analysis and Applications, 21 (2000), pp. 1300–1317.

[47] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, vol. 16 of Frontiers in Applied Mathematics, SIAM, Philadelphia, 1995.

[48] V. Klee and G. J. Minty, *How good is the simplex algorithm?, in inequalities iii*, O. Shisha, ed., Academic Press, London, New York, (1972), pp. 159–175.

[49] J. Korzak, *Convergence analysis of inexact infeasible-interior-point-algorithm for solving linear progamming problems*, SIAM Journal on Optimization, 11 (2000), pp. 133–148.

[50] Z. Lu, R. D. S. Monteiro, and J. W. O'Neal, *An iterative solver-based infeasible primal-dual path-following algorithm for convex QP*, SIAM Journal on Optimization, 17 (2006), pp. 287–310.

[51] L. Lukšan and J. Vlček, *Indefinitely preconditioned inexact Newton method for large sparse equality constrained nonlinear programming problems*, Numerical Linear Algebra with Applications, 5 (1998), pp. 219–247.

[52] I. Lustig, R. Marsten, and D. Shanno, *Computational experience with a primaldual interior point method for linear programming*, Linear Algebra and its Applications, 152 (1991), pp. 191–222.

[53] ——, *Interior point methods for linear programming: computational state of the art*, ORSA Journal on Computing, 6 (1994), pp. 1–14.

[54] S. Mehrotra, *Implementation of affine scaling methods: Approximate solutions of systems of linear equations using preconditioned conjugate gradient methods*, Journal on Computing, 4 (1992), pp. 103–118.

[55] S. Mehrotra and J. S. Wang, *Conjugate gradient based implementation of interior point methods for network flow problems*, in Linear and Nonlinear Conjugate Gradient-Related Methods, L. Adams and J. L.

Nazareth, eds., AMS-IMS-SIAM Joint Summer Research Conference, 1995.

[56] J. A. Meijerink and H. A. V. D. Vorst, *An iterative solution method for linear systems of which the coefficient matrix is symmetric M-matrix*, Mathematics of Computation, 31 (1977), pp. 148–162.

[57] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, 2000.

[58] S. Mizuno and F. Jarre, *Global and polynomial-time convergence of an infeasible-interior-point algorithm using inexact computation*, Mathematical Programming, 84 (1999), pp. 105–122.

[59] R. D. S. Monteiro and J. W. O'Neal, *Convergence analysis of long-step primal-dual infeasible interior point LP algorithm based on iterative linear solvers*, Georgia Institute of Technology, (2003).

[60] A. R. L. Oliveira and D. C. Sorensen, *A new class of preconditioners for large-scale linear systems from interior point methods for linear programming*, Linear Algebra and its Applications, 394 (2005), pp. 1–24.

[61] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[62] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM Journal on Numerical Analysis, 12 (1975), pp. 617–629.

[63] A. Pothen, *Sparse null space basis computations in structural optimization*, Numerische Mathematik, 55 (1989), pp. 501–519.

[64] M. G. C. Resende and G. Veiga, *An implementation of the dual affine scaling algorithm for minimum cost flow on bipartite uncapacitated networks*, SIAM Journal on Optimization, 3 (1993), pp. 516–537.

[65] M. Rozložník and V. Simoncini, *Krylov subspace methods for saddle point problems with indefinite preconditioning*, SIAM Journal of Matrix Analysis and Applications, 24 (2002), pp. 368–391.

[66] Y. Saad, *Iterative Method for Sparse Linear System, Second Edition*, SIAM, Philadelphia, 1995.

[67] Y. Saad and M. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM Journal on Scientific and Statistical Computing, 7 (1986), pp. 856–869.

[68] J. Shewchuk, *An introduction to the conjugate gradient method without the agonizing pain*, tech. report, School of Computer Science, Carnegie Mellon University, USA, 1994.

[69] J. A. Tomlin, *Pivoting for size and sparsity in linear programming inversion routes*, Journal of Mathematics and Applications, 10 (1972), pp. 289–295.

[70] C. H. Tong and Q. Ye, *Analysis of the finite precision Bi-conjugate gradient algorithm for nonsymmetric linear systems*, Mathematics of Computation, 69 (1999), pp. 1559–1575.

[71] L. N. Trefethen and D. Bau, *III, Numerical linear algebra, Society for Industrial and Applied Mathematics*, SIAM, Philadelphia, 1997.

[72] H. A. van der Vorst, *Iterative Krylov methods for large linear systems*, Cambridge University Press, Cambridge, (2003).

[73] R. Vanderbei, *LOQO : An interior point code for quadratic programming, program in statistics and operations research*, Princeton University, (1995).

[74] R. S. Varga, *Matrix Iterative Analysis*, Englewood Cliffs, NJ, 1962.

[75] W. Wang and D. P. O'Leary, *Adaptive use of iterative methods in predictor-corrector interior point methods for linear programming*, Numerical Algorithms, 25 (2000), pp. 387–406.

[76] M. H. Wright, *The interior-point revolution in optimization: history, recent developments, and lasting consequences*, American Mathematical Society, 42 (2004), pp. 39–65.

[77] S. J. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997.

[78] X. Xu, *An $O(\sqrt{n}L)$-iteration large-step infeasible path-following algorithm for linear programming*, Technical report, University of Lowa, (1994).

[79] Y. Ye, *Interior-point algorithm: theory and analysis*, John Wiley and Sons, New York, 1997.

[80] D. M. Young, *Iterative Soluation of Large Linear Systems*, Academic Press, New York, 1971.

[81] Y. Zhang, *On the convergence of a class of infeasible interior-point methods for the horizontal linear complementarity problem*, SIAM Journal on Optimization, 4 (1994), pp. 208–227.

[82] G. Zhou and K. C. Toh, *Polynomiality of an inexact infeasible interior point algorithm for semidefinite programming*, Mathematical Programming, 99 (2004), pp. 261–282.