NUMERICAL SOLUTION OF SEMIDEFINITE CONSTRAINED LEAST SQUARES PROBLEMS

by

NATHAN GAVIN BEAUREGARD KRISLOCK

B.Sc. Hon. (Combined Math/Computer Science) University of Regina, 2000

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

Department of Mathematics
Institute of Applied Mathematics

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

May 2003

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Mathematics
The University of British Columbia
Vancouver, Canada

Date April 19, 2003

Abstract

In this thesis, we are concerned with computing the least squares solution of the linear matrix equation

$$AX = B$$

subject to the constraint that the matrix X is positive semidefinite. For symmetric X, this is the previously studied *semidefinite least squares* (*SDLS*) *problem*; for nonsymmetric X, we introduce the *nonsymmetric semidefinite least squares* (*NS-SDLS*) *problem.* An application of this second problem is the estimation of the compliance matrix at some location on a deformable object. This is an important step in the process of making interactive computer models of deformable objects, a technique which is used in the area of medical simulation. We also introduce a third semidefinite constrained least squares problem called the *linear matrix inequality least squares* (*LMI-LS*) *problem*, which is a generalization of the first two problems.

Sufficient conditions for the existence and uniquenss of solutions for each of these three problems is provided. These solutions are characterized as solutions of nonlinear systems of equations known as the *Karush-Kuhn-Tucker* (*KKT*) *equations.* It is shown that the KKT equations for each of these problems can be stated as a *semidefinite linear complementarity problem* (*SDLCP*). Interior-point methods are proposed for the numerical solution of each of these three problems. Computational experiments are conducted which indicate that predictor-corrector interior-point methods solve these semidefinite constrained least squares problems efficiently. A noticable improvement is made over the current computational methods used for solving the SDLS problem.

Table of Contents

List of Tables

List of Figures

Acknowledgements

This thesis is dedicated to my late father, Brian Russel Krislock.

I wish to give thanks to my supervisor, Dr. James Varah, and my reader, Dr. Philip Loewen, for all the effort and advice they provided me during my studies and research at the University of British Columbia. I am also incredibly grateful to Dr. Jochen Lang and Dr. Dinesh Pai for providing me with such an interesting and exciting problem as the topic for my thesis. A special thanks is given to all my friends at St. John's College at UBC who provided me with some of the best years of my life. Of course, none of this would have been possible without the love and support of my family, especially my wonderful parents and brother. Thank you.

Chapter 1
Introduction

There are often times in which one would like to find a solution to a problem for which no exact solution exists. We would still like to be able to give such a problem to a computer and have it respond with more than just "`There is no exact solution.`" It is preferable to have the computer let us know that there is no exact solution, but provide us with a solution which gives us the smallest error possible in our problem. In addition, there are many times in which we would like to compute such a solution with least error for a problem which requires, for example, that the variables of this solution cannot be negative. We must find a way to have the computer provide us with a solution that satisfies our problem as close as is possible subject to such a constraint.

In this thesis, we are concerned with the problem of solving the linear matrix equation which is defined as

$$AX = B \tag{1.1}$$

where A and B are rectangular matrices and the variable X is a square matrix. There is often no exact solution to this equation, but we are still interested in a solution which minimizes the sum of the squares of the components of the matrix $AX - B$. Such a solution is called a *least squares solution* of this equation. In certain applications there are constraints that must be satisfied, constraints which the standard least squares solution does not usually satisfy. For example, we could require that all the entries of the matrix X be nonnegative, or that the matrix X be symmetric, or both.

We focus our attention on two possible constraints which we can place on the matrix X when finding the least squares solution of equation (1.1). The first is to insist that X be symmetric and have nonnegative eigenvalues; that is, to constrain X to be *positive semidefinite*. In the second case, we allow X to be nonsymmetric, but constrain the symmetric part of X, $\frac{1}{2}(X + X^T)$, to be positive semidefinite. We also consider a third problem which is a generalization of the first two problems. This problem seeks a least squares solution of the linear equation

$$Ax = b \tag{1.2}$$

where A is a rectangular matrix and x and b are vectors. In this case we constrain the variable x by insisting that a linear combination of symmetric matrices, which is defined by x, be positive semidefinite. We now define each of these three semidefinite constrained least squares problems explicitly.

1.1 The Semidefinite Constrained Least Squares Problems

The first of our three problems is called the *symmetric semidefinite least squares (SDLS) problem*, and is defined as follows.

$$\begin{array}{ll} \text{minimize} & \|AX - B\|_F \\ \text{subject to} & X \succeq \mathbf{0} \end{array} \tag{1.3}$$

Here we use the notation $X \succeq \mathbf{0}$ to mean that X is symmetric and positive definite. Moreover, $\|\cdot\|_F$ is the *Frobenius matrix norm* which is used to measure the size of the *residual matrix*, $AX - B$, for a given matrix X. We will define this norm in Chapter 2, but for now it is enough to observe that $\|AX - B\|_F > 0$ if X does not satisfy $AX = B$; if X satisfies $AX = B$, then $\|AX - B\|_F = 0$. Therefore, the SDLS problem is the problem of finding a symmetric $n \times n$ matrix X which minimizes this measure of the error in the equation $AX = B$, for some $m \times n$ matrices A and B, subject to the constraint that X be positive definite.

We call our second problem the *nonsymmetric semidefinite least squares (NS-SDLS) problem*, and we define it as follows.

$$\begin{array}{ll} \text{minimize} & \|AX - B\|_F \\ \text{subject to} & \frac{1}{2}(X + X^T) \succeq \mathbf{0} \end{array} \tag{1.4}$$

This time we are looking for an $n \times n$ matrix X, which is possibly nonsymmetric, and which minimizes the Frobenius norm of the residual matrix $AX - B$, where A and B are some $m \times n$ matrices, subject to the constraint that $\frac{1}{2}(X + X^T)$ be positive semidefinite.

Finally, our third problem is defined as

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|_2 \\ \text{subject to} & \mathcal{K}x \preceq C \end{array} \tag{1.5}$$

where

$$\mathcal{K}x = \sum_{i=1}^{n} x_i K_i$$

and $K_1, \ldots, K_n$ and C are $k \times k$ symmetric matrices. The constraint that $\mathcal{K}x \preceq C$ means that we require that the $k \times k$ symmetric matrix, $C - \mathcal{K}x$, be positive semidefinite. Since this type of constraint is commonly known as a *linear matrix inequality*, we call this problem the *linear matrix inequality least squares (LMI-LS) problem.* Here A is an $m \times n$ matrix, b is a vector of length m, and x is a vector of length n. This time we measure the size of the residual vector, $Ax - b$, using the *Euclidean vector norm*, $\|\cdot\|_2$, which we will define in Chapter 2.

We would like to note that while the SDLS problem has been previously studied in great detail, this seems to be the first time that the NS-SDLS and LMI-LS problems have been formulated and studied. Furthermore, as we will see, each of these problems are convex optimization problems whose local minimizers are global minimizers, and, under the condition of strict convexity, there is at most one global minimizer.

1.2 An Application

The inspiration behind the study of these semidefinite constrained least squares problems began with the need to solve the NS-SDLS problem in order to estimate the compliance (or stiffness) matrix of a deformable (or elastic) object (see [15]). This compliance matrix estimation is important in many areas including robotics, medical simulation, and computer graphics. The idea is to use this compliance matrix to be able to model a deformable object, such as a human organ, and interact with its virtual model on a computer. This interaction is done by using some computer interface which allows users to touch this virtual object, and which also provides some force-feedback depending on the elasticity of the object.

The method used for estimating the compliance matrix at a certain location on a deformable object is to experimentally measure the displacement vector u resulting from some force, represented by the vector p, applied to the object at this location. The compliance matrix X for this contact point governs the relationship between u and p:

$$u = Xp. \tag{1.6}$$

In order to estimate the compliance matrix at this contact point, many displacement measurements, $u^1, \ldots, u^l$, are taken for the same number of different forces, $p^1, \ldots, p^l$. We then want to find the matrix X which satisfies

$$[p^1 \cdots p^l]^T X^T = [u^1 \cdots u^l]^T. \tag{1.7}$$

Due to unavoidable errors in this recording process, we will not be able to find the matrix X which satisfies equation (1.7) exactly. Therefore, we aim to find a matrix X which minimizes the error in this equation.

However, a problem occurs when using the simple unconstrained least squares solution of equation (1.7). If the compliance matrix X has an eigenvalue with a negative real part, this may cause the virtual model to respond to the user touching that contact point by pulling the user's hand further in the direction of the touch. By insisting that $\frac{1}{2}(X+X^T)$ is positive definite, none of the real parts of the eigenvalues of X will be negative. Therefore, we are interested in solving the NS-SDLS problem in order to obtain a compliance matrix which produces a well-behaved virtual model for the deformable object. For more on this topic, see [14].

1.3 Numerical Solutions

The purpose of this thesis is to present algorithms for computing approximate solutions to each of these three problems. We would like these approximate solutions to agree with the exact solution to within a prescribed accuracy, and we would also like to compute these approximate solutions efficiently; i.e., the algorithm should not take too long to run on a computer.

The methods we choose to implement are the popular *interior-point methods* which have been used to solve many other similar constrained optimization problems. We choose these methods due to the great success they have had in solving these types of problems accurately and efficiently. Moreover, it does not appear that interior-point methods have been used in the design of algorithms for solving the SDLS problem before this.

1.4 Previous Research

Although neither the NS-SDLS problem nor the LMI-LS problem seem to have been previously studied, the SDLS problem was first proposed back in 1968 by Brock [7] for obtaining symmetric positive definite compliance matrices like those discussed in Section 1.2. Brock had proposed a method for computing a symmetric positive definite least squares solution to (1.1); however, the method he uses only returns the solution to

$$\begin{array}{ll} \text{minimize} & \|AX - B\|_F \\ \text{subject to} & X = X^T \end{array} \tag{1.8}$$

which is not always positive definite.

The SDLS problem was later reformulated and studied in 1988 by Allwright [3]. In that paper, Allwright proved that there is a unique solution to this problem if and only if the matrix A has full column rank, and proposed a computational method for approximating this solution under that condition. Allwright also mentions that although there does not seem to be a simple formula for the solution of (1.3), we do have the following result that was shown in the Ph.D. thesis of Woodgate [32].

- If $A = I$, then the solution of (1.3) is $X = \frac{1}{2}(B + B^T)_{(0)}$.

Here I represents the $n \times n$ identity matrix, and $M_{(0)}$ denotes the result of changing all of the negative eigenvalues of M to 0 in the eigenvalue decomposition of M. In 1990 Allwright provided a necessary and sufficient condition for the existence of a solution when A does not have full column rank in the follow-up paper [4] written jointly with Woodgate.

Woodgate went on in 1996 to provide a Lagrange multiplier type condition which characterized solutions of the SDLS problem in [33], although it was not stated in terms of Lagrange multipliers. This result was then used to provide expressions for the solution of the SDLS problem in a couple of special cases when A has full column rank:

- If AB^T is symmetric and positive definite, then the solution of (1.3) is

$$X = A^\dagger B = (A^T A)^{-1} A^T B.$$

- The solution of (1.3) is $X = \mathbf{0}$ if and only if $-A^T B - B^T A$ is positive definite.

Notice that $X = A^\dagger B$ is the solution to the unconstrained least squares problem

$$\min_X \|AX - B\|_F, \tag{1.9}$$

where $A^\dagger$ is the *Moore-Penrose pseudoinverse* of A. Woodgate also provided a computational method in [33] for solving the SDLS problem when A has full column rank.

In 1998, Woodgate proposed a new algorithm for solving the SDLS problem in [34] which is based on the related unconstrained problem (1.10).

$$\min_E \|AE^T E - B\|_F \tag{1.10}$$

The idea here is that X is symmetric and positive semidefinite if and only if $X = E^T E$ for some $n \times n$ matrix E. This approach was avoided in [3] and [33] due to the fact that (1.10) is no longer a convex optimization problem, which can make obtaining a global minimizer difficult. However, Woodgate proved in [34] that any local minimizer of (1.10) is actually a global minimizer. The resulting computational method showed great improvements over the previous methods from [3] and [33]. We will refer to this method as *Woodgate's Algorithm*, and we will use it as the basis for which we compare the interior-point methods described here for the SDLS problem.

Further progress was made in 1999 by Liao [16] in the area of determining an expression for the solution, X, of the SDLS problem under certain conditions. By considering the singular value decomposition of A, Liao was able to determine an expression for X under a fairly general assumption, which includes the case when the matrix $AB^T + BA^T$ is positive semidefinite. Notice that this is a more general result than the one provided in [33] for the case when AB^T is symmetric and positive definite. Liao also used this result to provide an expression for all of the solutions of a specific case of the SDLS problem, the *least squares inverse eigenvalue problem* for positive semidefinite matrices. This is the problem of determining the positive semidefinite matrices A for which, given some eigenvalues $\lambda_1, \ldots, \lambda_m$ with corresponding eigenvectors $v_1, \ldots, v_m$, minimize the norm of the eigenvalue equation $AV = V\Lambda$, where $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_m)$ and $V = [v_1 \cdots v_m]$.

$$\begin{array}{ll} \text{minimize} & \|AV - V\Lambda\|_F \\ \text{subject to} & A \succeq \mathbf{0} \end{array} \tag{1.11}$$

A related paper by Xie [35] provided the only example of the NS-SDLS problem (1.4) in the collection of literature we considered. In that paper, Xie studied the least squares inverse eigenvalue problem for nonsymmetric positive semidefinite matrices (1.12).

$$\begin{array}{ll} \text{minimize} & \|AV - V\Lambda\|_F \\ \text{subject to} & \frac{1}{2}(A + A^T) \succeq \mathbf{0} \end{array} \tag{1.12}$$

An interesting result which is found in [35] is the expression for the solution of the NS-SDLS problem when $A = I$:

- If $A = I$, then the solution of (1.4) is $X = \frac{1}{2}(B + B^T)_{(0)} + \frac{1}{2}(B - B^T)$.

Until now, we have only mentioned previous work done on semidefinite constrained least squares problems which are covered by the three problems which we are considering. However, there has also been much work done on problems that are generalizations of the SDLS problem. In 1995, Hu [11] proposed a computational method for solving the following matrix estimation problem.

$$\begin{array}{ll} \text{minimize} & \|AX - B\|_F \\ \text{subject to} & X = X^T, \\ & L \leq X \leq U, \\ & \lambda_{\min}(X) \geq \varepsilon > 0, \\ & X \in \mathcal{P} \end{array} \tag{1.13}$$

In (1.13), the inequality, $L \leq X \leq U$, is to be interpreted component-wise: $L_{ij} \leq X_{ij} \leq U_{ij}$, for all $i, j \in \{1, \ldots, n\}$. Furthermore, $\lambda_{\min}(X)$ represents the minimum eigenvalue of X, and $\mathcal{P}$

specifies a set of $n \times n$ matrices having a particular linear pattern. That is,

$$\mathcal{P} = \left\{ \mathcal{K}v = \sum_{i=1}^{m} v_i K_i \;\middle|\; v \in \mathbb{R}^m \right\},$$

where $K_1, \ldots, K_m$ are some symmetric $n \times n$ matrices. The idea behind Hu's method is to transform (1.13) into an equivalent convex quadratic program with infinitely many linear constraints, and then solve this problem by generating and solving a sequence of ordinary convex quadratic programs, adding a new linear constraint at every iteration.

Escalante and Raydan [9] also consider the matrix estimation problem (1.13), but with $A = I$. They propose using a different computational method, known as Dykstra's alternating projection algorithm. This method is based on viewing the constraints in (1.13) as the intersection of three sets, and performing a series of projections onto each of these sets until a desired tolerance is reached. This paper also contains the following interesting result:

- The solution to

$$\begin{array}{ll} \text{minimize} & \|X - B\|_F \\ \text{subject to} & X = X^T, \\ & \lambda_{\min}(X) \geq \varepsilon > 0 \end{array} \tag{1.14}$$

  is $X = \frac{1}{2}(B + B^T)_{(\varepsilon)}$, where $M_{(\varepsilon)}$ denotes the result of changing all of the eigenvalues less than ε to ε in the eigenvalue decomposition of M.

1.5 Outline of Thesis

We summarize the remainder of this thesis as follows. In Chapter 2 we provide a general discussion of convex optimization and develop the systems of equations (the *KKT conditions*) which will characterize the solutions of each of our three semidefinite constrained problems: the SDLS (1.3), the NS-SDLS (1.4), and the LMI-LS (1.5). This chapter also mentions the conditions which will guarantee the existence and uniqueness of solutions to our problems, and shows the connection of each of these problems to the *semidefinite linear complementarity problem* (*SDLCP*). Chapter 3 then goes on to describe the interior-point methods which we will be using to solve our three problems. Starting with some background about interior-point methods, Chapter 3 provides the intuition behind how we can define a *central path* which will be used as the basis for describing the interior-point algorithms for our problems. Due to the connection to the SDLCP, we are able to provide a uniform discussion of the *standard path-following* and *predictor-corrector* algorithms which we are considering; this discussion also includes some of the known theoretical convergence results for these two types of algorithms. Finally, Chapter 3 concludes by mentioning the issues involved in implementing the standard path-following algorithm and the predictor-corrector algorithm for each of our three problems. The results of the numerical experiments of our interior-point algorithms conducted in MATLAB, along with a comparison to Woodgate's Algorithm, will be presented in Chapter 4. We finish with our conclusions and ideas for future work in Chapter 5.

Chapter 2
Convex Analysis and Optimality Conditions

2.1 Preliminaries

We begin by discussing the framework in which the theoretical results will be presented. Since we will be working both in the space of vectors and in the space of matrices, it will be useful to generalize these spaces as is done in Borwein and Lewis' book [6]. There they define a *Euclidean space* $\mathbf{E}$ to be a finite-dimensional vector space over the reals with which there is an *inner product* $\langle \cdot, \cdot \rangle : \mathbf{E} \times \mathbf{E} \to \mathbb{R}$ that satisfies the following conditions ($x, y, z \in \mathbf{E}$; $\alpha, \beta \in \mathbb{R}$):

1. $\langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle$,
2. $\langle x, y \rangle = \langle y, x \rangle$,
3. $\langle x, x \rangle \geq 0$, where $\langle x, x \rangle = 0$ if and only if $x = \mathbf{0}$.

Examples of Euclidean spaces are $\mathbb{R}^n$ and $\mathbb{R}^{m \times n}$, which are the spaces of real column vectors $x = (x_1, \ldots, x_n)^T$ and real $m \times n$ matrices $A = (a_{ij})$, respectively. The standard inner product on $\mathbb{R}^n$ is defined as

$$\langle x, y \rangle := \sum_{i=1}^{n} x_i \, y_i = x^T y.$$

Similarly, the inner product we will use on $\mathbb{R}^{m \times n}$ will be defined as

$$\langle X, Y \rangle := \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} \, y_{ij} = \operatorname{tr}(X^T Y),$$

where $\operatorname{tr}(A) = \sum_{i=1}^{n} a_{ii}$ is the *trace* of a matrix $A \in \mathbb{R}^{n \times n}$.

The *norm* of an element x in $\mathbf{E}$ is defined to be $\|x\| := \sqrt{\langle x, x \rangle}$. This is the Euclidean norm in $\mathbb{R}^n$ and is denoted by $\|\cdot\|_2$. In $\mathbb{R}^{m \times n}$, this norm is the well known Frobenius norm, which is denoted by $\|\cdot\|_F$. When referring to topological properties in these spaces, it is with respect to these norms.

By the least squares solution to a matrix equation $AX = B$, where $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{m \times p}$, and $X \in \mathbb{R}^{n \times p}$, we mean a matrix X (or vector, if $p = 1$) such that the norm of the *residual matrix*, $R = B - AX$, is minimized. We notice that $\|AX - B\|_F$ is minimized exactly when the

sum of squares of the residuals,

$$\|AX - B\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}^2,$$

is minimized. It is also noticed (we will show this later) that if $f(X) = \frac{1}{2}\|AX - B\|_F^2$ and $g(X) = \|AX - B\|_F$, then

$$\nabla f(X) = A^T(AX - B) \quad \text{and} \quad \nabla g(X) = \frac{1}{\|AX - B\|_F} A^T(AX - B).$$

Thus, we can see that $\nabla f(X)$ is a simpler expression than $\nabla g(X)$ and avoids the difficulty arising when $\|AX - B\|_F = 0$. It is for these reasons that we will hence forth only deal with the *(normalized) sum of squares function*, $f(X) = \frac{1}{2}\|AX - B\|_F^2$, when finding the least squares solution of a matrix equation. For an account of the statistical justification behind the principle of least squares, see [5].

We will use $\mathcal{S}^n$ to denote the space of real symmetric $n \times n$ matrices A, where *symmetric* means $A = A^T$. We make $\mathcal{S}^n$ a Euclidean space by defining the inner product to be $\langle X, Y\rangle := \mathrm{tr}(XY)$ and we note that the dimension of $\mathcal{S}^n$ is $\frac{1}{2}n(n+1)$. A symmetric matrix A is called *positive semidefinite* if $x^T Ax \geq 0$ for all $x \in \mathbb{R}^n$, and *positive definite* if strict inequality holds whenever $x \neq \mathbf{0}$. The sets of positive semidefinite and positive definite matrices in $\mathcal{S}^n$ will be denoted respectively as $\mathcal{S}_+^n$ and $\mathcal{S}_{++}^n$. Notice that $\mathcal{S}_{++}^n$ is actually the interior of $\mathcal{S}_+^n$. For $X, Y \in \mathcal{S}^n$ we say $X \preceq Y$ if $Y - X \in \mathcal{S}_+^n$ and $X \prec Y$ if $Y - X \in \mathcal{S}_{++}^n$. This is known as the *Loewner partial ordering* of $\mathcal{S}^n$.

It is important to note that $\mathcal{S}_+^n$ is a cone in $\mathcal{S}^n$; a *cone* in $\mathbf{E}$ is a nonempty subset C that satisfies $C = \mathbb{R}_+ C$, where $\mathbb{R}_+ C := \{tx \mid t \in \mathbb{R}_+,\ x \in C\}$ and $\mathbb{R}_+ := \{t \in \mathbb{R} \mid t \geq 0\}$. Thus C is a cone if for every vector x in C, the ray from $\mathbf{0}$ through x is completely contained within C. Another example of a cone is the positive orthant in $\mathbb{R}^n$,

$$\mathbb{R}_+^n := \{x \in \mathbb{R}^n \mid x_i \geq 0 \ \text{ for } i = 1, \ldots, n\},$$

whose interior is the set $\mathbb{R}_{++}^n := \{x \in \mathbb{R}^n \mid x_i > 0 \ \text{ for } i = 1, \ldots, n\}$. For $x, y \in \mathbb{R}^n$, the partial ordering induced by $\mathbb{R}_+^n$ is written as $x \leq y$ if $y - x \in \mathbb{R}_+^n$ and $x < y$ if $y - x \in \mathbb{R}_{++}^n$.

We denote the *spectrum*, or *set of eigenvalues*, of a matrix $X \in \mathbb{R}^{n\times n}$ by $\sigma(X)$. If $X \in \mathcal{S}^n$, then

1. $\sigma(X) \subseteq \mathbb{R}$,
2. $X \in \mathcal{S}_+^n$ if and only if $\sigma(X) \subseteq \mathbb{R}_+$, and
3. $X \in \mathcal{S}_{++}^n$ if and only if $\sigma(X) \subseteq \mathbb{R}_{++}$.

A special feature of the cones $\mathbb{R}_+^n$ and $\mathcal{S}_+^n$ is the fact that they are self-dual. Let $H \subseteq \mathbf{E}$ be a cone. Then H is called *self-dual* if $H = H^*$, where

$$H^* = \{y \in \mathbf{E} \mid \langle x, y\rangle \geq 0 \ \text{ for all } x \in H\}$$

is the *dual cone* of H. As we will see later, the importance of self-dual cones in optimization is due to the fact that if H is self-dual, then

$$\inf_{x \in H} \langle x, y \rangle = \begin{cases} 0, & \text{if } y \in H \\ -\infty, & \text{otherwise.} \end{cases} \tag{2.1}$$

We will now summarize and prove these results in the following proposition.

Proposition 2.1.1 *Let H be a self-dual cone in a Euclidean space $\mathbf{E}$. Then equation (2.1) holds. Furthermore, $\mathbb{R}^n_+$ and $\mathcal{S}^n_+$ are self-dual cones.*

Proof. Let $y \in H$. Then $y \in H^*$, which implies that $\langle x, y \rangle \geq 0$ for all $x \in H$. Moreover, since H is a cone, $\mathbf{0} \in H$ and $\langle \mathbf{0}, y \rangle = 0$. Thus $\inf_{x \in H} \langle x, y \rangle = 0$. On the other hand, if $y \notin H$, then $y \notin H^*$, which implies that there exists an $x \in H$ such that $\langle x, y \rangle < 0$. Furthermore, $tx \in H$ for all $t > 0$ and $\langle tx, y \rangle = t \langle x, y \rangle < 0$. Now $\lim_{t \to \infty} t \langle x, y \rangle = -\infty$ implies $\inf_{x \in H} \langle x, y \rangle = -\infty$, which proves the first claim.

To show $\mathbb{R}^n_+$ and $\mathcal{S}^n_+$ are self-dual cones, we first note that both sets are clearly cones in their respective spaces. If $x, y \in \mathbb{R}^n_+$, then $x_i, y_i \geq 0$ for $i = 1, \ldots, n$, and thus $\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i \geq 0$. Therefore, $\mathbb{R}^n_+ \subseteq (\mathbb{R}^n_+)^*$. To show that $(\mathbb{R}^n_+)^* \subseteq \mathbb{R}^n_+$, suppose to the contrary that there is a $y \in (\mathbb{R}^n_+)^*$ such that $y \notin \mathbb{R}^n_+$. Then y has the properties that $\langle x, y \rangle \geq 0$ for all $x \in \mathbb{R}^n_+$ and that some $y_i < 0$. Take $x = \bar{e}_i$ (where $\bar{e}_i$ is the i^{th} unit vector: the vector with 1 in the i^{th} component, and zeros elsewhere). Then $x \in \mathbb{R}^n_+$ and $\langle x, y \rangle = y_i < 0$, which is a contradiction. Therefore, $y \in \mathbb{R}^n_+$.

To show that $\mathcal{S}^n_+$ is self-dual, we follow a proof given in [27, p. 520]. Let $X, Y \in \mathcal{S}^n_+$. Since X is positive semidefinite, it has a unique positive semidefinite square root, $\sqrt{X}$. Using this factorization of X and the fact that $\operatorname{tr}(A^T B) = \operatorname{tr}(B A^T)$ for all $A, B \in \mathbb{R}^{m \times n}$ (see [27, p. 521]), we get $\langle X, Y \rangle = \operatorname{tr}(\sqrt{X}\sqrt{X}Y) = \operatorname{tr}(\sqrt{X}Y\sqrt{X})$. Since Y is positive semidefinite, we know that $\sqrt{X}Y\sqrt{X}$ must also be positive semidefinite, and thus have real nonnegative eigenvalues. By the fact that the trace of a square matrix is equal to the sum of its eigenvalues, we get $\langle X, Y \rangle \geq 0$, showing that $\mathcal{S}^n_+ \subseteq (\mathcal{S}^n_+)^*$.

Suppose now that $Y \in (\mathcal{S}^n)^*$. Then $\langle X, Y \rangle \geq 0$ for all $X \in \mathcal{S}^n_+$. If $Y \notin \mathcal{S}^n_+$, then Y has a negative eigenvalue λ with eigenvector $u \in \mathbb{R}^n$. Let $X = uu^T$, then $X \in \mathcal{S}^n_+$ and $\langle X, Y \rangle = \operatorname{tr}(uu^T Y) = \operatorname{tr}(u^T Y u) = \lambda \|u\|_2 < 0$ gives the contradiction. Thus $Y \in \mathcal{S}^n_+$, proving that $(\mathcal{S}^n_+)^* \subseteq \mathcal{S}^n_+$. ∎

2.2 Convex Analysis

As we will see, the problems introduced in Chapter 1 fall under the scope of convex optimization. This is due to the fact that the sum of squares functions which we are considering are convex functions, and we are minimizing them over a convex set of points. As per the classical work of Rockafellar [25], we find that the convexity of sets and functions is a powerful concept in

optimization, largely due to the facts that a local minimizer of a convex function over a convex set is a global minimizer and that a minimizer of strictly convex function is unique. After defining these concepts in §2.2.1, we will state and prove these results in §2.2.2.

2.2.1 Convex sets and functions

We will begin by defining the concepts of convex sets and functions, after which we will show that the sum of squares functions that are the subject of this thesis are convex.

Definition 2.2.1 *A set Ω in a Euclidean space $\mathbf{E}$ is called convex if*

$$tx + (1-t)y \in \Omega, \quad \text{for all } x, y \in \Omega \text{ and } t \in [0,1].$$

Given a convex set Ω, a function $f : \Omega \to \mathbb{R}$ is called convex if

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y),$$

for all $x, y \in \Omega$ and $t \in [0,1]$. The function f is called strictly convex if the above inequality is strict whenever $x \neq y$ and $t \in (0,1)$.

Thus a set is convex if for any pair of points in the set, the line segment between that pair lies entirely inside the set. For example, the cones $\mathbb{R}^n_+$ and $\mathcal{S}^n_+$ are convex. A function is convex if between all pairs of points, x and y, the line segment connecting $(x, f(x))$ and $(y, f(y))$ lies on or above the graph of f at each point on the line segment between x and y (see Figure 2.1).

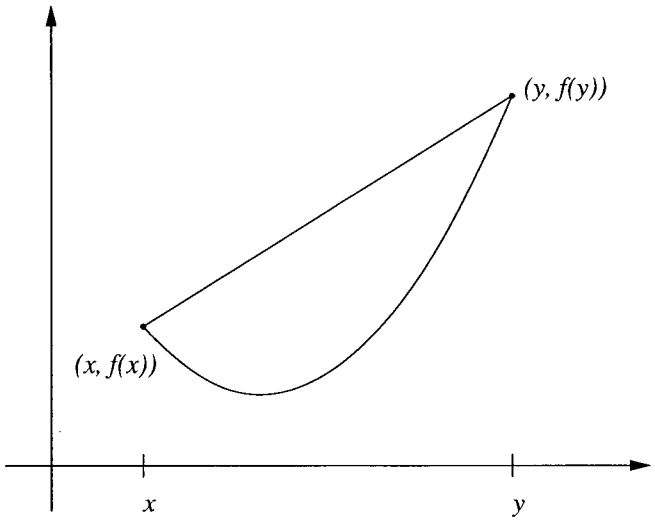


Figure 2.1: The graph of a convex function f.

Proposition 2.2.2 *Given $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{m \times p}$, the sum of squares function $f(X) = \frac{1}{2}\|AX - B\|_F^2$ is convex in $\mathbb{R}^{n \times p}$ and strictly convex if and only if A has full column rank.*

Proof. Here we give an extended version of a proof provided by Adlers [1, p. 5]. Let $X, Y \in \mathbb{R}^{n\times p}$ and $t \in [0,1]$. Since $f(X) = \frac{1}{2}\langle AX, AX\rangle - \langle AX, B\rangle + \frac{1}{2}\langle B, B\rangle$, then

$$\begin{aligned} f(tX + (1-t)Y) &= \tfrac{1}{2}t^2\langle AX, AX\rangle + t(1-t)\langle AX, AY\rangle + \tfrac{1}{2}(1-t)^2\langle AY, AY\rangle \\ &\quad -t\langle AX, B\rangle - (1-t)\langle AY, B\rangle + \tfrac{1}{2}\langle B, B\rangle \end{aligned}$$

which implies that

$$\begin{aligned} &[tf(X) + (1-t)f(Y)] - f(tX + (1-t)Y) \\ &\quad = \tfrac{1}{2}(t - t^2)\langle AX, AX\rangle - t(1-t)\langle AX, AY\rangle + \tfrac{1}{2}[(1-t) - (1-t)^2]\langle AY, AY\rangle \\ &\quad = \tfrac{1}{2}t(1-t)\langle AX - AY, AX - AY\rangle \\ &\quad = \tfrac{1}{2}t(1-t)\,\|A(X-Y)\|_F^2 \;\geq\; 0. \end{aligned}$$

If A has full column rank and $X \neq Y$, then $A(X-Y)$ can not be the zero matrix. Further, if $t \in (0,1)$ then the inequality above will be strict. However, if A does not have full column rank, then there exists a nonzero $U \in \mathbb{R}^{n\times p}$ such that $AU = \mathbf{0}$. Letting $X \in \mathbb{R}^{n\times p}$ be arbitrary, we can choose $Y = X - U$ so that $X \neq Y$ and $A(X-Y) = AU = \mathbf{0}$. In this case, even with $t \in (0,1)$, the inequality above is not strict, showing that f is only strictly convex when A has full column rank. ■

From Proposition 2.2.2 it also follows that $f(X) = \frac{1}{2}\|AX - B\|_F^2$ is convex over $\mathcal{S}^n$ since $\mathcal{S}^n$ is a convex subset of $\mathbb{R}^{n\times n}$. It is also useful to note that when $p = 1$ above, we get the result presented in [1] that $f(x) = \frac{1}{2}\|Ax - b\|_2^2$ is convex over $\mathbb{R}^n$.

2.2.2 Convex optimization

Given a convex set $\Omega \subseteq \mathbf{E}$ and a convex function $f : \Omega \to \mathbb{R}$, a *convex programming* problem is an optimization problem of the form

$$\inf\{f(x) \,|\, x \in \Omega\} \tag{2.2}$$

where we would like to determine the value of the infimum. If the infimum is attained in problem (2.2), then we would like to find a point $\bar{x} \in \Omega$ such that $f(\bar{x}) = \min\{f(x) \,|\, x \in \Omega\}$. Such an $\bar{x}$ is called a *global minimizer* of f over Ω, and is referred to as an *optimal solution* to the optimization problem. A *local minimizer* is a point $\hat{x} \in \Omega$ for which there is an open set $N \subseteq \mathbf{E}$ containing $\hat{x}$ such that $f(\hat{x}) = \min\{f(x) \,|\, x \in \Omega \cap N\}$. Any point x in Ω is called *feasible*; if x is in the interior of Ω, then x is called *strictly feasible*. A *feasible direction* at a point $x \in \Omega$ is a vector $v \in \mathbf{E}$ such that $x + tv \in \Omega$ for all $t > 0$ small enough.

Theorem 2.2.3 *Let $\hat{x} \in \Omega$ be a local minimizer of the convex programming problem (2.2). Then $\hat{x}$ is a global minimizer. If f is strictly convex, then $\hat{x}$ is the unique global minimizer.*

Proof. Suppose $\hat{x}$ is not a global minimum. Then there exists a $\hat{y} \in \Omega$ such that $f(\hat{y}) < f(\hat{x})$. Since f is convex,

$$\begin{aligned} f(t\hat{x} + (1-t)\hat{y}) &\leq tf(\hat{x}) + (1-t)f(\hat{y}) \\ &< tf(\hat{x}) + (1-t)f(\hat{x}) \\ &= f(\hat{x}), \end{aligned}$$

for all $t \in (0,1)$. Since $\hat{x}$ is a local minimizer, there exists an open set $N \subseteq \mathbf{E}$ containing $\hat{x}$ such that $f(\hat{x}) = \min\{f(x) \,|\, x \in \Omega \cap N\}$. But N being open implies that there exists a $t_0 \in (0,1)$ such that $t_0\hat{x} + (1-t_0)\hat{y} \in N$, contradicting the fact that $\hat{x}$ is a local minimizer. Therefore, no such $\hat{y}$ exists and $\hat{x}$ is a global minimizer.

Now suppose that f is strictly convex and that there exists another global minimizer $\hat{z} \in \Omega$, which means $f(\hat{z}) = f(\hat{x})$. Letting $t \in (0,1)$ we get

$$\begin{aligned} f(t\hat{x} + (1-t)\hat{z}) &< tf(\hat{x}) + (1-t)f(\hat{z}) \\ &= tf(\hat{x}) + (1-t)f(\hat{x}) \\ &= f(\hat{x}), \end{aligned}$$

contradicting the fact that $\hat{x}$ is a global minimizer. This implies that no such $\hat{z}$ exists and $\hat{x}$ is the unique global minimizer. ■

2.2.3 Cone constraints

For our purposes we will be specifying the constraint set Ω to be the set of points which are mapped by some "convex" function into either $\mathbb{R}^n_+$ or $\mathcal{S}^n_+$. By "convex" we mean convex in terms of the partial order induced by the closed convex cones $\mathbb{R}^n_+$ or $\mathcal{S}^n_+$. If $C \subseteq \mathbf{E}$ is some general closed cone, we write $x \leq_C y$ if $y - x \in C$ and $x <_C y$ if $y - x \in \operatorname{int} C$. Using this notation we have that $\leq$ represents $\leq_{\mathbb{R}^n_+}$ and $\preceq$ represents $\leq_{\mathcal{S}^n_+}$. It turns out that the binary relation $\leq_C$ is a partial ordering when C is convex and pointed, where *pointed* means that $C \cap -C = \{\mathbf{0}\}$.

Proposition 2.2.4 *If C is a pointed convex cone in $\mathbf{E}$, then $\leq_C$ is a partial ordering of $\mathbf{E}$.*

Proof. The binary relation $\leq_C$ is a partial ordering if $\leq_C$ is reflexive, antisymmetric, and transitive.

1. Reflexive: $x \leq_C x,\ \forall x \in \mathbf{E}$
   Since C is a cone we have $\mathbf{0} \in C$, which implies that $x - x \in C$ for all $x \in \mathbf{E}$, and so $\leq_C$ is reflexive.

2. Antisymmetric: $x \leq_C y$ and $y \leq_C x \Rightarrow x = y,\ \forall x, y \in \mathbf{E}$
   Given $x, y \in \mathbf{E}$ such that $x \leq_C y$ and $y \leq_C x$, we have $y - x \in C$ and $-(y-x) \in C$. Since C is pointed and $y - x \in C \cap -C$, we have $x = y$.

3. Transitive: $x \leq_C y$ and $y \leq_C z \Rightarrow x \leq_C z,\ \forall x, y, z \in \mathbf{E}$
   Let $x, y, z \in \mathbf{E}$ such that $x \leq_C y$ and $y \leq_C z$. Then $y - x \in C$ and $z - x \in C$, and the convexity of C implies $\frac{1}{2}(z-x) = \frac{1}{2}(y-x) + \frac{1}{2}(z-y) \in C$. Now, since C is a cone, we have $z - x \in C$. ■

A slightly different version of the following definition can be found in [6, p. 59]

Definition 2.2.5 *Let $\mathbf{E}$ and $\mathbf{Y}$ be Euclidean spaces and C a pointed closed convex cone in $\mathbf{Y}$. A function $g : \mathbf{E} \to \mathbf{Y}$ is called C-convex if*

$$g(tx + (1-t)y) \leq_C tg(x) + (1-t)g(y),$$

for all $x, y \in \mathbf{E}$ and $t \in [0,1]$. Furthermore, g is called strictly C-convex if the above inequality holds with $<_C$ when $x \neq y$ and $t \in (0,1)$.

Proposition 2.2.6 *Let $\mathbf{E}$ and $\mathbf{Y}$ be Euclidean spaces, C a pointed closed convex cone in $\mathbf{Y}$, and $g : \mathbf{E} \to \mathbf{Y}$ a C-convex function. Then the set $\Omega = \{x \in \mathbf{E} \mid g(x) \leq_C \mathbf{0}\}$ is convex.*

Proof. Let $x, y \in \Omega$ and $t \in [0,1]$. Then $-g(x), -g(y) \in C$ and the convexity of C implies $-tg(x) - (1-t)g(y) \in C$. Therefore $g(tx + (1-t)y) \leq_C tg(x) + (1-t)g(y) \leq_C \mathbf{0}$ and the transitivity of $\leq_C$ gives us $tx + (1-t)y \in \Omega$. ■

In §2.1 we stated that the cones of primary interest to us are cones that are self-dual. Recall that a cone H in $\mathbf{E}$ is called self-dual if $H = H^*$, where

$$H^* = \{y \in \mathbf{E} \mid \langle x, y \rangle \geq 0 \text{ for all } x \in H\}.$$

It turns out that self-dual cones are always pointed, closed, and convex.

Proposition 2.2.7 *Let H be self-dual cone in $\mathbf{E}$. Then H is a pointed closed convex cone.*

Proof. Let $y \in H$, and suppose $-y \in H$. Then we have $\langle x, y \rangle \geq 0$ and $\langle x, -y \rangle \geq 0$ for all $x \in H$. Thus $\langle x, y \rangle = 0$ for all $x \in H$, which implies that $\langle y, y \rangle = 0$. Therefore $y = \mathbf{0}$, which shows that $H \cap -H \subseteq \{\mathbf{0}\}$. Since every cone satisfies $\mathbf{0} \in H$, we see that H is pointed.

To show H is closed, it suffices to show that for every sequence $\{y_n\}$ in H that converges to some point $y \in \mathbf{E}$, we have $y \in H$. Let $x \in H$, then $\langle x, y_n \rangle \geq 0$ for all $n \in \mathbb{N}$, and since the inner product is continuous, we have $\langle x, y \rangle = \lim_{n \to \infty} \langle x, y_n \rangle \geq 0$. Since x was arbitrary, $\langle x, y \rangle \geq 0$ for all $x \in H$, and therefore $y \in H$.

Finally, to show that H is convex, we let $u, v \in H$, $t \in [0,1]$, and we notice that

$$\langle x, tu + (1-t)v \rangle = t \langle x, u \rangle + (1-t) \langle x, v \rangle \geq 0$$

for all $x \in H$, which implies that $tu + (1-t)v \in H$. ■

2.3 Lagrangian Duality and Optimality Conditions

We are now in a position to describe the duality theory for the convex program

$$\inf\{f(x) \mid g(x) \leq_H \mathbf{0}\} \tag{2.3}$$

where $f : \mathbf{E} \to \mathbb{R}$ is convex, $g : \mathbf{E} \to \mathbf{Y}$ is H-convex, $\mathbf{E}$ and $\mathbf{Y}$ are Euclidean spaces, and H is a self-dual cone in $\mathbf{Y}$. Recall that $\mathbb{R}^n$, $\mathbb{R}^{m \times n}$, and $\mathcal{S}^n$ are all Euclidean spaces, and that $\mathbb{R}^n_+$ and $\mathcal{S}^n_+$ are both self-dual cones. Therefore this general formulation allows us to apply the results obtained to the many problems of interest in this thesis, and the proofs of the results are practically identical to the case of semidefinite constraints, which we will be considering for our least squares problems. The majority of this section is drawn from Borwein and Lewis' book [6, §3.2, §4.3] where they present the duality theory of problem (2.3) for the case of $\mathbf{Y} = \mathbb{R}^m$ and $H = \mathbb{R}^m_+$. Furthermore, the following development is undertaken here because it is believed that it is not to be found elsewhere stated in these terms.

2.3.1 Lagrangian duality

We begin by introducing the central concept in this section, the Lagrangian function $L : \mathbf{E} \times \mathbf{Y} \to \mathbb{R}$, defined by

$$L(x;\lambda) := f(x) + \langle \lambda, g(x) \rangle .$$

The immediate power of the Lagrangian function becomes apparent when we notice that by equation (2.1), the self-duality of H implies that if $-g(x) \in H$, then $\langle \lambda, g(x) \rangle \leq 0$ for all $\lambda \in H$, and if $-g(x) \notin H$, we can pick $\lambda \in H$ to make $\langle \lambda, g(x) \rangle$ as large as we want. Thus,

$$\sup_{\lambda \in H} L(x;\lambda) = \begin{cases} f(x), & \text{if } x \text{ is feasible} \\ +\infty, & \text{otherwise,} \end{cases}$$

and we can state our optimization problem as

$$p := \inf_{x \in \mathbf{E}} \sup_{\lambda \in H} L(x;\lambda), \tag{2.4}$$

where $p \in [-\infty, +\infty]$ is called the *primal value* of the *primal problem* (2.3).

It is also useful to quickly note that this same trick can be applied to problems with equality constraints. For example, consider the problem

$$\inf\{f(x) \mid g(x) = \mathbf{0}\}$$

which will then have the Lagrangian interpretation

$$\sup_{\lambda \in \mathbf{Y}} L(x;\lambda) = \begin{cases} f(x), & \text{if } x \text{ is feasible} \\ +\infty, & \text{otherwise,} \end{cases}$$

where $L(x;\lambda) := f(x) + \langle \lambda, g(x) \rangle$. In a sense, this is due to the fact that $\mathbf{Y}$ is the dual of the cone $\{\mathbf{0}\}$. That is, $\{\mathbf{0}\}^* = \mathbf{Y}$.

Returning to problem (2.3), we define the Lagrangian *dual problem* as

$$d := \sup_{\lambda \in H} \inf_{x \in \mathbf{E}} L(x;\lambda), \tag{2.5}$$

where $d \in [-\infty, +\infty]$ is called the *dual value*, and notice the following fundamental result.

Theorem 2.3.1 (Weak Duality)
Defining the primal function $\Psi : \mathbf{E} \to (-\infty, +\infty]$ *by*

$$\Psi(x) := \sup_{\lambda \in H} L(x;\lambda)$$

and the dual function $\Phi : H \to [-\infty, +\infty)$ *by*

$$\Phi(\lambda) := \inf_{x \in \mathbf{E}} L(x;\lambda)$$

we have $\Psi(x) \geq \Phi(\lambda)$ *for all* $x \in \mathbf{E}$ *and* $\lambda \in H$. *Moreover,* $p \geq d$.

Proof. Let $\hat{x} \in \mathbf{E}$ and $\hat{\lambda} \in H$. Then

$$\Psi(\hat{x}) = \sup_{\lambda \in H} L(\hat{x};\lambda) \geq L(\hat{x};\hat{\lambda}) \geq \inf_{x \in \mathbf{E}} L(x;\hat{\lambda}) = \Phi(\hat{\lambda}).$$

Since $\hat{x} \in \mathbf{E}$ and $\hat{\lambda} \in H$ are arbitrary and independent, $p = \inf_{x \in \mathbf{E}} \Psi(x) \geq \sup_{\lambda \in H} \Phi(\lambda) = d$. ■

2.3.2 Sufficient conditions for optimality

The usefulness of the Weak Duality Theorem is that if we find $\bar{x} \in \mathbf{E}$ and $\bar{\lambda} \in H$ such that $\Psi(\bar{x}) = \Phi(\bar{\lambda}) \in (-\infty, +\infty)$, then we know that $\bar{x}$ is feasible for (2.3) and attains the minimum. Not only is $\bar{x}$ optimal for the primal problem, but $\bar{\lambda}$ is optimal for the dual problem as well. Furthermore, we notice that

$$\left.\begin{array}{rcl} 0 = \Psi(\bar{x}) - \Phi(\bar{\lambda}) & = & f(\bar{x}) - \inf_{x \in \mathbf{E}} [\, f(x) + \langle \bar{\lambda}, g(x) \rangle \,] \\ & \geq & f(\bar{x}) - f(\bar{x}) - \langle \bar{\lambda}, g(\bar{x}) \rangle \\ & = & \langle \bar{\lambda}, -g(\bar{x}) \rangle \;\geq\; 0, \end{array}\right\} \tag{2.6}$$

where the last inequality comes from the self-duality of H. Therefore, equality holds throughout, and we have that $\bar{x}$ minimizes the function $L(\,\cdot\,; \bar{\lambda})$ over $\mathbf{E}$ and that $\langle \bar{\lambda}, -g(\bar{x}) \rangle = 0$ (hence $\langle \bar{\lambda}, g(\bar{x}) \rangle = 0$). This motivates the following definition and theorem.

Definition 2.3.2 *Let $\bar{x} \in \mathbf{E}$ be feasible for the convex program (2.3). We call $\bar{\lambda} \in H$ a Lagrange multiplier for $\bar{x}$ if $\bar{x}$ minimizes the function $L(\,\cdot\,; \bar{\lambda})$ over $\mathbf{E}$ and $\langle \bar{\lambda}, g(\bar{x}) \rangle = 0$.*

Theorem 2.3.3 (Lagrangian sufficient conditions)
If $\bar{x} \in \mathbf{E}$ is feasible for the convex program (2.3) and has a Lagrange multiplier $\bar{\lambda}$, then $\bar{x}$ is optimal.

Proof. Assuming $\bar{\lambda}$ is a Lagrange multiplier for $\bar{x}$, equation (2.6) shows that $\Psi(\bar{x}) = \Phi(\bar{\lambda})$, and the optimality of $\bar{x}$ follows. ■

2.3.3 The KKT conditions

Up until this point, we have not had to use the convexity of the problem (2.3) in the above results. However, using the facts that f is convex, g is H-convex, and H is a self-dual cone, we have the following result.

Lemma 2.3.4 (Convexity of the Langrangian) *Let $\mathbf{E}$ and $\mathbf{Y}$ be Euclidean spaces, $H \subseteq \mathbf{Y}$ a self-dual cone, $f : \mathbf{E} \to \mathbb{R}$ a convex function, and $g : \mathbf{E} \to \mathbf{Y}$ an H-convex function. Defining $L : \mathbf{E} \times \mathbf{Y} \to \mathbb{R}$ by $L(x; \lambda) = f(x) + \langle \lambda, g(x) \rangle$, we have the following results:*

(a) If $\mu \leq_H \nu$ and $\lambda \in H$, then $\langle \lambda, \mu \rangle \leq \langle \lambda, \nu \rangle$.

(b) The function $L(\,\cdot\,; \lambda) : \mathbf{E} \to \mathbb{R}$ is convex for all $\lambda \in H$.

Proof. Part (a) follows from the self-duality of H and the following implications:

$$\begin{array}{rcl} \mu \leq_H \nu & \Rightarrow & \nu - \mu \in H \\ & \Rightarrow & \langle \lambda, \nu - \mu \rangle \geq 0 \\ & \Rightarrow & \langle \lambda, \mu \rangle \leq \langle \lambda, \nu \rangle \,. \end{array}$$

For part (b), we let $x, y \in \mathbf{E}$ and $t \in [0,1]$. Then

$$\begin{aligned} L(tx+(1-t)y;\lambda) &= f(tx+(1-t)y) + \langle \lambda, g(tx+(1-t)y)\rangle \\ &\leq tf(x)+(1-t)f(y) + \langle \lambda, tg(x)+(1-t)g(y)\rangle \\ &= tL(x;\lambda)+(1-t)L(y;\lambda), \end{aligned}$$

where the inequality follows from the convexity of f, the H-convexity of g, and part (a) of this lemma. ■

Therefore, if we are dealing with differentiable functions, and in particular, if the function $L(x;\lambda)$ is differentiable with respect to x, then any critical point of $L(\,\cdot\,;\lambda)$ is in fact a global minimizer of $L(\,\cdot\,;\lambda)$ over $\mathbf{E}$. Thus, finding a solution to the following system of equations will give us a feasible point $\bar{x}$ with Lagrange multiplier $\bar{\lambda}$, and hence an optimal solution to the convex program (2.3).

$$\left.\begin{array}{rcl} \nabla_x L(\bar{x};\bar{\lambda}) & = & \mathbf{0} \\ -g(\bar{x}) & \geq_H & \mathbf{0} \\ \bar{\lambda} & \geq_H & \mathbf{0} \\ \langle \bar{\lambda}, g(\bar{x})\rangle & = & 0 \end{array}\right\} \tag{2.7}$$

This system of equations is often known in other settings as the *Karush-Kuhn-Tucker (KKT) conditions* (for example, [21, p. 328]).

In the KKT conditions (2.7), ∇_x denotes the derivative, or *gradient*, with respect to x, where the gradient of a function $h : \mathbf{E} \to \mathbb{R}$ at $\bar{x} \in \mathbf{E}$, denoted $\nabla h(\bar{x})$, is defined as the element $a \in \mathbf{E}$ (if it exists) such that $h'(\bar{x};d) = \langle a, d\rangle$ for all $d \in \mathbf{E}$. Here $h'(\bar{x};d)$ is the *directional derivative* of h at $\bar{x}$ in the direction d, and is defined as

$$h'(\bar{x};d) := \lim_{t\to 0^+} \frac{h(\bar{x}+td)-h(\bar{x})}{t},$$

if this limit exists. Thus, h has a (Gâteaux) derivative $\nabla h(\bar{x})$ at $\bar{x}$ if $h'(\bar{x};d)$ exists for all $d \in \mathbf{E}$ and if $h'(\bar{x};\,\cdot\,)$ is a linear function on $\mathbf{E}$. If $\nabla h(\bar{x})$ exists for all $\bar{x} \in \mathbf{E}$, then we say that h is *differentiable*. If $\nabla h : \mathbf{E} \to \mathbf{E}$ is a continuous function, then we say that h is *continuously differentiable*.

Example 2.3.5 *Consider the sum of squares function $f(X) = \frac{1}{2}\|AX-B\|_F^2$ for $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{m\times p}$, and $X \in \mathbb{R}^{n\times p}$. Letting $D \in \mathbb{R}^{n\times p}$ be some direction, we have*

$$\begin{aligned} f'(X;D) &= \lim_{t\to 0^+} \frac{f(X+tD)-f(X)}{t} \\ &= \lim_{t\to 0^+} \frac{\frac{1}{2}\langle A(X+tD)-B, A(X+tD)-B\rangle - \frac{1}{2}\langle AX-B, AX-B\rangle}{t} \\ &= \lim_{t\to 0^+} \frac{t\langle AX-B, AD\rangle + \frac{1}{2}t^2\langle AD, AD\rangle}{t} \\ &= \left\langle A^T(AX-B), D\right\rangle. \end{aligned}$$

Therefore, $\nabla f(X) = A^T(AX-B)$.

If we consider f defined over the Euclidean space $\mathcal{S}^n$ with $A, B \in \mathbb{R}^{m\times n}$, then we require $\nabla f(X) \in \mathcal{S}^n$. Letting $D \in \mathcal{S}^n$ and $Z := A^T(AX - B)$, we have $f'(X; D) = \langle Z, D\rangle$ and

$$\begin{aligned}\left\langle \tfrac{1}{2}(Z + Z^T), D\right\rangle &= \tfrac{1}{2}\langle Z, D\rangle + \tfrac{1}{2}\langle Z^T, D\rangle \\ &= \tfrac{1}{2}\langle Z, D\rangle + \tfrac{1}{2}\langle Z, D\rangle \\ &= \langle Z, D\rangle,\end{aligned}$$

where the second equality follows from the symmetry of D. Since $\frac{1}{2}(Z + Z^T) \in \mathcal{S}^n$, we find that $\nabla f(X) = \frac{1}{2}(Z + Z^T)$. □

2.3.4 Necessary conditions for optimality

Now we know that if we are able to solve the KKT system (2.7), then we have solved our optimization problem, but how do we know that the KKT system is in fact solvable for our problem? That is, how do we know that if an optimal solution exists, it must have a Lagrange multiplier? To answer this question, we state the *Slater constraint qualification* for problem (2.3):

$$\text{There exists } \hat{x} \in \mathbf{E} \text{ such that } g(\hat{x}) <_H \mathbf{0}. \tag{2.8}$$

It turns out that the Slater condition is enough to guarantee the existence of a Lagrange multiplier for every optimal solution.

Theorem 2.3.6 (Lagrangian necessary conditions) *Suppose that $\bar{x} \in \mathbf{E}$ is optimal for the convex program (2.3) and that the Slater condition (2.8) holds. Then there is a Lagrange multiplier for $\bar{x}$.*

Before presenting a proof to this theorem, we need to develop some further concepts. The first of these is the *value function* $v : \mathbf{Y} \to [-\infty, +\infty]$ defined by

$$v(y) := \inf\{f(x) \mid g(x) \leq_H y\}, \tag{2.9}$$

so that $v(\mathbf{0})$ is the optimal value of the convex program (2.3). We will use the value function in the proof of Theorem 2.3.6 by showing that any subgradient of v at $\mathbf{0}$ is a Lagrange multiplier for any optimal $\bar{x}$, where subgradient is defined as follows:

Definition 2.3.7 *Let $h : \mathbf{Y} \to (-\infty, +\infty]$. We say $\phi \in \mathbf{Y}$ is a subgradient of h at $\bar{y}$ if*

$$\langle \phi, y - \bar{y}\rangle \leq h(y) - h(\bar{y}), \text{ for all } y \in \mathbf{Y}$$

We denote the set of subgradients of h at $\bar{y}$ as $\partial h(\bar{y})$.

For a convex function $h : \mathbf{Y} \to (-\infty, +\infty]$, where convex here means h is convex on its *domain*,

$$\text{dom}(h) := \{y \in \mathbf{Y} \mid h(y) < +\infty\},$$

the connection between the gradient and subgradients at a point can be seen in the following theorem.

Theorem 2.3.8 (Borwein & Lewis [6, p. 36]) *If the function $h : \mathbf{Y} \to (-\infty, +\infty]$ is convex then any point $\bar{y} \in \text{core}(\text{dom}\, h)$ and any direction $d \in \mathbf{E}$ satisfy*

$$h'(\bar{y}; d) = \max\{\langle \phi, d \rangle \mid \phi \in \partial h(\bar{y})\}. \tag{2.10}$$

In particular, the subdifferential $\partial h(\bar{y})$ is nonempty.

Here, the *core* of a set $C \subseteq \mathbf{Y}$ is the set of points $y \in C$ such that for all directions $d \in \mathbf{Y}$ we have $y + td \in C$ for all $t > 0$ small enough.

For extended real valued functions $h : \mathbf{Y} \to [-\infty, +\infty]$ we say h is a convex function if

$$\text{epi}(h) = \{(y, r) \in \mathbf{Y} \times \mathbb{R} \mid h(y) \leq r\}$$

is a convex set. Using this extended definition of convexity, we can show that the value function is actually convex.

Proposition 2.3.9 *The value function defined by (2.9) is convex.*

Proof. Let $(y_1, r_1), (y_2, r_2) \in \text{epi}(v)$, $t \in [0, 1]$, and set $(y, r) := t(y_1, r_1) + (1 - t)(y_2, r_2)$. Then $v(y_1) \leq r_1$, $v(y_2) \leq r_2$, and we want to show that $v(y) \leq r$.

First of all, we notice that

$$\begin{aligned} v(y_i) \leq r_i &\Rightarrow \inf\{f(x) \mid g(x) \leq_H y_i\} \leq r_i \\ &\Rightarrow \exists\, x_i \in \mathbf{E} \text{ such that } g(x_i) \leq_H y_i \end{aligned} \tag{2.11}$$

for $i = 1, 2$. Now we let $x := tx_1 + (1 - t)x_2$, and the H-convexity of g implies that

$$\begin{aligned} g(x) &\leq_H tg(x_1) + (1 - t)g(x_2) \\ &\leq_H ty_1 + (1 - t)y_2 = y, \end{aligned}$$

so that $v(y) \leq f(x)$. This fact and the convexity of f implies that

$$v(y) \leq tf(x_1) + (1 - t)f(x_2).$$

Taking the infimum over all x_1 and x_2 satisfying 2.11 gives

$$\begin{aligned} v(y) &\leq tv(y_1) + (1 - t)v(y_2) \\ &\leq tr_1 + (1 - t)r_2 = r. \end{aligned}$$

Therefore, $\text{epi}(v)$ is a convex set, and thus v is a convex function. ■

As we will see in the proof of Theorem 2.3.6, the convexity of the value function v allows us to use the following result to show that the Slater condition (2.8) implies that v never takes on the value $-\infty$.

Lemma 2.3.10 (Borwein & Lewis [6, p. 44]) *If the function $h : \mathbf{Y} \to [-\infty, +\infty]$ is convex and some point $\hat{y} \in \text{core}(\text{dom}\, h)$ satisfies $h(\hat{y}) > -\infty$, then h never takes the value $-\infty$.*

We are now in a position to prove the Lagrangian necessary conditions.

Proof of Theorem 2.3.6. Recall that we are assuming that $\bar{x} \in \mathbf{E}$ is optimal for the convex program (2.3) and that the Slater condition (2.8) holds: $\exists\, \hat{x} \in \mathbf{E}$ such that $g(\hat{x}) <_H \mathbf{0}$.

Since $\bar{x}$ is optimal, we have $v(\mathbf{0}) = f(\bar{x}) > -\infty$. By showing that $\mathbf{0} \in \operatorname{core}(\operatorname{dom} v)$, we can use Lemma 2.3.10 to show that $v : \mathbf{Y} \to (-\infty, +\infty]$. Thus, we need to show that for all directions $d \in \mathbf{E}$ we have $\mathbf{0} + td \in \operatorname{dom}(v)$ for all $t > 0$ small enough. Letting $d \in \mathbf{E}$ we have

$$\begin{aligned} td \in \operatorname{dom}(v) &\Leftrightarrow v(td) = \inf\{f(x) \mid g(x) \leq_H td\} < +\infty \\ &\Leftrightarrow C(t) := \{x \in \mathbf{E} \mid g(x) \leq_H td\} \neq \emptyset. \end{aligned}$$

By showing that $\exists\, \varepsilon > 0$ such that $\forall\, t \in (0, \varepsilon)$ we have $\hat{x} \in C(t)$, we shall have proved our claim that $\mathbf{0} \in \operatorname{core}(\operatorname{dom} v)$.

First of all, we will show that $\exists\, k \in \mathbb{N}$ such that $\hat{x} \in C(\frac{1}{k})$. Suppose, on the contrary, that $\hat{x} \notin C(\frac{1}{k})$ for all $k \in \mathbb{N}$. That is, the sequence defined by $a_k = \frac{1}{k}d - g(\hat{x})$ lies entirely in $\mathbf{E} \setminus H$, which implies that $-g(\hat{x}) = \lim_{k\to\infty} a_k \in \operatorname{cl}(\mathbf{E} \setminus H) = \mathbf{E} \setminus \operatorname{int}(H)$, with the last equality following from the fact that H is closed. This gives us our contradiction, since $-g(\hat{x}) \in \operatorname{int}(H)$.

Letting $\varepsilon := \frac{1}{k}$, where $\hat{x} \in C(\frac{1}{k})$, we will now show that $\hat{x} \in C(t)$ for all $t \in (0, \varepsilon)$. This can be achieved by showing that the set

$$T := \{t \geq 0 \mid \hat{x} \in C(t)\}$$

is convex, and using the fact that $0, \varepsilon \in T$.

Let $t_1, t_2 \in T$ and $\alpha \in [0, 1]$. We want to show that $t := \alpha t_1 + (1 - \alpha)t_2 \in T$. Since

$$\begin{aligned} t_i \in T &\Leftrightarrow \hat{x} \in C(t_i) \\ &\Leftrightarrow g(\hat{x}) \leq_H t_i d, \end{aligned}$$

for $i = 1, 2$, we have

$$\begin{aligned} td - g(\hat{x}) &= \alpha t_1 d + (1 - \alpha)t_2 d - g(\hat{x}) \\ &= \alpha(t_1 d - g(\hat{x})) + (1 - \alpha)(t_2 d - g(\hat{x})) \\ &\in H. \end{aligned}$$

Therefore, $g(\hat{x}) \leq_H td$, which implies $t \in T$. Hence, T is convex, finally proving that $\mathbf{0} \in \operatorname{core}(\operatorname{dom} v)$.

Thus, by Lemma 2.3.10, we have $v : \mathbf{Y} \to (-\infty, +\infty]$, so we can apply Theorem 2.3.8 to get the fact that $\partial v(\mathbf{0}) \neq \emptyset$, which implies the existence of a subgradient $-\bar{\lambda} \in \mathbf{Y}$ of v at $\mathbf{0}$. It remains to show that $\bar{\lambda}$ is a Lagrange multiplier for $\bar{x}$. That is, $\bar{\lambda} \in H$, $\bar{x}$ minimizes $L(\cdot\,; \bar{\lambda})$, and $\langle \bar{\lambda}, g(\bar{x}) \rangle = 0$.

We begin by showing that $\bar{\lambda} \in H$, which, by the self-duality of H, is equivalent to showing $\langle \bar{\lambda}, y \rangle \geq 0$ for all $y \in H$. Letting $y \in H$, we have that $g(x) \leq_H \mathbf{0}$ implies $g(x) \leq_H y$. Thus,

$$\begin{aligned} v(y) &= \inf\{f(x) \mid g(x) \leq_H y\} \\ &\leq \inf\{f(x) \mid g(x) \leq_H \mathbf{0}\} = v(\mathbf{0}). \end{aligned}$$

Since $-\bar{\lambda} \in \partial v(\mathbf{0})$, Definition 2.3.7 shows $\langle -\bar{\lambda}, y - \mathbf{0} \rangle \leq v(y) - v(\mathbf{0})$, so

$$\begin{aligned} v(\mathbf{0}) &\leq v(y) + \langle \bar{\lambda}, y \rangle \\ &\leq v(\mathbf{0}) + \langle \bar{\lambda}, y \rangle, \end{aligned}$$

giving us $\langle \bar{\lambda}, y \rangle \geq 0$ for all $y \in H$.

To show that $\bar{x}$ minimizes $L(\,\cdot\,; \bar{\lambda})$ and $\langle \bar{\lambda}, g(\bar{x}) \rangle = 0$, we again use the definition of subgradients, this time to get that $\langle -\bar{\lambda}, g(x) - \mathbf{0} \rangle \leq v(g(x)) - v(\mathbf{0})$, so that

$$\begin{aligned} v(\mathbf{0}) &\leq v(g(x)) + \langle \bar{\lambda}, g(x) \rangle \\ &= \inf\{f(x) \mid x \in \mathbf{E}\} + \langle \bar{\lambda}, g(x) \rangle \\ &\leq f(x) + \langle \bar{\lambda}, g(x) \rangle \end{aligned}$$

for all $x \in \mathbf{E}$. Since $\bar{x}$ is optimal, we also have $v(\mathbf{0}) = f(\bar{x})$, which implies that $f(\bar{x}) \leq f(x) + \langle \bar{\lambda}, g(x) \rangle$ for all $x \in \mathbf{E}$. Using this fact, we find that $f(\bar{x}) \leq f(\bar{x}) + \langle \bar{\lambda}, g(\bar{x}) \rangle$, so we get $\langle \bar{\lambda}, g(\bar{x}) \rangle \geq 0$. On the other hand, $\langle \bar{\lambda}, g(\bar{x}) \rangle \leq 0$ since $-g(\bar{x}), \bar{\lambda} \in H$. Therefore, $\langle \bar{\lambda}, g(\bar{x}) \rangle = 0$, which implies that

$$\begin{aligned} L(\bar{x}; \bar{\lambda}) &= f(\bar{x}) + \langle \bar{\lambda}, g(\bar{x}) \rangle \\ &= f(\bar{x}) \\ &\leq f(x) + \langle \bar{\lambda}, g(x) \rangle \;=\; L(x; \bar{\lambda}). \end{aligned}$$

Therefore, $\bar{\lambda}$ is a Lagrange multiplier for $\bar{x}$. ■

2.3.5 Summary

We will now summarize the preceding results and introduce some new terminology which will be used to guide our discussion. For our purposes here, we will be assuming that the Lagrangian function is differentiable with respect to x.

We first introduce the *slack vector*, $s \in \mathbf{Y}$, which we use to "add slack" to the inequality in problem (2.3), giving us the equivalent problem

$$\begin{array}{ll} \inf & f(x) \\ \text{s.t.} & g(x) + s = \mathbf{0}, \\ & s \geq_H \mathbf{0}, \end{array} \tag{2.12}$$

with associated dual problem

$$\begin{array}{ll} \sup & L(x; \lambda) \\ \text{s.t.} & \nabla_x L(x; \lambda) = \mathbf{0}, \\ & \lambda \geq_H \mathbf{0}. \end{array} \tag{2.13}$$

With this slack vector, the KKT conditions for optimality can now be stated as

$$\left.\begin{aligned} \nabla_x L(\bar{x}; \bar{\lambda}) &= \mathbf{0}, \quad \bar{\lambda} \geq_H \mathbf{0}, \\ g(\bar{x}) + \bar{s} &= \mathbf{0}, \quad \bar{s} \geq_H \mathbf{0}, \\ \langle \bar{\lambda}, \bar{s} \rangle &= 0, \end{aligned}\right\} \tag{2.14}$$

where the first two lines can be understood as dual feasibility and primal feasibility, respectively, and the last condition is often coined as *complementary slackness.*

We say that a point $(x, s, \lambda) \in \mathbf{E} \times \mathbf{Y} \times \mathbf{Y}$ is *primal-dual feasible* if $g(x)+s = \mathbf{0}$, $\nabla_x L(x; \lambda) = \mathbf{0}$, and $s,\ \lambda \succeq_H \mathbf{0}$. Given a primal-dual feasible point (x, s, λ), we know from Theorem 2.3.1 (Weak Duality) that the value of the objective function of the primal is always greater or equal to that of the dual, with equality implying that (x, s) is optimal for the primal problem and λ is optimal for the dual problem. Letting $\mu(x, s, \lambda)$ represent this *duality gap*, we find that

$$\begin{aligned} \mu(x, s, \lambda) &:= f(x) - L(x; \lambda) \\ &= f(x) - f(x) - \langle \lambda, g(x) \rangle \\ &= \langle \lambda, -g(x) \rangle \ = \ \langle \lambda, s \rangle \ \geq \ 0 \end{aligned} \tag{2.15}$$

with $\mu(x, s, \lambda) = 0$ implying that (x, s, λ) is optimal for problems (2.12) and (2.13). Furthermore, if (x, s) is optimal and the Slater condition holds for problem (2.12), then there exists $\lambda \in \mathbf{Y}$ such that (x, s, λ) is primal-dual feasible and $\mu(x, s, \lambda) = 0$.

Finally, we summarize these results in the following theorem.

Theorem 2.3.11 *Suppose the Lagrangian function for problem (2.12) is differentiable with respect to x and that the Slater condition (2.8) holds. Then $\bar{x} \in \mathbf{E}$ is optimal if and only if there exist points $\bar{s},\ \bar{\lambda} \in \mathbf{Y}$ such that $(\bar{x}, \bar{s}, \bar{\lambda})$ is primal-dual feasible and $\mu(\bar{x}, \bar{s}, \bar{\lambda}) = 0$, which is equivalent to $(\bar{x}, \bar{s}, \bar{\lambda})$ satisfying the KKT conditions (2.14).*

2.4 Application to Semidefinite Constrained Least Squares

We can now apply the results of the previous section to the three problems of interest in order to state the conditions necessary and sufficient for optimality.

2.4.1 The symmetric semidefinite least squares problem (SDLS)

As we mentioned in Chapter 1, we would like to find a solution to the matrix equation $AX = B$, where $A, B \in \mathbb{R}^{m \times n}$ and $X \in \mathcal{S}^n$, in the least squares sense, but where we require that X be positive semidefinite. The SDLS problem can be stated formally as

$$\begin{array}{ll} \inf & \frac{1}{2}\|AX - B\|_F^2 \\ \text{s.t.} & X \succeq \mathbf{0} \end{array} \tag{2.16}$$

which is a special case of problem (2.3) with $\mathbf{E} = \mathbf{Y} = \mathcal{S}^n$, $H = \mathcal{S}^n_+$, $f(X) = \frac{1}{2}\|AX - B\|_F^2$, and $g(X) = -X$. We can use Proposition 2.2.2 to show that f is a convex function over $\mathcal{S}^n$, and we note that since g is clearly a linear function, it is necessarily H-convex.

For this problem the Lagrangian function $L : \mathcal{S}^n \times \mathcal{S}^n \to \mathbb{R}$ is defined by

$$\begin{aligned} L(X; \Lambda) &= f(X) + \langle \Lambda, g(X) \rangle \\ &= \tfrac{1}{2}\|AX - B\|_F^2 - \langle \Lambda, X \rangle , \end{aligned}$$

with X derivative

$$\nabla_X L(X;\Lambda) = \tfrac{1}{2}(Z + Z^T) - \Lambda, \quad \text{where } Z = A^T(AX - B).$$

Since the Slater condition (2.8) clearly holds for this problem (for example, $\hat{X} = I$), we have that $\bar{X} \in \mathcal{S}^n$ is optimal for (2.16) if and only if there exists $\bar{\Lambda} \in \mathcal{S}^n$ such that the following KKT conditions are satisfied:

$$\begin{aligned} A^T(A\bar{X} - B) &= \bar{Z}, \\ \tfrac{1}{2}(\bar{Z} + \bar{Z}^T) &= \bar{\Lambda}, \\ \langle \bar{\Lambda}, \bar{X} \rangle &= 0, \\ \bar{X},\, \bar{\Lambda} &\succeq \mathbf{0}. \end{aligned}$$

We now state the following important fact, which is found as an exercise in [6, p. 108].

Proposition 2.4.1 (Semidefinite complementarity) *If $X, Y \succeq \mathbf{0}$, then the following are equivalent:*

(i) $\langle X, Y \rangle = 0$,

(ii) $XY = \mathbf{0}$,

(iii) $\frac{1}{2}(XY + YX) = \mathbf{0}$.

Proof. *(i)* $\Rightarrow$ *(ii)*: To show that $\langle X, Y \rangle = \mathrm{tr}(XY) = 0$ implies $XY = \mathbf{0}$, we follow Todd's advice [27, p. 523] by considering the eigenvalue decomposition of X, $X = U\Lambda U^T$, where U is an orthonormal matrix ($UU^T = I$) and Λ is a diagonal matrix with the eigenvalues of X,

$$\lambda_1 \geq \ldots \geq \lambda_r > \lambda_{r+1} = \ldots = \lambda_n = 0,$$

along its diagonal.

We first notice that

$$\begin{aligned} 0 = \mathrm{tr}(XY) &= \mathrm{tr}(U\Lambda U^T Y) \\ &= \mathrm{tr}(\Lambda U^T Y U) \\ &= \langle \Lambda, U^T Y U \rangle \\ &= \sum_{i=1}^{n} \lambda_i (U^T Y U)_{ii}. \end{aligned}$$

Since Y is positive semidefinite, so is U^TYU, and thus $(U^TYU)_{ii} \geq 0$ for $i = 1, \ldots, n$. This implies that $\lambda_i(U^TYU)_{ii} = 0$ for $i = 1, \ldots, n$, which tells us that $(U^TYU)_{ii} = 0$ for $i = 1, \ldots, r$. Since U^TYU is positive semidefinite with zeros along the diagonal in the first r positions, we must have

$$U^TYU = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \tilde{Y} \end{bmatrix}$$

for some $\tilde{Y} \in \mathcal{S}_+^{n-r}$. Letting $\tilde{\Lambda} = \mathrm{Diag}(\lambda_1, \ldots, \lambda_r)$, we also have

$$\Lambda = \begin{bmatrix} \tilde{\Lambda} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix},$$

and so $\Lambda U^T Y U = \mathbf{0}$, which in turn implies that $XY = U(\Lambda U^T Y U)U^T = U \cdot \mathbf{0} \cdot U^T = \mathbf{0}$.

(ii) $\Rightarrow$ *(iii)*: Clearly, if $XY = \mathbf{0}$, then $\frac{1}{2}(XY + YX) = \frac{1}{2}(XY + (XY)^T) = \mathbf{0}$.

(iii) $\Rightarrow$ *(i)*: This follows from the fact that $\mathrm{tr}(\frac{1}{2}(XY + YX)) = \mathrm{tr}(XY)$. ■

Therefore, the KKT conditions for problem (2.16) are equivalent to the following nonlinear system of equations.

$$\left.\begin{array}{rcl} A^T(A\bar{X} - B) & = & \bar{Z} \\ \frac{1}{2}(\bar{Z} + \bar{Z}^T) & = & \bar{\Lambda} \\ \bar{\Lambda}\bar{X} & = & \mathbf{0} \\ \bar{X},\, \bar{\Lambda} & \succeq & \mathbf{0} \end{array}\right\} \tag{2.17}$$

It should be noted that although such a Lagrange multiplier type characterization of optimality was stated by Woodgate in [33], it was not stated using the Lagrange multiplier matrix Λ. For comparative purposes, we provide this characterization of Woodgate:

$$\bar{X} \in \bar{\mathcal{S}}_+^n \quad \text{and} \quad L(\bar{X}) \in \mathcal{S}_+^n,$$

where

$$\begin{array}{rcl} L(X) & := & A^T A X + X A^T A - A^T B - B^T A, \\ \bar{\mathcal{S}}_+^n & := & \{X \in \mathcal{S}_+^n \mid \mathrm{tr}(X L(X)) = 0\}. \end{array}$$

We can clearly see why Woodgate's characterization of optimality is equivalent to (2.17), however, as we will see, using (2.17) will be highly useful when discussing the interior-point algorithms of Chapter 3.

Now for a note on whether or not the SDLS problem actually has an optimal solution. We are concerned about the existence of an optimal solution because the usefulness of the above KKT characterization of optimality only applies if such a solution exists, and, as we see from the following example, this is not always the case.

Example 2.4.2 (Woodgate's counterexample, [4]) *Consider the SDLS problem with*

$$A = \begin{bmatrix} 1 & 0 \end{bmatrix},\ B = \begin{bmatrix} -1 & 1 \end{bmatrix},\ \text{and}\ X = \begin{bmatrix} a & b \\ b & c \end{bmatrix}.$$

Then $f(X) = \frac{1}{2}\|AX - B\|_F^2 = \frac{1}{2}[(a+1)^2 + (b-1)^2]$ *and* $X \succeq \mathbf{0}$ *if and only if* $a \geq 0$ *and* $ac - b^2 \geq 0$. *This implies that* $f(X) \geq \frac{1}{2}(a+1)^2 \geq \frac{1}{2}$. *Furthermore, if*

$$X_n = \begin{bmatrix} \frac{1}{n} & 1 \\ 1 & n \end{bmatrix},$$

then $X_n \succeq \mathbf{0}$ *for all* $n \in \mathbb{N}$ *and* $\lim_{n\to\infty} f(X_n) = \lim_{n\to\infty} \frac{1}{2}(\frac{1}{n} + 1)^2 = \frac{1}{2}$. *Therefore, the optimal value is* $p = \frac{1}{2}$, *but as we will now show, this value is not attained.*

Suppose $X \succeq \mathbf{0}$ *and* $f(X) = \frac{1}{2}[(a+1)^2 + (b-1)^2] = \frac{1}{2}$. *Since* $a \geq 0$, *it must be the case that* $a = 0$ *and* $b = 1$, *which implies that* $ac - b^2 = -1 \not\geq 0$, *contradicting the assumption that* X *is positive semidefinite. Therefore, no* $X \in \mathcal{S}^2_+$ *satisfies* $f(X) = \frac{1}{2}$. □

Originally, in [3, Theorem 3.1], Allwright claimed that the minimum in the SDLS problem always exists. However, after the discovery by Woodgate of the above counterexample, they jointly produced an erratum and addendum paper [4] where they point out the error in the proof of that theorem, and correct the statement to say that the minimum in the SDLS problem exists when the matrix A has full column rank. Using the following theorem of Weierstrass, we will present an alternate proof of this result.

Theorem 2.4.3 (Weierstrass, [6, p. 4])
Suppose that the set $D \subseteq \mathbf{E}$ *is nonempty and closed, and that all the sub-level sets of the continuous function* $f : D \to \mathbb{R}$ *are bounded. Then* f *has a global minimizer.*

Here, the *sub-level sets* of a function $f : D \to \mathbb{R}$ are defined to be the family of sets

$$\{x \in D \mid f(x) \leq \alpha\} \tag{2.18}$$

for $\alpha \in \mathbb{R}$.

Lemma 2.4.4 *Let* $f : \mathbb{R}^{n\times p} \to \mathbb{R}$ *be the sum of squares function* $f(X) = \frac{1}{2}\|AX - B\|_F^2$, *with* $A \in \mathbb{R}^{m\times n}$ *and* $B \in \mathbb{R}^{m\times p}$. *Then* f *has bounded sub-level sets if and only if* A *has full column rank.*

Proof. Since f is always nonnegative, we need only consider sub-level sets

$$C(\alpha) := \{X \in \mathbb{R}^{n\times p} \mid f(X) \leq \alpha\}$$

for $\alpha \geq 0$.

Let us first show that if A does not have full column rank, then some of the sub-level sets of f are not bounded. For example, if $\alpha \geq \frac{1}{2}\|B\|_F^2$, we will show that $C(\alpha)$ is unbounded. Since $\operatorname{rank}(A) < n$, there exists a nonzero $\hat{X} \in \mathbb{R}^{n\times p}$ such that $A\hat{X} = \mathbf{0}$. Furthermore, letting $t \geq 0$, we also have $A(t\hat{X}) = \mathbf{0}$, which implies that $f(t\hat{X}) = \frac{1}{2}\|B\|_F^2 \leq \alpha$, and so $t\hat{X} \in C(\alpha)$ for all $t \geq 0$. However, since $\|\hat{X}\|_F > 0$, we find that $\lim_{t\to\infty} \|t\hat{X}\|_F = \lim_{t\to\infty} t\|\hat{X}\|_F = +\infty$. Therefore, $C(\alpha)$ is unbounded.

Now suppose that A has full column rank. Let $\alpha \geq 0$ and $X \in C(\alpha)$. Then $\frac{1}{2}\|AX - B\|_F^2 \leq \alpha$, which implies that $\|AX - B\|_F \leq \sqrt{2\alpha}$. This in turn implies that

$$\begin{aligned} \|AX\|_F &= \|AX - B + B\|_F \\ &\leq \|AX - B\|_F + \|B\|_F \\ &\leq \sqrt{2\alpha} + \|B\|_F, \end{aligned}$$

and so $\mathrm{tr}(X^T A^T A X) = \|AX\|_F^2 \leq \beta := (\sqrt{2\alpha} + \|B\|_F)^2$. Since A has full column rank, the $n \times n$ symmetric matrix $A^T A$ is positive definite and has eigenvalue decomposition $A^T A = Q^T \Lambda Q$, where Q is orthonormal and Λ is diagonal with the eigenvalues of $A^T A$,

$$\lambda_1 \geq \cdots \geq \lambda_n > 0,$$

along its diagonal. Letting $Y = QX$ (then $X = Q^T Y$ and $\|Y\|_F = \|X\|_F$), we have

$$\begin{aligned} \beta \geq \mathrm{tr}(X^T A^T A X) &= \mathrm{tr}(Y^T \Lambda Y) \\ &= \mathrm{tr}(\Lambda Y Y^T) \\ &= \sum_{i=1}^{n} \lambda_i (Y^T Y)_{ii} \\ &\geq \sum_{i=1}^{n} \lambda_n (Y^T Y)_{ii} \\ &= \lambda_n \mathrm{tr}(Y^T Y) = \lambda_n \|X\|_F^2, \end{aligned}$$

which implies that $\|X\|_F \leq M := \sqrt{\beta/\lambda_n}$, and so $C(\alpha)$ is bounded. ■

Therefore, if A has full column rank, then $f(X) = \frac{1}{2}\|AX - B\|_F^2$ must also have bounded sub-level sets over any subset of $\mathbb{R}^{n \times p}$. Since $\mathcal{S}_+^n$ is nonempty and closed, and f is a strictly convex and continuous function, Theorem 2.4.3 gives us the desired result.

Theorem 2.4.5 *If A has full column rank, then the SDLS problem has a unique optimal solution.*

2.4.2 The nonsymmetric semidefinite least squares problem (NS-SDLS)

It is sometimes desirable to find the least squares solution to the matrix equation $AX = B$, where $A, B \in \mathbb{R}^{m \times n}$, $X \in \mathbb{R}^{n \times n}$, and requiring that X be "positive semidefinite" in the sense that $v^T X v \geq 0$ for all $v \in \mathbb{R}^n$. Since

$$\begin{aligned} v^T \left(\tfrac{1}{2}(X + X^T)\right) v &= \tfrac{1}{2} v^T X v + \tfrac{1}{2} v^T X^T v \\ &= \tfrac{1}{2} v^T X v + \tfrac{1}{2} v^T X v \\ &= v^T X v, \end{aligned}$$

this requirement turns out to be equivalent to requiring that the *symmetric part* of X, $\frac{1}{2}(X + X^T)$, be positive semidefinite. This specific problem, which was the starting point for this thesis, does not seem to have been previously studied.

We can state the NS-SDLS problem as follows.

$$\begin{array}{ll} \inf & \frac{1}{2}\|AX - B\|_F^2 \\ \text{s.t.} & \frac{1}{2}(X + X^T) \succeq 0 \end{array} \tag{2.19}$$

Here we have the instance of problem (2.3) where $\mathbf{E} = \mathbb{R}^{n \times n}$, $\mathbf{Y} = \mathcal{S}^n$, $H = \mathcal{S}_+^n$, $f(X) = \frac{1}{2}\|AX - B\|_F^2$, and $g(X) = -\frac{1}{2}(X + X^T)$. Proposition 2.2.2 states that f is convex over $\mathbb{R}^{n \times n}$, and g being a linear function implies that it is an H-convex function.

This problem then has the Lagrangian function $L : \mathbb{R}^{n\times n} \times \mathcal{S}^n \to \mathbb{R}$ defined as

$$\begin{aligned} L(X;\Lambda) &= \tfrac{1}{2}\|AX - B\|_F^2 - \langle \Lambda, \tfrac{1}{2}(X + X^T)\rangle \\ &= \tfrac{1}{2}\|AX - B\|_F^2 - \tfrac{1}{2}\langle \Lambda, X\rangle - \tfrac{1}{2}\langle \Lambda, X^T\rangle \\ &= \tfrac{1}{2}\|AX - B\|_F^2 - \langle \Lambda, X\rangle, \end{aligned}$$

and therefore satisfies

$$\nabla_X L(X;\Lambda) = A^T(AX - B) - \Lambda.$$

Thus, with Proposition 2.4.1 in mind, and given that $\hat{X} = I$ implies that the Slater condition holds, we find that $\bar{X} \in \mathbb{R}^{n\times n}$ is optimal for the NS-SDLS problem if and only if there exists a slack matrix $\bar{S} \in \mathcal{S}^n$ and a Lagrange multiplier matrix $\bar{\Lambda} \in \mathcal{S}^n$ such that the following KKT conditions hold.

$$\left.\begin{aligned} A^T(A\bar{X} - B) &= \bar{\Lambda} \\ \tfrac{1}{2}(\bar{X} + \bar{X}^T) &= \bar{S} \\ \bar{\Lambda}\bar{S} &= \mathbf{0} \\ \bar{S},\, \bar{\Lambda} &\succeq \mathbf{0} \end{aligned}\right\} \tag{2.20}$$

Again, we can use Theorem 2.4.3, Lemma 2.4.4, and the fact that the set of feasible points,

$$D := \{X \in \mathbb{R}^{n\times n} \mid \tfrac{1}{2}(X + X^T) \succeq \mathbf{0}\},$$

is nonempty and closed to get the following result.

Theorem 2.4.6 *If A has full column rank, then the NS-SDLS problem has a unique optimal solution.*

2.4.3 The linear matrix inequality least squares problem (LMI-LS)

The third problem under consideration is actually a generalization of the first two problems. Here we seek a least squares solution to the linear equation $Ax = b$, where $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and we require that x satisfies a *linear matrix inequality*, i.e., that a linear combination of symmetric matrices be positive semidefinite. More specifically, the LMI-LS problem is described as

$$\begin{array}{ll} \inf & \tfrac{1}{2}\|Ax - b\|_2^2 \\ \text{s.t.} & \mathcal{K}x \preceq C \end{array} \tag{2.21}$$

where

$$\mathcal{K}x = \sum_{i=1}^{n} x_i K_i$$

and $K_1, \ldots, K_n, C \in \mathcal{S}^k$. In terms of problem (2.3), we have $\mathbf{E} = \mathbb{R}^n$, $\mathbf{Y} = \mathcal{S}^k$, $H = \mathcal{S}^k_+$, $f(x) = \frac{1}{2}\|Ax - b\|_2^2$, and $g(x) = \mathcal{K}x - C$. The constraints here are defined by an *affine* function (a linear function plus a constant), which satisfies $g(tx + (1-t)y) = tg(x) + (1-t)g(y)$ for all $x, y \in \mathbb{R}^n$ and $t \in [0,1]$. Therefore, g is an H-convex function, and the convexity of f follows from Proposition 2.2.2.

To show why this formulation is equivalent to the previous two problems, consider, for example, the NS-SDLS problem with $n = 2$. Then

$$X = \begin{bmatrix} x_1 & x_3 \\ x_2 & x_4 \end{bmatrix}$$

and we have the constraints

$$\tfrac{1}{2}(X + X^T) = \begin{bmatrix} x_1 & \frac{1}{2}(x_2 + x_3) \\ \frac{1}{2}(x_2 + x_3) & x_4 \end{bmatrix} \succeq \mathbf{0},$$

which is equivalent to the constraints of the LMI-LS problem, with $x = (x_1, x_2, x_3, x_4)^T$ and

$$K_1 = \begin{bmatrix} -1 & 0 \\ 0 & 0 \end{bmatrix}, \; K_2 = K_3 = \begin{bmatrix} 0 & -\frac{1}{2} \\ -\frac{1}{2} & 0 \end{bmatrix}, \; K_4 = \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}, \text{ and } C = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

If the matrices associated with the NS-SDLS problem are $\tilde{A}$ and $\tilde{B} = \begin{bmatrix} \tilde{b}_1 & \tilde{b}_2 \end{bmatrix}$, then setting

$$A = \begin{bmatrix} \tilde{A} & \mathbf{0} \\ \mathbf{0} & \tilde{A} \end{bmatrix} \text{ and } b = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \end{bmatrix}$$

implies that $\frac{1}{2}\|Ax - b\|_2^2 = \frac{1}{2}\|\tilde{A}X - \tilde{B}\|_F^2$. In a similar manner, it can also be shown that the SDLS problem is a special case of the LMI-LS problem.

The observation that this third problem generalizes the first two problems already gives us more than enough justification for studying it. However, a further reason to study the LMI-LS problem is that its constraints are the same as those arising in the area of (linear) semidefinite optimization (see the recent survey paper by Todd, [27]). A *semidefinite programming problem* (SDP) is described as

$$\begin{array}{ll} \inf & \langle C, X \rangle \\ \text{s.t.} & \mathcal{K}^* X = d, \\ & X \succeq \mathbf{0} \end{array} \tag{2.22}$$

where $\mathcal{K}^* X = (\langle K_i, X \rangle)_{i=1}^n$, and whose dual problem is described as

$$\begin{array}{ll} \sup & d^T \lambda \\ \text{s.t.} & \mathcal{K}\lambda \preceq C. \end{array} \tag{2.23}$$

It is also instructive to note that the KKT conditions for the SDP problem are described as

$$\left.\begin{array}{rcl} \mathcal{K}\bar{\lambda} + \bar{S} & = & C, \\ \mathcal{K}^* \bar{X} & = & d, \\ \bar{X}\bar{S} & = & \mathbf{0}, \\ \bar{S}, \bar{X} & \succeq & \mathbf{0}. \end{array}\right\} \tag{2.24}$$

The linear functions $\mathcal{K} : \mathbb{R}^n \to \mathcal{S}^k$ and $\mathcal{K}^* : \mathcal{S}^k \to \mathbb{R}^n$ described here are *adjoints* of each other, in that $\langle \Lambda, \mathcal{K}x \rangle = \langle \mathcal{K}^* \Lambda, x \rangle$ for all $\Lambda \in \mathcal{S}^k$ and $x \in \mathbb{R}^n$. This is in fact easy to show:

$$\begin{aligned} \langle \Lambda, \mathcal{K}x \rangle &= \sum_{i=1}^n x_i \langle \Lambda, K_i \rangle \\ &= x^T (\mathcal{K}^* \Lambda) \\ &= \langle \mathcal{K}^* \Lambda, x \rangle . \end{aligned}$$

The fact that the functions $\mathcal{K}$ and $\mathcal{K}^*$ are adjoints allows us to show that the Lagrangian function for problem (2.21), $L : \mathbb{R}^n \times \mathcal{S}^k \to \mathbb{R}$, satisfies

$$\begin{aligned} L(x;\Lambda) &= \tfrac{1}{2}\|Ax-b\|_2^2 + \langle \Lambda, \mathcal{K}x - C\rangle \\ &= \tfrac{1}{2}\|Ax-b\|_2^2 + \langle \mathcal{K}^*\Lambda, x\rangle - \langle \Lambda, C\rangle, \end{aligned}$$

which in turn implies that

$$\nabla_x L(x;\Lambda) = A^T(Ax-b) + \mathcal{K}^*\Lambda.$$

Unlike the problems SDLS and NS-SDLS, the Slater condition (2.8) does not always hold for the LMI-LS problem. However, if the Slater condition does hold for this problem, then we have that $\bar{x} \in \mathbb{R}^n$ is optimal if and only if there exist $\bar{S}, \bar{\Lambda} \in \mathcal{S}^k$ such that the KKT conditions (2.25) hold.

$$\left.\begin{aligned} A^T(A\bar{x}-b) + \mathcal{K}^*\bar{\Lambda} &= \mathbf{0} \\ \mathcal{K}\bar{x} + \bar{S} &= C \\ \bar{\Lambda}\bar{S} &= \mathbf{0} \\ \bar{S},\, \bar{\Lambda} &\succeq \mathbf{0} \end{aligned}\right\} \tag{2.25}$$

For example, if the set of matrices $\{K_1, \ldots, K_n\}$ spans $\mathcal{S}^k$ (i.e., $\operatorname{span}\{K_1, \ldots, K_n\} = \mathcal{S}^k$), then the function $\mathcal{K}$ maps $\mathbb{R}^n$ onto $\mathcal{S}^k$. In this case, the Slater condition (2.8) clearly holds: $\{x \in \mathbb{R}^n \mid \mathcal{K}x \prec C\} \neq \emptyset$.

For this more general problem, we have to be a little more careful when stating the existence theorem. The set of feasible points,

$$D := \{x \in \mathbb{R}^n \mid \mathcal{K}x \preceq C\},$$

can be shown to be closed using the fact that $\mathcal{K}$ is continuous and $\mathcal{S}^k_+$ is closed. However, we do not know if D is nonempty. These facts give rise to the following theorem.

Theorem 2.4.7 *If A has full column rank and the set of feasible points is nonempty, then the LMI-LS problem has a unique optimal solution.*

Again we notice that a useful assumption which guarantees that the set of feasible points is nonempty is to have $\operatorname{span}\{K_1, \ldots, K_n\} = \mathcal{S}^k$.

More on the assumption $\operatorname{span}\{K_1, \ldots, K_n\} = \mathcal{S}^k$

Twice now, we have found the assumption

$$\operatorname{span}\{K_1, \ldots, K_n\} = \mathcal{S}^k \tag{2.26}$$

to be very useful. First of all, when assuming that A has full column rank, assuming (2.26) implies that the LMI-LS problem has a unique optimal solution. Secondly, assuming (2.26) implies that optimality is characterized by the KKT conditions (2.25). We now take some time to investigate some other useful implications of assuming (2.26). We begin by giving the following definition.

Definition 2.4.8 *Let* $\mathbf{E}$ *and* $\mathbf{Y}$ *be Euclidean spaces, and let* $T : \mathbf{E} \to \mathbf{Y}$ *be a linear function. We define the null space of* T *as*

$$\mathit{null}\, T = \{x \in \mathbf{E} \mid Tx = \mathbf{0}\}$$

and the range of T *as*

$$\mathit{range}\, T = \{y \in \mathbf{Y} \mid Tx = y, \text{ for some } x \in \mathbf{E}\}.$$

Given a subset Ω *of* $\mathbf{E}$*, we define the orthogonal complement of* Ω *as*

$$\Omega^{\perp} = \{x \in \mathbf{E} \mid \langle x, v\rangle = 0, \text{ for all } v \in \Omega\}.$$

We immediately have the following familiar result from linear algebra.

Proposition 2.4.9 *Let* $\mathcal{K} : \mathbb{R}^n \to \mathcal{S}^k$ *be defined as in the LMI-LS problem (2.21). Then* $\mathit{null}(\mathcal{K}^*) = (\mathit{range}\, \mathcal{K})^{\perp}$.

Proof. The result follows from the following series of implications.

$$\begin{aligned} U \in \text{null}(\mathcal{K}^*) &\Leftrightarrow \mathcal{K}^*U = \mathbf{0} \\ &\Leftrightarrow \langle \mathcal{K}^*U, x\rangle = 0, \text{ for all } x \in \mathbb{R}^n \\ &\Leftrightarrow \langle U, \mathcal{K}x\rangle = 0, \text{ for all } x \in \mathbb{R}^n \\ &\Leftrightarrow \langle U, V\rangle = 0, \text{ for all } V \in \text{range } \mathcal{K} \\ &\Leftrightarrow U \in (\text{range } \mathcal{K})^{\perp} \end{aligned}$$

■

We will find the following corollary to the preceding result to be extremely useful in the upcoming discussion.

Corollary 2.4.10 *Let* $\mathcal{K} : \mathbb{R}^n \to \mathcal{S}^k$ *be defined as in the LMI-LS problem (2.21). Assume that (2.26) is true; that is,* $\text{span}\{K_1, \ldots, K_n\} = \mathcal{S}^k$. *Then* $\frac{1}{2}k(k+1) \leq n$. *Moreover, if* $\Lambda \in \mathcal{S}^k$ *is nonzero, then* $\mathcal{K}^*\Lambda \neq \mathbf{0}$.

Proof. It is clear that the dimension of $\text{span}\{K_1, \ldots, K_n\}$ can be at most n. However, by assumption, the dimension of $\text{span}\{K_1, \ldots, K_n\}$ equals $\frac{1}{2}k(k+1)$, which gives us our first claim. By the definition of $\mathcal{K}$, we have range $\mathcal{K} = \text{span}\{K_1, \ldots, K_n\}$. Thus, range $\mathcal{K} = \mathcal{S}^k$, which implies that $\text{null}(\mathcal{K}^*) = (\text{range } \mathcal{K})^{\perp} = \{\mathbf{0}\}$. Therefore, if $\Lambda \in \mathcal{S}^k$ is nonzero, then $\Lambda \notin \text{null}(\mathcal{K}^*)$, which implies that $\mathcal{K}^*\Lambda \neq \mathbf{0}$. ■

2.5 The Semidefinite Linear Complementarity Problem

A fantastic connection between our three semidefinite least squares problems was discovered while researching the interior-point methods which we will be using to compute solutions to our problems (see Chapter 3). It turns out that each of our three problems can be stated as a *semidefinite linear complementarity problem* (SDLCP), which was first introduced by Kojima, Shinoh, and Hara [13] as an extension of the LCP, or *(monotone) linear complementarity problem* (2.27), and as a generalization of the SDP, or *semidefinite programming problem* (2.22). For comparative purposes, we state the definition of the LCP (see [31, Chap. 8]), which is the problem of finding vectors x and y in $\mathbb{R}^n$ such that

$$y = Mx + q, \quad x \geq \mathbf{0}, \quad y \geq \mathbf{0}, \quad \text{and} \quad x^T y = 0, \tag{2.27}$$

where q is a vector in $\mathbb{R}^n$ and M is an $n \times n$ positive semidefinite (but not necessarily symmetric) matrix; i.e., $u^T M u \geq 0$ for all $u \in \mathbb{R}^n$.

On the other hand, the SDLCP is defined as the problem of finding matrices X and Y in $\mathcal{S}^n$ such that

$$(X, Y) \in \mathcal{L}, \quad X \succeq \mathbf{0}, \quad Y \succeq \mathbf{0}, \quad \text{and} \quad \langle X, Y \rangle = 0. \tag{2.28}$$

Here $\mathcal{L}$ is a monotone $\frac{1}{2}n(n+1)$-dimensional affine subspace of $\mathcal{S}^n \times \mathcal{S}^n$, where *monotone* means that $\mathcal{L}$ satisfies

$$\langle X' - X, Y' - Y \rangle \geq 0, \text{ for all } (X', Y'), (X, Y) \in \mathcal{L}. \tag{2.29}$$

If strict inequality holds in (2.29) for all distinct pairs (X', Y') and $(X, Y) \in \mathcal{L}$, then we say that $\mathcal{L}$ is *strictly monotone*. Furthermore, we say that a monotone subset $\mathcal{A}$ of $\mathcal{S}^n \times \mathcal{S}^n$ is *maximal* if there is no monotone subset of $\mathcal{S}^n \times \mathcal{S}^n$ which properly contains $\mathcal{A}$. The following result from a paper by Shida, Shindoh, and Kojima [26] provides insight into the need for $\mathcal{L}$ to have dimension $\frac{1}{2}n(n+1) = \dim \mathcal{S}^n$.

Lemma 2.5.1 ([26, p. 388]) *Let* $\mathbf{E}$ *be a Euclidean space. Suppose* $\mathcal{L}$ *is a monotone affine subspace of* $\mathbf{E} \times \mathbf{E}$. *Then* $\mathcal{L}$ *is maximal if and only if* $\dim \mathcal{L} = \dim \mathbf{E}$.

The next result, which is a slightly modified version of a result from the same paper, illustrates the connection between the definitions of the LCP and the SDLCP, and will be a useful tool for showing that the SDLS (2.16), NS-SDLS (2.19), and LMI-LS (2.21) problems can each be stated as an SDLCP (2.28).

Corollary 2.5.2 ([26, p. 389]) *Let* $\mathbf{E}$ *be a Euclidean space. Suppose that* $\mathcal{L}$ *is an affine subspace of* $\mathbf{E} \times \mathbf{E}$ *given by*

$$\mathcal{L} = \{(x, y) \in \mathbf{E} \times \mathbf{E} \mid y = \mathcal{M}x + q\}, \tag{2.30}$$

where $\mathcal{M} : \mathbf{E} \to \mathbf{E}$ *is a linear function and* $q \in \mathbf{E}$. *Then* $\mathcal{L}$ *is monotone and maximal if and only if* $\mathcal{M}$ *is positive semidefinite; i.e.,* $\langle u, \mathcal{M}u \rangle \geq 0$ *for every* $u \in \mathbf{E}$.

We conclude with some further notation. We define the *set of feasible solutions*, the *set of interior feasible solutions*, and the *set of solutions* of the SDLCP (2.28) as

$$\begin{aligned} \mathcal{L}_+ &= \{(X,Y) \in \mathcal{L} \mid X \succeq \mathbf{0},\ Y \succeq \mathbf{0}\}, && (2.31)\\ \mathcal{L}_{++} &= \{(X,Y) \in \mathcal{L} \mid X \succ \mathbf{0},\ Y \succ \mathbf{0}\}, && (2.32)\\ \mathcal{L}_* &= \{(X,Y) \in \mathcal{L}_+ \mid \langle X,Y\rangle = 0\}, && (2.33) \end{aligned}$$

respectively. Thus we can state the SDLCP as the problem of finding a pair of matrices (X,Y) in $\mathcal{L}_*$.

2.5.1 The SDLS problem is an SDLCP

Recall that the semidefinite least squares (SDLS) problem (2.16) is defined as

$$\begin{array}{ll} \inf & \frac{1}{2}\|AX - B\|_F^2 \\ \text{s.t.} & X \succeq \mathbf{0} \end{array},$$

where $A, B \in \mathbb{R}^{m\times n}$ and $X \in \mathcal{S}^n$. We were able to show in Section 2.4.1 that the SDLS problem can be solved by finding symmetric matrices X and Λ satisfying the following KKT equations:

$$\left.\begin{aligned} A^T(AX - B) &= Z,\\ \tfrac{1}{2}(Z + Z^T) &= \Lambda,\\ \langle X, \Lambda\rangle &= 0,\\ X,\ \Lambda &\succeq \mathbf{0}. \end{aligned}\right\} \qquad (2.34)$$

Letting $\mathcal{M} : \mathcal{S}^n \to \mathcal{S}^n$ and $Q \in \mathcal{S}^n$ be defined as

$$\begin{aligned} \mathcal{M}X &= \tfrac{1}{2}(A^TAX + XA^TA), \quad \text{and}\\ Q &= -\tfrac{1}{2}(A^TB + B^TA), \end{aligned}$$

and letting

$$\mathcal{L} = \{(X,\Lambda) \in \mathcal{S}^n \times \mathcal{S}^n \mid \Lambda = \mathcal{M}X + Q\},$$

we see that (X,Λ) satisfies the KKT equations (2.34) if and only if (X,Λ) satisfies

$$(X,\Lambda) \in \mathcal{L}, \quad X \succeq \mathbf{0}, \quad \Lambda \succeq \mathbf{0}, \quad \text{and} \quad \langle X,\Lambda\rangle = 0. \qquad (2.35)$$

Since $\mathcal{M}$ is clearly linear, by showing that $\mathcal{M}$ is positive semidefinite, we can use Lemma 2.5.1 and Corollary 2.5.2 to conclude that the SDLS problem (2.16) is indeed an SDLCP (2.28).

To show that $\mathcal{M}$ is positive semidefinite, we first let $U \in \mathcal{S}^n$. Then

$$\begin{aligned} \langle U, \mathcal{M}U\rangle &= \langle U, \tfrac{1}{2}(A^TAU + UA^TA)\rangle\\ &= \tfrac{1}{2}[\mathrm{tr}(UA^TAU) + \mathrm{tr}(UUA^TA)]\\ &= \tfrac{1}{2}[\mathrm{tr}(UA^TAU) + \mathrm{tr}(UA^TAU)]\\ &= \|AU\|_F^2 \ \geq\ 0, \end{aligned}$$

where the third equality follows from the fact that $\mathrm{tr}(UV) = \mathrm{tr}(VU)$ for all $V \in \mathbb{R}^{n\times n}$ (see [27, p. 521]). Thus, $\mathcal{M}$ is positive semidefinite. We also notice that if A has full column rank, and U is nonzero, then the last inequality would be strict, which shows that $\mathcal{M}$ is positive definite in this case.

2.5.2 The NS-SDLS problem is an SDLCP

We stated the nonsymmetric semidefinite least squares (NS-SDLS) problem (2.19) as

$$\begin{array}{ll} \inf & \frac{1}{2}\|AX - B\|_F^2 \\ \text{s.t.} & \frac{1}{2}(X + X^T) \succeq \mathbf{0} \end{array},$$

where $A, B \in \mathbb{R}^{m\times n}$ and $X \in \mathbb{R}^{n\times n}$. Recall that in Section 2.4.2, we showed that we can solve the NS-SDLS problem by finding symmetric matrices X and Λ satisfying the following KKT equations:

$$\left.\begin{array}{rcl} A^T(AX - B) & = & \Lambda, \\ \frac{1}{2}(X + X^T) & = & S, \\ \langle \Lambda, S\rangle & = & 0, \\ \Lambda,\ S & \succeq & \mathbf{0}. \end{array}\right\} \tag{2.36}$$

To show that the KKT equations (2.36) can be stated as an SDLCP (2.28), we assume that A has full column rank and let $\mathcal{M} : \mathcal{S}^n \to \mathcal{S}^n$ and $Q \in \mathcal{S}^n$ be defined as

$$\begin{array}{rcl} \mathcal{M}\Lambda & = & \frac{1}{2}(G^{-1}\Lambda + \Lambda G^{-1}), \quad \text{and} \\ Q & = & \frac{1}{2}(G^{-1}A^TB + B^TAG^{-1}), \end{array}$$

where $G = A^TA$ is positive definite. Letting

$$\mathcal{L} = \{(\Lambda, S) \in \mathcal{S}^n \times \mathcal{S}^n \mid S = \mathcal{M}\Lambda + Q\},$$

we see that the equations (2.36) imply

$$(\Lambda, S) \in \mathcal{L}, \quad \Lambda \succeq \mathbf{0}, \quad S \succeq \mathbf{0}, \quad \text{and} \quad \langle \Lambda, S\rangle = 0. \tag{2.37}$$

Conversely, if (Λ, S) satisfy (2.37), defining $X := G^{-1}(\Lambda + A^TB)$ gives us a solution of (2.36). Again we see that $\mathcal{M}$ is clearly linear, and by showing that $\mathcal{M}$ is positive definite, we can use Lemma 2.5.1 and Corollary 2.5.2 to conclude that the NS-SDLS problem (2.19) is also an SDLCP (2.28).

To show that $\mathcal{M}$ is positive definite, we first let $U \in \mathcal{S}^n$ be nonzero. Then

$$\begin{array}{rcl} \langle U, \mathcal{M}U\rangle & = & \langle U, \frac{1}{2}(G^{-1}U + UG^{-1})\rangle \\ & = & \frac{1}{2}[\mathrm{tr}(UG^{-1}U) + \mathrm{tr}(UUG^{-1})] \\ & = & \mathrm{tr}(UG^{-1}U) \\ & = & \|\sqrt{G^{-1}}U\|_F^2 \ > \ 0, \end{array}$$

where we used the same trick as before to get the third equality, and the inequality follows from the fact that G^{-1} is positive definite, and thus $\sqrt{G^{-1}}$ is positive definite.

2.5.3 The LMI-LS problem is an SDLCP

As before, we state the linear matrix inequality least squares problem (2.21) as,

$$\begin{array}{ll} \inf & \frac{1}{2}\|Ax - b\|_2^2 \\ \text{s.t.} & \mathcal{K}x \preceq C \end{array},$$

where

$$\mathcal{K}x = \sum_{i=1}^{n} x_i K_i,$$

and $K_1, \ldots, K_n, C \in \mathcal{S}^k$. Here we also have $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $x \in \mathbb{R}^n$. In this context, we make the following assumptions:

1. A has full column rank; and
2. the set of strictly feasible points, $\{x \in \mathbb{R}^n \mid \mathcal{K}x \prec C\}$, is nonempty.

Under these assumptions, we know from Section 2.4.3 that there is a unique optimal solution to the LMI-LS problem, and that to find this solution we need only find a vector $x \in \mathbb{R}^n$ and matrices $S, \Lambda \in \mathcal{S}^k$ which satisfy the following KKT equations:

$$\left.\begin{array}{rcl} A^T(Ax - b) + \mathcal{K}^*\Lambda &=& \mathbf{0}, \\ \mathcal{K}x + S &=& C, \\ \langle \Lambda, S \rangle &=& 0, \\ \Lambda,\ S &\succeq& \mathbf{0}. \end{array}\right\} \tag{2.38}$$

We would now like to show that these KKT equations (2.38) can be stated as an SDLCP (2.28). We begin by rewriting the first equation of (2.38) as

$$x = G^{-1}(A^T b - \mathcal{K}^*\Lambda),$$

where $G = A^T A$, and substituting into the second equation of (2.38) to get

$$S = \mathcal{K}G^{-1}\mathcal{K}^*\Lambda + (C - \mathcal{K}G^{-1}A^T b).$$

Thus, if we define $\mathcal{M} : \mathcal{S}^k \to \mathcal{S}^k$ and $Q \in \mathcal{S}^k$ as

$$\begin{array}{rcl} \mathcal{M}\Lambda &=& \mathcal{K}G^{-1}\mathcal{K}^*\Lambda, \quad \text{and} \\ Q &=& C - \mathcal{K}G^{-1}A^T b, \end{array}$$

and let

$$\mathcal{L} = \{(\Lambda, S) \in \mathcal{S}^n \times \mathcal{S}^n \mid S = \mathcal{M}\Lambda + Q\},$$

then a solution of the KKT equations (2.38) gives us a pair (Λ, S) which satisfies

$$(\Lambda, S) \in \mathcal{L}, \quad \Lambda \succeq \mathbf{0}, \quad S \succeq \mathbf{0}, \quad \text{and} \quad \langle \Lambda, S \rangle = 0. \tag{2.39}$$

Conversely, if (Λ, S) satisfy (2.39), defining $x := G^{-1}(A^T b - \mathcal{K}^*\Lambda)$ gives us a solution of (2.38).

Since $\mathcal{M}$ is composed from linear functions, it too is linear. Now we would like to show that $\mathcal{M}$ is also positive semidefinite. Letting $U \in \mathcal{S}^k$, we have

$$\begin{array}{rcl} \langle U, \mathcal{M}U \rangle &=& \langle U, \mathcal{K}G^{-1}\mathcal{K}^*U \rangle \\ &=& \langle \mathcal{K}^*U, G^{-1}\mathcal{K}^*U \rangle \\ &=& v^T G^{-1} v \ \geq\ 0, \end{array}$$

where $v = \mathcal{K}^* U \in \mathbb{R}^n$. Thus, $\mathcal{M}$ is positive semidefinite. Also, if we assume that

$$\text{span}\{K_1, \ldots, K_n\} = \mathcal{S}^k,$$

then having $U \neq \mathbf{0}$ implies that $v \neq \mathbf{0}$ (see Corollary 2.4.10), which then makes the inequality above strict by the fact that G^{-1} is positive definite; therefore, under this assumption, $\mathcal{M}$ is positive definite.

Since $\mathcal{M}$ is positive semidefinite, we know from Lemma 2.5.1 and Corollary 2.5.2 that $\mathcal{L}$ is indeed a monotone $\frac{1}{2}k(k+1)$-dimensional subspace of $\mathcal{S}^k \times \mathcal{S}^k$, showing that we have stated the LMI-LS problem (2.21) as an SDLCP (2.28).

Chapter 3
Interior-Point Methods and Algorithms

3.1 Overview

Based on the results of Chapter 2 that describe when the SDLS (2.16), NS-SDLS (2.19), and LMI-LS (2.21) problems have a unique optimal solution, we will hence forth assume that the coefficient matrix A in these problems has full column rank and that $\{x \in \mathbb{R}^n \mid \mathcal{K}x \prec C\} \neq \emptyset$ in the LMI-LS problem. Using these facts, we are now interested in describing algorithms to generate a close approximation to the unique optimal solution of each of these problems.

According to Nocedal and Wright [21, p. 7], the basic idea behind all optimization algorithms is to start at some initial guess and iterate through a sequence of points until we are "close enough" to an optimal solution of the problem. Many of these optimization algorithms can be seen as following the framework provided in Figure 3.1; in each iteration they first choose a direction in which to move and then they choose the distance to travel. Interior-point methods for constrained convex optimization problems do exactly this, with the additional requirement that all iterates remain in the interior of the feasible region. Depending on whether the iterates are primal feasible points, dual feasible points, or primal-dual feasible points, an interior-point method will be classified as a primal method, a dual method, or a primal-dual method. We also find that the most important interior-point methods fall under the classes of path-following methods or potential-reduction methods.

Algorithm 3.1

Given an initial point $x \in \mathbf{E}$.

While x is not "close enough" to an optimal solution, **do**

1. Find a suitable direction $d \in \mathbf{E}$.
2. Find a suitable step length $\theta > 0$.
3. Update $x := x + \theta d$.

End

Figure 3.1: A framework for many optimization algorithms

The path-following methods for convex programming problems use a path defined in the feasible region that traces a route to the optimal solution, and, at the same time, stays as far away from the boundary of the feasible region as possible. By following this path, the iterates are guided toward the solution of the problem before getting too close to the boundary. This technique is used because it was noticed that if iterates are still far from the solution, but close to the boundary, then taking steps toward the solution can be hindered by the requirement that all iterates remain in the interior of the feasible region. This situation can force us to choose very small step lengths, making the overall progress of the algorithm extremely slow. See [21, p. 398] for a description of this behaviour in the case of linear programming.

Potential-reduction methods take a different approach by defining a potential function that is small both at the optimal solution and when we are far away from the boundary. By choosing directions and step lengths with the purpose of reducing the potential function at each step, our iterates will tend toward the solution of the problem while staying away from the boundary of the feasible region. See [30] for a description of a potential-reduction method used for solving the SDP problem (2.22).

For our purposes here, we will only be considering primal-dual path-following methods for solving the SDLS, NS-SDLS, and LMI-LS problems. As our guidance, we will be looking at the primal-dual path-following methods used for solving the SDP problem (2.22), which are described in [27], and the same methods for solving the nonnegative least squares problem (NNLS),

$$\begin{array}{ll} \inf & \frac{1}{2}\|Ax-b\|_2^2 \\ \text{s.t.} & x \geq \mathbf{0}, \end{array} \tag{3.1}$$

which are described in [23]. We do this because, in many ways, the SDLS and NS-SDLS problems are extensions of the NNLS problem from the vector case to the matrix case. Further, the LMI-LS problem can be seen as a combination of the SDP and NNLS problems, with the constraints coming from the SDP problem and the objective function coming from the NNLS problem. In fact, we can also notice many similarities in the KKT conditions for the NNLS problem,

$$\left.\begin{array}{rcll} A^T(A\bar{x}-b) & = & \bar{\lambda}, & \\ \bar{\lambda}_i\bar{x}_i & = & 0, & \text{for } i=1,\ldots,n, \\ \bar{x},\,\bar{\lambda} & \geq & \mathbf{0}, & \end{array}\right\} \tag{3.2}$$

to the KKT conditions (2.17) and (2.20). Notice that the conditions (3.2) follow from applying the results of Chapter 2 to problem (2.3) with $\mathbf{E} = \mathbf{Y} = \mathbb{R}^n$, $H = \mathbb{R}^n_+$, $f(x) = \frac{1}{2}\|Ax-b\|_2^2$, and $g(x) = -x$, and further that if $x, \lambda \geq \mathbf{0}$, then $\langle \lambda, x\rangle = \lambda^T x = 0$ if and only if $\lambda_i x_i = 0$ for each i.

It should also be noted that the primary reason for considering primal-dual path-following interior-point methods for solving the SDLS, NS-SDLS, and LMI-LS problems is the success these methods have had for solving the NNLS problem (see [23]) and especially for solving the SDP problem (see [27]). In particular, these methods are not only efficient in practice, they are also efficient in theory in that they have a provably polynomial complexity. That is, these methods have a worst-case running time that is a polynomial function of the size of the problem (see [20] and [24]).

This chapter is summarized as follows. We begin by discussing the log barrier problems associated with each of our three problems in Section 3.2, and show how solutions to these

problems satisfy systems of equations which define our central path. We then provide a uniform treatment of the existence of the central for each of our three problems under the setting of the SDLCP (2.28) in Section 3.3. The existence of this central path which traces a route to the optimal solution provides us with the motivation for discussing in Section 3.4 the interior-point methods under consideration. After providing some convergence results for these interior-point methods in Section 3.5, we discuss the issues involved in implementing these methods in Section 3.6.

3.2 Log Barrier Problems

Consider the function $F : \mathcal{S}^n \to (-\infty, +\infty]$ defined by

$$F(X) = \begin{cases} -\ln\det X, & \text{if } X \succ \mathbf{0} \\ +\infty, & \text{otherwise.} \end{cases} \tag{3.3}$$

Notice that F has the *barrier property* for $\mathcal{S}^n_+$: as $X \in \mathcal{S}^n_{++}$ approaches the boundary of $\mathcal{S}^n_+$, $\det X > 0$ and approaches 0, and thus $-\ln\det X$ approaches $+\infty$. The function F defined here is known as the *logarithmic barrier function* for the cone $\mathcal{S}^n_+$. As we will see, we will be able to use this function to impose the positive semidefinite constraints in our optimization problems. There is also a *logarithmic barrier function* for the cone $\mathbb{R}^n_+$, which is the function $F : \mathbb{R}^n \to (-\infty, +\infty]$ defined by

$$F(x) = \begin{cases} -\sum_{i=1}^n \ln x_i, & \text{if } x > \mathbf{0} \\ +\infty, & \text{otherwise.} \end{cases} \tag{3.4}$$

As put forth in the highly significant book of Nesterov and Nemirovskii [20] (and more recently in the work of Renegar [24]), at the heart of many interior-point methods lies a *self-concordant barrier function* which is responsible for the rate of convergence of the method. The rate of this convergence is in turn based on the parameter of this function (which is referred to as the *complexity value* in [24]), from which it is implied that the smaller this complexity value, the faster the convergence (see [20, p. 42] and [24, pp. 35, 39]). The functions defined in (3.3) and (3.4) are special types of self-concordant barrier functions in that they have the smallest possible complexity value over all such functions for $\mathcal{S}^n_+$ and $\mathbb{R}^n_+$ (see [20, pp. 42, 198] and [24, p. 40]). We will now follow Renegar's geometric presentation and define these terms for a general self-dual cone H in a Euclidean space $\mathbf{E}$.

First of all, we need to define the second derivative, or *Hessian*, of a differentiable function $f : \mathbf{U} \to \mathbb{R}$, where $\mathbf{U}$ is an open subset of $\mathbf{E}$. We denote the Hessian of f at $x \in \mathbf{U}$ as the linear function $\nabla^2 f(x) : \mathbf{E} \to \mathbf{E}$ (if it exists) which satisfies

$$\lim_{\|v\| \to 0} \frac{\|\nabla f(x+v) - \nabla f(x) - \nabla^2 f(x)v\|}{\|v\|} = 0.$$

If $\nabla^2 f(x)$ exists for all $x \in \mathbf{U}$, then we say that f is *twice differentiable*. If the function $x \mapsto \nabla^2 f(x)$ is a continuous function, then we say that f is *twice continuously differentiable*, in which case, the function $\nabla^2 f(x)$ is self-adjoint for all $x \in \mathbf{U}$ (i.e., $\langle u, \nabla^2 f(x)v \rangle = \langle \nabla^2 f(x)u, v \rangle$ for all $u, v \in \mathbf{E}$) [24, p. 9].

In Theorem 3.2.3 we will show that the derivative of the function F defined in (3.3) is $\nabla F(X) = -X^{-1}$ for all $X \in \mathcal{S}^n_{++}$. Just for interest, we will mention without proof (see [24, p. 9]) that second derivative of this function at $X \in \mathcal{S}^n_{++}$ is the linear function $\nabla^2 F(X) : \mathcal{S}^n \to \mathcal{S}^n$ defined by

$$\nabla^2 F(X)V = X^{-1}VX^{-1}.$$

It is interesting to note the similarity which these formulas for $\nabla F(X)$ and $\nabla^2 F(X)$ have to the well-known scalar case: if $f(x) = -\ln(x)$, then $f'(x) = -x^{-1}$ and $f''(x) = x^{-2}$.

Definition 3.2.1 *Let $\mathbf{E}$ be a Euclidean space, $H \subseteq \mathbf{E}$ be a self-dual cone with nonempty interior, and $F : \operatorname{int} H \to \mathbb{R}$ be a twice continuously differentiable strictly convex function with $\nabla^2 F(x)$ positive definite for all $x \in \operatorname{int} H$ (i.e., $\langle u, \nabla^2 F(x)u \rangle > 0$ for all $u \neq \mathbf{0}$).*

The local inner product at $x \in \operatorname{int} H$ is defined as

$$\langle u, v \rangle_x := \langle u, \nabla^2 F(x)v \rangle ,$$

for all $u, v \in \mathbf{E}$. Furthermore, we define the local norm at $x \in \operatorname{int} H$ as $\|u\|_x := \sqrt{\langle u, u \rangle_x}$ for all $u \in \mathbf{E}$. Using the local inner product at $x \in \operatorname{int} H$ to define the gradient of F at $\bar{x} \in \operatorname{int} H$, we get the local gradient at x of F at $\bar{x}$, denoted as $\nabla F_x(\bar{x})$, and which equals $\nabla^2 F(\bar{x})^{-1} \nabla F(\bar{x})$.

The function F is said to be self-concordant if the following hold (see [24, p. 58]):

1. *For all $x, y \in \operatorname{int} H$, if $\|y - x\|_x < 1$, then*

$$\frac{\|v\|_y}{\|v\|_x} \leq \frac{1}{1 - \|y - x\|_x}, \text{ for all } v \neq \mathbf{0}$$

2. *F has the barrier property for H; that is, if a sequence $\{x_k\} \subset \operatorname{int} H$ converges to a point on the boundary of H, then $F(x_k) \to +\infty$.*

The function F is said to be a (self-concordant) barrier function if F is self-concordant and

$$\vartheta_F := \sup_{x \in \operatorname{int} H} \|\nabla F_x(x)\|_x^2 < +\infty,$$

where ϑ_F is called the complexity value of F.

We will now state the following facts from [20] and [24].

Theorem 3.2.2 *Any self-concordant barrier function F for $\mathcal{S}^n_+$ or $\mathbb{R}^n_+$ must satisfy $\vartheta_F \geq n$. The functions defined in (3.3) and (3.4) are self-concordant barrier functions for $\mathcal{S}^n_+$ and $\mathbb{R}^n_+$, respectively, and each have complexity value $\vartheta_F = n$.*

Therefore, it is in this sense that (3.3) and (3.4) are the best possible self-concordant barrier functions for $\mathcal{S}^n_+$ and $\mathbb{R}^n_+$. In fact, we also have the following important result which can be found in [27], and for reasons of interest we will also present a proof.

Theorem 3.2.3 *The logarithmic barrier function F for $\mathcal{S}^n_+$, defined by (3.3), is a strictly convex function over $\mathcal{S}^n_{++}$ with derivative $\nabla F(X) = -X^{-1}$ for $X \in \mathcal{S}^n_{++}$.*

Proof. Here we present a different proof than the one given in [27], and we begin by showing that $\nabla f(X) = -X^{-1}$ for $X \in \mathcal{S}^n_{++}$.

It is a fairly well-known fact that $\nabla \det(A) = \mathrm{adj}(A)^T$ for $A \in \mathbb{R}^{n\times n}$, where $\mathrm{adj}(A) = (m_{ij})$ is the $n \times n$ adjugate matrix defined by

$$m_{ij} = (-1)^{i+j} \det M_{ji}$$

and M_{ji} is formed by deleting row j and column i from A. This can easily be seen by considering the partial derivative of $\det(A)$ with respect to a_{ij}, which we can calculate by expanding the determinant along row i:

$$\det(A) \;=\; \sum_{k=1}^{n} a_{ik} m_{ki} \;=\; a_{ij} m_{ji} + \sum_{k\neq j} a_{ik} m_{ki}.$$

Since neither m_{ji} nor the summation term at the end of this expression depend on a_{ij}, we have $\frac{\partial \det(A)}{\partial a_{ij}} = m_{ji}$, and the result follows. If A is invertible, we also have that $A^{-1} = \frac{1}{\det(A)}\mathrm{adj}(A)$, and so $\nabla \det(A) = \det(A)A^{-T}$. For $X \in \mathcal{S}^n_{++}$, this result becomes $\nabla \det(X) = \det(X)X^{-1}$, and since $F(X) = -\ln \det X$, we apply the chain rule to get

$$\nabla F(X) \;=\; \frac{-1}{\det(X)} \nabla \det(X) \;=\; -X^{-1}.$$

We follow the advice given in an exercise from [6, p. 40] to show that F is strictly convex over $\mathcal{S}^n_{++}$. There it states that F is convex if and only if

$$\langle \nabla F(X) - \nabla F(Y), X - Y \rangle \geq 0 \quad \text{for all } X, Y \in \mathcal{S}^n_{++},$$

and strictly convex if and only if the inequality is strict whenever $X \neq Y$. Letting $X, Y \in \mathcal{S}^n_{++}$ such that $X \neq Y$, we have

$$\begin{aligned} \langle \nabla F(X) - \nabla F(Y), X - Y \rangle &= \langle Y^{-1} - X^{-1}, X - Y \rangle \\ &= \mathrm{tr}(Y^{-1}X) - 2\mathrm{tr}(I) + \mathrm{tr}(X^{-1}Y) \\ &= \mathrm{tr}\left(\sqrt{X}Y^{-1}\sqrt{X}\right) + \mathrm{tr}\left(\sqrt{X}^{-1}Y\sqrt{X}^{-1}\right) - 2n \;>\; 0, \end{aligned}$$

where the inequality comes from the fact [6, p. 8] that $\mathrm{tr}(Z) + \mathrm{tr}(Z^{-1}) \geq 2n$ for all $Z \in \mathcal{S}^n_{++}$ with equality if and only if $Z = I$. ∎

3.2.1 The SDLS log barrier problem

Notice how we can use the barrier function F defined in (3.3) along with a positive parameter τ to impose the positive semidefinite constraints in the SDLS problem:

$$\min_{X \in \mathcal{S}^n} \quad \tfrac{1}{2}\|AX - B\|_F^2 + \tau F(X) \tag{3.5}$$

Thus we get an unconstrained problem (3.5) called the *SDLS log barrier problem*, whose solution is necessarily positive definite. If $\bar{X} \in \mathcal{S}^n_{++}$ is such a solution, it is clearly characterized as a critical point of the strictly convex objective function $f(\cdot)+\tau F(\cdot)$, where $f(X) = \frac{1}{2}\|AX-B\|_F^2$. That is,

$$\begin{aligned} \nabla(f+\tau F)(\bar{X}) &= \nabla f(\bar{X}) + \tau \nabla F(\bar{X}) \\ &= \tfrac{1}{2}(\bar{Z}+\bar{Z}^T) - \tau \bar{X}^{-1} = \mathbf{0}, \end{aligned}$$

where $\bar{Z} = A^T(A\bar{X}-B)$. Letting $\bar{\Lambda} = \tau \bar{X}^{-1}$, we have the following necessary and sufficient conditions for an optimal solution of (3.5).

$$\left.\begin{aligned} A^T(A\bar{X}-B) &= \bar{Z} \\ \tfrac{1}{2}(\bar{Z}+\bar{Z}^T) &= \bar{\Lambda} \\ \bar{\Lambda}\bar{X} &= \tau\, \mathrm{I} \\ \bar{X},\, \bar{\Lambda} &\succ \mathbf{0} \end{aligned}\right\} \tag{3.6}$$

If $(\bar{X}, \bar{\Lambda})$ satisfies (3.6), then it is a primal-dual feasible point with duality gap

$$\begin{aligned} \mu(\bar{X}, \bar{\Lambda}) &= \langle \bar{\Lambda}, \bar{X} \rangle \\ &= \mathrm{tr}(\bar{\Lambda}\bar{X}) \\ &= n\tau. \end{aligned}$$

We immediately notice the similarity of conditions (3.6) to the KKT conditions (2.17) for the SDLS problem. In Section 3.3, we will show that there is a unique solution (X_τ, Λ_τ) to (3.6) for each $\tau > 0$, and that as τ approaches 0, (X_τ, Λ_τ) approaches the unique optimal solution to the SDLS problem.

3.2.2 The NS-SDLS log barrier problem

Again we use the barrier function for $\mathcal{S}^n_+$ from (3.3) to define the *NS-SDLS log barrier problem* with positive parameter τ:

$$\min_{X \in \mathbb{R}^{n\times n}} \quad \tfrac{1}{2}\|AX-B\|_F^2 + \tau F\left(\tfrac{1}{2}(X+X^T)\right) \tag{3.7}$$

Letting $f(X) = \frac{1}{2}\|AX-B\|_F^2$ and $g(X) = \frac{1}{2}(X+X^T)$, we see that the function which we are minimizing, $f(\cdot)+\tau F(g(\cdot))$, is strictly convex. This can be shown by noticing that

$$g(tX+(1-t)Y) = tg(X) + (1-t)g(Y)$$

for all $X, Y \in \mathbb{R}^{n\times n}$ and $t \in [0,1]$, which implies that $F \circ g$ is convex on its domain. Since the sum of a strictly convex function and a convex function must be strictly convex, we find that we are minimizing a strictly convex function.

A matrix $\bar{X} \in \mathbb{R}^{n\times n}$ is a solution to (3.7) if and only if it is a critical point of the function $f(\cdot)+\tau F(g(\cdot))$, and $\bar{S} = \frac{1}{2}(\bar{X}+\bar{X}^T)$ is necessarily positive definite. First we must determine

$\nabla(F \circ g)(\bar{X})$. Let $D \in \mathbb{R}^{n \times n}$ be a direction, then using the chain rule and the fact that g is linear, and so $g'(\bar{X}; D) = g(D)$, we get

$$\begin{aligned}(F \circ g)'(\bar{X}; D) &= F'(g(\bar{X}); g'(\bar{X}; D)) \\ &= \langle \nabla F(g(\bar{X})), g'(\bar{X}; D) \rangle \\ &= \langle -(\tfrac{1}{2}(\bar{X} + \bar{X}^T))^{-1}, \tfrac{1}{2}(D + D^T) \rangle \\ &= \langle -\bar{S}^{-1}, D \rangle ,\end{aligned}$$

where the last equality follows from the symmetry of $-\bar{S}^{-1}$. Therefore, $\nabla(F \circ g)(\bar{X}) = -\bar{S}^{-1}$, and we have

$$\begin{aligned}\nabla(f + \tau F \circ g)(\bar{X}) &= \nabla f(\bar{X}) + \tau \nabla(F \circ g)(\bar{X}) \\ &= A^T(A\bar{X} - B) - \tau \bar{S}^{-1} = \mathbf{0}.\end{aligned}$$

Now we let $\bar{\Lambda} = \tau \bar{S}^{-1}$, and we get that $\bar{X}$ is optimal for problem (3.7) if and only if it satisfies the following conditions.

$$\left.\begin{aligned} A^T(A\bar{X} - B) &= \bar{\Lambda} \\ \tfrac{1}{2}(\bar{X} + \bar{X}^T) &= \bar{S} \\ \bar{\Lambda}\bar{S} &= \tau I \\ \bar{S}, \bar{\Lambda} &\succ \mathbf{0} \end{aligned}\right\} \tag{3.8}$$

Again we have that the conditions for optimality of the log barrier problem are remarkably similar to the KKT conditions (2.20) for the original problem. Again we notice that if $(\bar{X}, \bar{S}, \bar{\Lambda})$ satisfies (3.8), then it is a primal-dual feasible point of the NS-SDLS problem with duality gap $\mu(\bar{X}, \bar{S}, \bar{\Lambda}) = \langle \bar{\Lambda}, \bar{S} \rangle = n\tau$. Moreover, as we will see in Section 3.3, for each $\tau > 0$, there is a unique solution $(X_\tau, S_\tau, \Lambda_\tau)$ to (3.8), and that this solution approaches the unique optimal solution to the NS-SDLS problem as $\tau \to 0$.

3.2.3 The LMI-LS log barrier problem

We now define the *LMI-LS log barrier problem* using the logarithmic barrier function for $\mathcal{S}^k_+$ with $\tau > 0$ as follows.

$$\min_{x \in \mathbb{R}^n} \quad \tfrac{1}{2}\|Ax - b\|_2^2 + \tau F(C - \mathcal{K}x) \tag{3.9}$$

If we let $f(x) = \frac{1}{2}\|Ax - b\|_2^2$ and $g(x) = C - \mathcal{K}x$, then $\bar{x} \in \mathbb{R}^n$ is a solution to problem (3.9) if and only if $\bar{S} = C - \mathcal{K}\bar{x}$ is positive definite and $\bar{x}$ is a critical point of the function $f(\cdot) + \tau F(g(\cdot))$, which can be shown to be strictly convex by a similar argument as was used for the NS-SDLS log barrier problem (3.7). Let us first compute $\nabla(F \circ g)(\bar{x})$ by considering a direction $d \in \mathbb{R}^n$ and using the chain rule with the fact that $g'(x; d) = -\mathcal{K}d$:

$$\begin{aligned}(F \circ g)'(\bar{x}; d) &= F'(g(\bar{x}); g'(\bar{x}; d)) \\ &= \langle \nabla F(g(\bar{x})), g'(\bar{x}; d) \rangle \\ &= \langle -(C - \mathcal{K}\bar{x})^{-1}, -\mathcal{K}d \rangle \\ &= \langle \mathcal{K}^* \bar{S}^{-1}, d \rangle .\end{aligned}$$

Therefore, we have

$$\begin{aligned}\nabla(f+\tau F\circ g)(\bar{x}) &= \nabla f(\bar{x})+\tau\nabla(F\circ g)(\bar{x})\\ &= A^T(A\bar{x}-b)+\tau\mathcal{K}^*\bar{S}^{-1} \;=\; \mathbf{0},\end{aligned}$$

and letting $\bar{\Lambda}=\tau\bar{S}^{-1}$, we have the following conditions for optimality of problem (3.9).

$$\left.\begin{array}{rcl} A^T(A\bar{x}-b)+\mathcal{K}^*\bar{\Lambda} &=& \mathbf{0}\\ \mathcal{K}\bar{x}+\bar{S} &=& C\\ \bar{\Lambda}\bar{S} &=& \tau I\\ \bar{S},\,\bar{\Lambda} &\succ& \mathbf{0}\end{array}\right\} \tag{3.10}$$

Once again, except for the right hand side of these equations, these conditions are exactly the LMI-LS KKT conditions (2.25). If $(\bar{x},\bar{S},\bar{\Lambda})$ satisfies (3.10) then it is a primal-dual feasible point with duality gap $\mu(\bar{x},\bar{S},\bar{\Lambda})=\langle\bar{\Lambda},\bar{S}\rangle=n\tau$. Finally, as we will see in Section 3.3, under the assumption that there exists a primal-dual feasible point $(\hat{x},\hat{S},\hat{\Lambda})$ of the LMI-LS problem such that $\hat{S},\hat{\Lambda}\succ\mathbf{0}$, we have, for each $\tau>0$, that there exists a unique solution $(x_\tau,S_\tau,\Lambda_\tau)$ to (3.10), and this solution converges to the unique optimal solution of the LMI-LS problem as $\tau\to 0$.

3.3 The Central Path

Now that we have described the log barrier problems associated with our three semidefinite constrained least squares problems, intuitively we can see how the solutions to these problems stay as far away from the boundary of the semidefinite cone as possible, while approaching solutions of our original problems as we let $\tau\to 0$. Each of the systems of equations, (3.6), (3.8), and (3.10), describes what is called the *central path* which we would like the iterates of our algorithms to follow. A key question to be raised now is that of the existence of the central path for each of our problems. Luckily, we are able to give a uniform treatment for all three problems due to the fact that each problem can be stated as an SDLCP (see Section 2.5).

Recall that we were able to write each of our three problems in the form:

$$\text{Find } (X,Y)\in\mathcal{L},\ \text{such that}\ \ X\succeq\mathbf{0},\ \ Y\succeq\mathbf{0},\ \ \text{and}\ \ \langle X,Y\rangle=0.$$

For each of our problems, we were also able to express $\mathcal{L}$ as

$$\mathcal{L}=\{(X,Y)\in\mathcal{S}^p\times\mathcal{S}^p\mid Y=\mathcal{M}X+Q\},$$

with $\mathcal{M}:\mathcal{S}^p\to\mathcal{S}^p$ being a linear positive semidefinite function, and $Q\in\mathcal{S}^p$. Also recall the definitions of $\mathcal{L}_+$, $\mathcal{L}_{++}$, and $\mathcal{L}_*$ (2.31-2.33). In each of our three problems, we know that there is a unique optimal solution $(\bar{X},\bar{Y})$; i.e., $\mathcal{L}_*=\{(\bar{X},\bar{Y})\}$.

Using this notation, we can write the central path equations for each of our problems as

$$(X,Y)\in\mathcal{L},\ \ X\succ\mathbf{0},\ \ Y\succ\mathbf{0},\ \ \text{and}\ \ XY=\tau I,$$

or more succinctly as

$$\left.\begin{array}{rcl}(X,Y) &\in& \mathcal{L}_{++}\\ XY &=& \tau I\end{array}\right\} \tag{3.11}$$

where $\tau>0$. We now state the following result from the paper that introduced the SDLCP.

Theorem 3.3.1 ([13, p. 98]) *If there exists an* $(\hat{X}, \hat{Y}) \in \mathcal{L}_{++}$, *then:*

(1) For all $\tau > 0$, *there exists a unique* $(X_\tau, Y_\tau) \in \mathcal{L}_{++}$ *such that*

$$X_\tau Y_\tau = \tau I.$$

(2) The set $C := \{(X_\tau, Y_\tau) \mid \tau > 0\}$ *forms a smooth path, called the central path.*

(3) $\lim_{\tau \to 0^+} (X_\tau, Y_\tau) = (\bar{X}, \bar{Y}) \in \mathcal{L}_*$.

In order to apply Theorem 3.3.1 to our problems, it suffices to show that there exists an $(\hat{X}, \hat{Y}) \in \mathcal{L}_{++}$. For the first two problems, the SDLS, and the NS-SDLS, showing that $\mathcal{L}_{++} \neq \emptyset$ reduces to showing that the solution to a Lyapunov equation of the form

$$GX + XG = H \tag{3.12}$$

is in $\mathcal{S}^p_{++}$, where $G, H \in \mathcal{S}^p_{++}$. To show this, we will use the Kronecker product, $\otimes$, and symmetric Kronecker product, $\otimes_s$, which we will find much use for later, and so we give a treatment of their properties in Appendix A. Unfortunately, for the LMI-LS problem, even after much effort, we were not able to prove that $\mathcal{L}_{++} \neq \emptyset$ in general, nor were we able to find a counter-example. We begin by stating an important theorem about the Lyapunov equation.

Theorem 3.3.2 *Let* $G, H \in \mathcal{S}^p_{++}$. *Then there exists a unique* $X \in \mathcal{S}^p_{++}$ *such that (3.12) holds.*

Proof. This proof is drawn from [10, p. 268–271]. First of all, we notice that we can rewrite equation (3.12) as

$$(I \otimes G + G \otimes I)\mathrm{vec}(X) = \mathrm{vec}(H).$$

Since $(I \otimes G)^T = I^T \otimes G^T = I \otimes G$ and similarly for the matrix $G \otimes I$, we find that $K := I \otimes G + G \otimes I \in \mathcal{S}^{p^2}$. Furthermore, if $\sigma(G) = \{\lambda_i\}$, then $\sigma(I \otimes G) = \sigma(G \otimes I) = \{\lambda_i\}$, and so G being positive definite implies that K is positive definite. Therefore, $X = \mathrm{mat}(K^{-1}\mathrm{vec}(H)) \in \mathbb{R}^{p \times p}$ is the unique solution to equation (3.12).

To show that X is symmetric, we simply take the transpose of both sides of (3.12) to get

$$X^T G + G X^T = H,$$

and the uniqueness of X implies that $X = X^T$.

Finally, Lyapunov's Theorem [10, p. 96] states that if we are given symmetric matrices G, X, and H satisfying (3.12) such that H is positive definite, then G being positive definite implies that X is positive definite. Therefore, $X \in \mathcal{S}^p_{++}$ is the unique solution to equation (3.12). ■

Recall that for the SDLS problem (2.16), we had $p = n$ and had defined the function $\mathcal{M}$ by

$$\mathcal{M}X = \tfrac{1}{2}(GX + XG),$$

where $G = A^T A \in \mathcal{S}^n_{++}$. Thus, if $(X, Y) \in \mathcal{L}$, then

$$Y = \mathcal{M}X + Q,$$

which can be rewritten as

$$GX + XG = 2(Y - Q).$$

Since $Q \in \mathcal{S}^n$, we can clearly choose $Y \in \mathcal{S}^n_{++}$ so that $H := 2(Y - Q) \in \mathcal{S}^n_{++}$. Applying Theorem 3.3.2, we are now able to conclude that there exists a unique $X \in \mathcal{S}^n_{++}$ satisfying $GX + XG = H$. Therefore, $\mathcal{L}_{++} \neq \emptyset$ for the SDLS problem, which, by Theorem 3.3.1, implies that the SDLS central path exists.

Similarly, for the NS-SDLS problem (2.19), we also had $p = n$, but had defined the function $\mathcal{M}$ by

$$\mathcal{M}X = \tfrac{1}{2}(G^{-1}X + XG^{-1})$$

where $G = A^T A \in \mathcal{S}^n_{++}$, and so $G^{-1} \in \mathcal{S}^n_{++}$. For $(X, Y) \in \mathcal{L}$, we have

$$Y = \mathcal{M}X + Q,$$

which we rewrite as

$$G^{-1}X + XG^{-1} = H,$$

where we can again choose $Y \in \mathcal{S}^n_{++}$ to make $H := 2(Y - Q)$ positive definite. Now Theorem 3.3.2 provides the existence and uniqueness of an $X \in \mathcal{S}^n_{++}$ which satisfies $G^{-1}X + XG^{-1} = H$. Thus, for the NS-SDLS problem, we also have $\mathcal{L}_{++} \neq \emptyset$. Using Theorem 3.3.1, we conclude that the NS-SDLS central path also exists.

Since we were not able to determine whether or not $\mathcal{L}_{++} \neq \emptyset$ is always true for the LMI-LS problem, for the remainder of the discussion we will be assuming that there exists a primal-dual feasible point $(\hat{x}, \hat{S}, \hat{\Lambda})$ of the LMI-LS problem such that $\hat{S}, \hat{\Lambda} \succ \mathbf{0}$.

3.4 Interior-Point Methods for the SDLCP

We are now in a position to describe interior-point methods used for solving the SDLCP (2.28). The structure of our discussion is inspired by material regarding interior-point methods in [21] on the linear programming problem (LP), and in [27] on the SDP (2.22).

Suppose we are given an SDLCP with

$$\mathcal{L} = \{(X, Y) \in \mathcal{S}^p \times \mathcal{S}^p \mid Y = \mathcal{M}X + Q\}, \tag{3.13}$$

where $\mathcal{M} : \mathcal{S}^p \to \mathcal{S}^p$ is a linear positive definite function, and $Q \in \mathcal{S}^p$. Here we will assume that $\mathcal{L}_{++} \neq \emptyset$ and that $\mathcal{L}_* = \{(\bar{X}, \bar{Y})\}$. The basic idea behind interior-point methods for solving the SDLCP is to construct a sequence $\{(X_r, Y_r)\}$ in $\mathcal{S}^p_{++} \times \mathcal{S}^p_{++}$ which converges to the unique solution $(\bar{X}, \bar{Y})$. That is, we require that X_r and Y_r be positive definite for each iteration r. A *feasible*-interior-point method also requires that the positive definite initial point (X_0, Y_0) is in $\mathcal{L}$ so that $Y_r = \mathcal{M}X_r + Q$ in each iteration; an *infeasible*-interior-point method only requires that (X_0, Y_0) is positive definite and the method will approach the set $\mathcal{L}$ as it iterates.

The methods we describe here will be *path-following* methods in that they will use the central path discussed in previous section in order to be guided toward the solution while staying as far away from the boundary of positive semidefinite region as possible. As in the framework described in Figure 3.1, in each iteration we will update our current iterate (X, Y) to $(\tilde{X}, \tilde{Y}) = (X, Y) + \theta(\Delta X, \Delta Y)$ for some suitable direction $(\Delta X, \Delta Y) \in \mathcal{S}^p \times \mathcal{S}^p$ and some suitable step length $\theta > 0$ chosen so that $(\tilde{X}, \tilde{Y})$ is positive definite.

3.4.1 Choosing a suitable direction $(\Delta X, \Delta Y)$

In each iteration, we want to take a step toward a point (X_τ, Y_τ) on the central path, which is defined as the solution of the following system of nonlinear equations.

$$F(X, Y) := \begin{bmatrix} \mathcal{M}X + Q - Y \\ XY - \tau I \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{3.14}$$

Here we choose $\tau = \sigma\hat{\mu}$, where $\hat{\mu} := \langle X, Y \rangle / p$ is the *normalized duality gap*, and σ is a *centering parameter* chosen from the interval $[0, 1]$. When we choose $\sigma = 1$, we will be stepping toward the point on the central path with the same duality gap as our current iterate:

$$\begin{aligned} \mu(X_\tau, Y_\tau) \;=\; \langle X_\tau, Y_\tau \rangle \;&=\; \mathrm{tr}(X_\tau Y_\tau) \\ &=\; \mathrm{tr}(\hat{\mu} I) \\ &=\; p\hat{\mu} \\ &=\; \langle X, Y \rangle \;=\; \mu(X, Y). \end{aligned}$$

Thus, when $\sigma = 1$, we will step toward the center of the feasible region, away from the boundary, but we will make little or no progress toward the solution of the SDLCP. On the other hand, by choosing $\sigma = 0$, we will be stepping directly toward the solution of the SDLCP, but we may be hindered in the next iteration because we end up being too close to the boundary, forcing us to take only very small steps toward the solution. Typically σ is chosen in the open interval $(0, 1)$ in order to balance between these two goals of reducing the duality gap of our iterates and staying as far away from the boundary as possible.

As discussed in [21, Chap. 11], at the heart of many important algorithms for solving nonlinear equations like (3.14) lies *Newton's method.* This is largely due to the rapid convergence properties of this method under fairly reasonable assumptions. The *Newton step* taken in each iteration of Newton's method is defined as the solution of the linear approximation of the equations at the current point. For example, the Newton step taken toward the solution of (3.14) is defined as the direction $(\Delta X, \Delta Y) \in \mathcal{S}^p \times \mathcal{S}^p$ satisfying

$$F(X, Y) + \nabla F(X, Y) \begin{bmatrix} \Delta X \\ \Delta Y \end{bmatrix} = \mathbf{0}.$$

Thus, to calculate the Newton step for (3.14) at our current iterate (X, Y), we need only solve the system of linear equations,

$$\nabla F(X, Y) \begin{bmatrix} \Delta X \\ \Delta Y \end{bmatrix} = -F(X, Y), \tag{3.15}$$

which can also be written as

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ \Delta X Y + X \Delta Y \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau I - XY \end{bmatrix}. \tag{3.16}$$

Notice that the first equation in (3.16) can be rearranged as $Y + \Delta Y = \mathcal{M}(X + \Delta X) + Q$. In the literature for the SDLCP, when describing the Newton step for a general $\mathcal{L}$, they state this first equation as $(X + \Delta X, Y + \Delta Y) \in \mathcal{L}$.

Here arises the key problem that is the focus of much research in interior-point methods for the SDP (see [27]). The function F defined in (3.14) maps a point in $\mathcal{S}^p \times \mathcal{S}^p$ to a point in $\mathcal{S}^p \times \mathbb{R}^{p \times p}$. The reason behind this is that even if the matrices X and Y are symmetric, the matrix XY is usually nonsymmetric. This means that the gradient of F at (X, Y) is a linear function mapping between vector spaces of different dimension. In other words, $\nabla F(X, Y)$ cannot be a bijective function, and hence it is not invertible. We could compute the solution to (3.16) where the matrices ΔX and ΔY are in $\mathbb{R}^{p \times p}$, but we find that, contrary to our wishes, this solution is often nonsymmetric.

Since this problem is due to the second equation in (3.14), that $XY = \tau I$, we need to find a way to replace this equation with an equivalent symmetric equation. A simple way of doing this is to just take the symmetric part of both sides of this equation, from which we get

$$\tfrac{1}{2}(XY + YX) = \tau I. \tag{3.17}$$

This idea was introduced by Alizadeh, Haeberly, and Overton in [2], and the resulting Newton step, defined by

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ \frac{1}{2}(Y\Delta X + \Delta X Y) + \frac{1}{2}(X\Delta Y + \Delta Y X) \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau I - \frac{1}{2}(XY + YX) \end{bmatrix}, \tag{3.18}$$

has been coined the *AHO direction.*

However, there are many other ways of symmetrizing the equation $XY = \tau I$. A whole family of directions was proposed by Monteiro and Zhang (see [18] and [36]), called the *MZ-family of directions.* In [36], Zhang proposed using replacing $XY = \tau I$ in (3.14) with

$$H_P(XY) = \tau I \tag{3.19}$$

where $H_P : \mathbb{R}^{p \times p} \to \mathcal{S}^p$ is the linear transformation defined by

$$H_P(U) = \tfrac{1}{2}\left(PUP^{-1} + P^{-T}U^T P^T\right) \tag{3.20}$$

with $P \in \mathbb{R}^{p \times p}$ being some constant nonsingular matrix. In fact, we have

$$H_P(XY) = \tau I \;\Leftrightarrow\; XY = \tau I$$

for all symmetric matrices X and Y (see [36, p. 377]). Thus, substituting $XY = \tau I$ with (3.19), the Newton step (3.15) can be stated as

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ H_P(\Delta X Y + X \Delta Y) \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau I - H_P(XY) \end{bmatrix}. \tag{3.21}$$

Notice that the AHO direction (3.18) is a member of the MZ-family with $P = I$.

Another important direction in the MZ-family is the *HRVW/KSH/M* (or *HKM*) *direction*. This is the MZ-direction (3.21) defined with $P = \sqrt{Y}$. That is, the HKM direction is obtained by solving

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ \sqrt{Y}\Delta X\sqrt{Y} + \frac{1}{2}(\sqrt{Y}X\Delta Y\sqrt{Y}^{-1} + \sqrt{Y}^{-1}\Delta Y X\sqrt{Y}) \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau I - \sqrt{Y}X\sqrt{Y} \end{bmatrix},$$

or equivalently,

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ \Delta X + \frac{1}{2}(X\Delta Y Y^{-1} + Y^{-1}\Delta Y X) \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau Y^{-1} - X \end{bmatrix}, \tag{3.22}$$

for $(\Delta X, \Delta Y)$. This direction was independently discovered by Helmberg, Rendl, Vanderbei, and Wolkowicz, and by Kojima, Shindoh, and Hara, only to be later rediscovered by Monteiro as a member of the MZ-family. We also mention the *dual-HKM direction* as the MZ-direction (3.21) with $P = \sqrt{X}^{-1}$, which could be obtained by solving the following linear system.

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ \frac{1}{2}(Y\Delta X X^{-1} + X^{-1}\Delta X Y) + \Delta Y \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau X^{-1} - Y \end{bmatrix} \tag{3.23}$$

A third direction from the MZ-family which is still widely used for SDP is the Nesterov-Todd or *NT direction*. This direction is defined as the solution of (3.21) with $P = W^{-\frac{1}{2}}$, where W is the unique positive definite matrix satisfying $WYW = X$. We can even state this matrix W explicitly as

$$W = \sqrt{X}\left(\sqrt{X}Y\sqrt{X}\right)^{-\frac{1}{2}}\sqrt{X}. \tag{3.24}$$

It was shown in [29] that the NT-direction is the solution to the following equivalent linear systems.

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ \Delta X + W\Delta Y W \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau Y^{-1} - X \end{bmatrix}, \quad \begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ W^{-1}\Delta X W^{-1} + \Delta Y \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau X^{-1} - Y \end{bmatrix} \tag{3.25}$$

For each of the directions already mentioned, we also have their corresponding *predictor-corrector* direction. Given a nonsingular direction $P \in \mathbb{R}^{p\times p}$, we compute this direction $(\Delta X, \Delta Y)$ by first computing the predictor direction $(\delta X, \delta Y)$ as the solution of

$$\begin{bmatrix} \mathcal{M}\delta X - \delta Y \\ H_P(\delta X Y + X\delta Y) \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ -H_P(XY) \end{bmatrix}, \tag{3.26}$$

and then correct this direction by $(\widehat{\delta X}, \widehat{\delta Y})$, which is computed as the solution of

$$\begin{bmatrix} \mathcal{M}\widehat{\delta X} - \widehat{\delta Y} \\ H_P(\widehat{\delta X} Y + X\widehat{\delta Y}) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \tau I - H_P(\delta X \delta Y) \end{bmatrix}, \tag{3.27}$$

where $\sigma \in (0, 1]$ and $\tau = \sigma\hat{\mu}$. Together, these two directions give us the predictor-corrector direction, defined as

$$(\Delta X, \Delta Y) := (\delta X, \delta Y) + (\widehat{\delta X}, \widehat{\delta Y}).$$

Clearly, once we have solved (3.26) for the predictor direction, we can compute $(\Delta X, \Delta Y)$ directly as the solution of

$$\begin{bmatrix} \mathcal{M}\Delta X - \Delta Y \\ H_P(\Delta X Y + X \Delta Y) \end{bmatrix} = \begin{bmatrix} Y - \mathcal{M}X - Q \\ \tau I - H_P(XY + \delta X \delta Y) \end{bmatrix}. \tag{3.28}$$

The idea behind this predictor-corrector direction is as follows. First we attempt to reduce the duality gap as much as possible by setting $\sigma = 0$, which may end up taking us far away from the central-path and too close to the boundary of the semidefinite region. We then correct this step by computing a direction with $\sigma \neq 0$ chosen to take us toward a point on the central-path with a duality gap as small as the one obtained by the predictor direction. We can also interpret this as having used second-order information to determine this predictor-corrector direction; if the predictor direction gives us the direction tangent to the trajectory we are following, then the predictor-corrector direction takes into account the curvature of the trajectory as well (see [21, §14.2]). An alternate interpretation of the predictor-corrector direction is that it is the direction which comes from taking a step with $\sigma = 0$, and then a step with $\sigma \in (0, 1]$ (see [27, p. 549]). We choose to give the above description of the predictor-corrector direction to highlight the fact that the two linear systems we must solve, (3.26) and (3.28), both have the same coefficient matrix, which greatly reduces the computational effort required to compute this direction.

We would also like to mention that there are many more search directions and families of search directions that have been studied in the area of SDP. However, the most important of those studied appear to be the AHO, HKM, dual-HKM, NT directions, and their predictor-corrector variants, which we have mentioned above. Moreover, as we will discuss further in Section 3.6, HKM is currently the direction of choice for large SDP problems [22].

Existence and uniqueness of search directions

Now that we have defined a few search directions that can be used in interior-point methods for solving an SDLCP, we would like to know under what conditions these search directions exist. The answer comes from a paper by Shida, Shindoh, and Kojima [26].

Theorem 3.4.1 ([26, p. 392]) *Let $\mathcal{L}$ be given as in (3.13). Suppose that $(X, Y) \in \mathcal{S}^p_{++} \times \mathcal{S}^p_{++}$ and $\tau \geq 0$. Then both the dual-HKM (3.23) and NT (3.25) search directions exist and are unique. Suppose further that $\frac{1}{2}(XY + YX) \in \mathcal{S}^p_+$. Then the AHO search direction (3.18) exists and is unique.*

The assumption that $\frac{1}{2}(XY + YX) \in \mathcal{S}^p_+$ is necessary in Theorem 3.4.1, for even when X and Y are both positive definite, the matrix $\frac{1}{2}(XY + YX)$ may not be positive semidefinite. Consider the counter-example provided in [26] where

$$X = \begin{bmatrix} 49 & 0 \\ 0 & 1 \end{bmatrix}, \text{ and } Y = \begin{bmatrix} 4 & 1 \\ 1 & 1 \end{bmatrix}.$$

Clearly X and Y are positive definite. However, the matrix

$$\tfrac{1}{2}(XY + YX) = \begin{bmatrix} 196 & 25 \\ 25 & 1 \end{bmatrix}$$

is clearly not positive semidefinite. In fact, a further example is provided in [29] of an SDP for which the AHO direction does not exist for a given positive definite iterate.

3.4.2 Choosing a suitable step length θ

Recall that our current iterate is $(X, Y) \in \mathcal{S}^p_{++} \times \mathcal{S}^p_{++}$. Once we have picked a step direction $(\Delta X, \Delta Y)$ from the directions discussed in Section 3.4.1, we need to compute a step length $\theta > 0$, where

$$\begin{aligned} \tilde{X} &:= X + \theta \Delta X, \\ \tilde{Y} &:= Y + \theta \Delta Y, \end{aligned}$$

and we require that $\tilde{X}, \tilde{Y} \succ \mathbf{0}$. The natural step length for a Newton direction is $\theta = 1$, but a step length of 1 may cause us to exceed the positive definite region to which we are constrained. Therefore, we need to determine the maximum possible step length $\theta_{\max}$ such that if $0 < \theta < \theta_{\max}$, then $\tilde{X}, \tilde{Y} \succ \mathbf{0}$.

The maximum step length in the direction ΔX can easily be seen as being the smallest positive number θ_1 such that

$$\det(X + \theta_1 \Delta X) = 0,$$

or $\theta_1 = \infty$ if no positive solutions exist. Similarly, the maximum step length in the direction ΔY is defined as the smallest positive number θ_2 such that

$$\det(Y + \theta_2 \Delta Y) = 0,$$

or $\theta_2 = \infty$ if no positive solutions exist. Our maximum possible step length is then the minimum of these two numbers:

$$\theta_{\max} = \min\{\theta_1, \theta_2\}. \tag{3.29}$$

Finally, since we do not want to take a step all the way to the boundary, we define our step length as

$$\theta := \min\{1, c \times \theta_{\max}\}, \tag{3.30}$$

for some $c \in (0, 1)$ which we will call the *step length parameter*; in practice, c is usually a fixed number chosen from the interval [0.9, 1.0). Defining θ in this way guarantees that our next iterate, $(\tilde{X}, \tilde{Y})$, satisfies $\tilde{X}, \tilde{Y} \succ \mathbf{0}$.

We can compute θ_1 and θ_2 as the minimum positive eigenvalues of the respective generalized eigenvalue problems

$$Xv = \lambda(-\Delta X)v, \quad \text{and} \quad Yv = \lambda(-\Delta Y)v.$$

Since algorithms for solving generalized eigenvalue problems prefer to have the matrix on the right-hand-side be positive definite (see [17]), it is better computationally to find the maximum eigenvalues λ_1 and λ_2 of the respective problems

$$-\Delta X v = \lambda X v, \quad \text{and} \quad -\Delta Y v = \lambda Y v,$$

then let

$$\theta_i = \begin{cases} 1/\lambda_i, & \text{if } \lambda_i > 0, \\ +\infty, & \text{otherwise,} \end{cases} \tag{3.31}$$

for $i = 1, 2$. See Lemma 5.1 from [13].

Algorithm 3.2

Given initial matrices $X_0, Y_0 \succ \mathbf{0}$, and tolerances $TOL_1, TOL_2 \geq 0$.

Let $(X, Y) := (X_0, Y_0)$ and $r = 0$.

While $\|\mathcal{M}X + Q - Y\|_F^2 > TOL_1$ **or** $\langle X, Y \rangle > TOL_2$, **do**

1. Let $\hat{\mu} := \langle X, Y \rangle / p$ and choose a centering parameter $\sigma \in [0, 1]$.
2. Choose a nonsingular matrix $P \in \mathbb{R}^{p \times p}$ and compute the corresponding search direction $(\Delta X, \Delta Y) \in \mathcal{S}^p \times \mathcal{S}^p$ by solving (3.21) with $\tau := \sigma\hat{\mu}$.
3. Choose a step length parameter $c \in (0, 1)$ and compute the step length θ by equation (3.30) so that
$$(\tilde{X}, \tilde{Y}) := (X, Y) + \theta(\Delta X, \Delta Y)$$
satisfies $\tilde{X}, \tilde{Y} \succ \mathbf{0}$.
4. Update $(X, Y) := (\tilde{X}, \tilde{Y})$.
5. Let $r := r + 1$, and $(X_r, Y_r) := (X, Y)$.

End

Figure 3.2: A general path-following algorithm for the SDLCP based on the MZ-family of search directions.

3.4.3 Summary

In summary, we describe a general path-following algorithm and a general predictor-corrector algorithm for solving the SDLCP (2.28) with $\mathcal{L}$ defined as in (3.13). The general path-following algorithm is given in Figure 3.2 while Figure 3.3 contains the general outline used for many predictor-corrector algorithms. Note that when $TOL_2 = 0$, these algorithms will each consistently generate an infinite sequence $\{(X_r, Y_r)\}$ in $\mathcal{S}^p_{++} \times \mathcal{S}^p_{++}$. Now we would like to know when $\{(X_r, Y_r)\}$ converges to the unique solution of the SDLCP, and, given $\varepsilon > 0$, how many iterations will it take for Algorithm 3.2 and Algorithm 3.3 to terminate when we take the tolerances to be

$$TOL_1 = \varepsilon\|\mathcal{M}X_0 + Q - Y_0\|_F^2 \quad \text{and} \quad TOL_2 = \varepsilon \langle X_0, Y_0 \rangle .$$

We call such an iterate (X_r, Y_r) which satisfies

$$\left.\begin{array}{rcl} \|\mathcal{M}X_r + Q - Y_r\|_F^2 & \leq & \varepsilon\|\mathcal{M}X_0 + Q - Y_0\|_F^2 \\ \langle X_r, Y_r \rangle & \leq & \varepsilon \langle X_0, Y_0 \rangle \end{array}\right\} \tag{3.32}$$

an *ε-approximate solution* of the SDLCP. The next section deals with the questions mentioned above.

Algorithm 3.3

Given initial matrices $X_0, Y_0 \succ \mathbf{0}$, and tolerances $TOL_1, TOL_2 \geq 0$.

Let $(X, Y) := (X_0, Y_0)$ and $r = 0$.

While $\|\mathcal{M}X + Q - Y\|_F^2 > TOL_1$ **or** $\langle X, Y \rangle > TOL_2$, **do**

1. Let $\hat{\mu} := \langle X, Y \rangle / p$ and choose a nonsingular $P \in \mathbb{R}^{p \times p}$.
2. (*Predictor step*)
   Compute the predictor direction $(\delta X, \delta Y) \in \mathcal{S}^p \times \mathcal{S}^p$ by solving (3.26).
3. Choose a centering parameter $\sigma \in (0, 1]$.
4. (*Corrector step*)
   Compute the predictor-corrector direction $(\Delta X, \Delta Y)$ by solving (3.28) with $\tau := \sigma\hat{\mu}$.
5. Choose a step length parameter $c \in (0, 1)$ and compute the step length θ by equation (3.30) so that
   $$(\tilde{X}, \tilde{Y}) := (X, Y) + \theta(\Delta X, \Delta Y)$$
   satisfies $\tilde{X}, \tilde{Y} \succ \mathbf{0}$.
6. Update $(X, Y) := (\tilde{X}, \tilde{Y})$.
7. Let $r := r + 1$, and $(X_r, Y_r) := (X, Y)$.

End

Figure 3.3: A general predictor-corrector algorithm for the SDLCP based on the MZ-family of search directions.

3.5 Convergence Results

A number of algorithms of theoretical importance have been formulated for solving the SDLCP (2.28), many of which fit nicely into the framework of either Algorithm 3.2 or Algorithm 3.3. Their importance comes from the fact that, in each case, it has been shown that the sequence of iterates converges to a solution of the SDLCP, and that the number of iterations required to get an ε-approximate solution is a polynomial function in both $1/\varepsilon$ and in the size of the problem, p. An algorithm which has this polynomial iteration complexity is said to be a *fully polynomial-time approximation scheme*, and any constant-factor decrease in ε, or increase in p, only results in a corresponding constant-factor increase in the running time of the algorithm (see [8, Ch. 37]). Also see [31, Ch. 3] for a complete discussion, in the case of linear programming, of why we need only be concerned with bounding the number of iterations by a polynomial in order to conclude that the algorithm itself has a polynomial running time.

These interior-point path-following algorithms for solving the SDLCP come under many different classifications:

- feasible / infeasible,
- short-step / long-step / predictor-corrector,
- AHO / HKM / dual-HKM / NT search directions.

Recall that the difference between feasible and infeasible algorithms is whether we are assuming that the initial matrices satisfy $(X_0, Y_0) \in \mathcal{L}$ or not. The difference between short-step, and long-step algorithms comes down to a difference in the strategies used for choosing the centering parameter σ and the step length parameter c in Algorithm 3.2, while predictor-corrector algorithms are based on the framework of Algorithm 3.3. Finally, we classify each algorithm according to which search direction it uses.

The key issue with any interior-point path-following algorithm is that the Newton steps we take toward the central path are not guaranteed to land us directly on the central path; the most we can hope for is that we are "near" the central path in each iteration. This idea of nearness is made concrete with the notion of neighbourhoods of the central path. First notice that we can state the central path as the set

$$\mathcal{C} = \left\{ (X,Y) \in \mathcal{L}_{++} \;\middle|\; XY = \hat{\mu} I, \text{ where } \hat{\mu} = \tfrac{\langle X,Y \rangle}{p} \right\}, \tag{3.33}$$

where we are assuming that $\mathcal{L}$ is defined as in Section 3.4. The two most common feasible neighbourhoods considered are the *narrow neighbourhood*,

$$\mathcal{N}_F(\gamma) := \left\{ (X,Y) \in \mathcal{L}_{++} \;\middle|\; \|\sqrt{X}Y\sqrt{X} - \hat{\mu} I\|_F \leq \gamma \hat{\mu}, \text{ where } \hat{\mu} = \tfrac{\langle X,Y \rangle}{p} \right\}, \tag{3.34}$$

and the *wide neighbourhood*,

$$\mathcal{N}_{-\infty}(\gamma) := \left\{ (X,Y) \in \mathcal{L}_{++} \;\middle|\; \lambda_{\min}(XY) \geq (1-\gamma)\hat{\mu}, \text{ where } \hat{\mu} = \tfrac{\langle X,Y \rangle}{p} \right\}, \tag{3.35}$$

where $\lambda_{\min}(XY)$ is the minimum eigenvalue of XY, and $\gamma \in (0,1)$ is a parameter which determines the width of the neighbourhoods. Clearly, we can see that $\mathcal{C} \subset \mathcal{N}_F(\gamma)$ and $\mathcal{C} \subset$

$\mathcal{N}_{-\infty}(\gamma)$ for all $\gamma \in (0,1)$. We can also consider the *infeasible central path*, which is defined as

$$\mathcal{SC} = \left\{ (X,Y) \in \mathcal{S}^p_{++} \times \mathcal{S}^p_{++} \;\middle|\; XY = \hat{\mu} I, \text{ where } \hat{\mu} = \tfrac{\langle X,Y \rangle}{p} \right\}. \tag{3.36}$$

The corresponding infeasible neighbourhoods are then

$$\mathcal{SN}_F(\gamma) := \left\{ (X,Y) \in \mathcal{S}^p_{++} \times \mathcal{S}^p_{++} \;\middle|\; \|\sqrt{X}Y\sqrt{X} - \hat{\mu} I\|_F \leq \gamma \hat{\mu}, \text{ where } \hat{\mu} = \tfrac{\langle X,Y \rangle}{p} \right\}, \tag{3.37}$$

and

$$\mathcal{SN}_{-\infty}(\gamma) := \left\{ (X,Y) \in \mathcal{S}^p_{++} \times \mathcal{S}^p_{++} \;\middle|\; \lambda_{\min}(XY) \geq (1-\gamma)\hat{\mu}, \text{ where } \hat{\mu} = \tfrac{\langle X,Y \rangle}{p} \right\}, \tag{3.38}$$

where $\gamma \in (0,1)$.

Now that we have a way of measuring nearness to the central path, we would like to determine strategies for choosing σ and c so that each iterate is within some neighbourhood, either narrow or wide, of the central path. *Short-step* algorithms are those which keep all iterates within a (feasible or infeasible) narrow neighbourhood, while those algorithms which keep all iterates within a (feasible or infeasible) wide neighbourhood are called *long-step*. The predictor-corrector type algorithms use two narrow neighbourhoods of different sizes, and keep all the predictor iterates in the larger neighbourhood, and all the corrector iterates in the smaller neighbourhood.

Kojima, Shindoh, and Hara proposed the first algorithm for solving the SDLCP in [13]. It is a feasible short-step algorithm using the dual-HKM direction (3.23) (see [13, p. 115] and [26, p. 391]) and keeps all iterates in the neighbourhood $\mathcal{N}_F(\gamma)$, where γ is chosen from the interval (0, 0.1]. Beginning at a point $(X_0, Y_0) \in \mathcal{N}_F(\gamma)$, they choose $\sigma = 1 - \gamma/\sqrt{p}$ in each iteration, and are able to take a full step, $\theta = 1$, while remaining in the neighbourhood of the central path. We now state the theorem from that paper which provides this result.

Theorem 3.5.1 ([13, Thm 8.1]) *Let $\gamma \in (0, 0.1]$. Suppose that $(X,Y) \in \mathcal{N}_F(\gamma)$. In Algorithm 3.2, let $\sigma = 1 - \gamma/\sqrt{p}$, $(\Delta X, \Delta Y)$ be the dual-HKM direction (3.23), and $\theta = 1$. Then the next iterate, $(\tilde{X}, \tilde{Y})$, satisfies*

$$(\tilde{X}, \tilde{Y}) = (X,Y) + (\Delta X, \Delta Y) \in \mathcal{N}_F(\gamma), \quad \textit{and}$$

$$\sigma\hat{\mu} \leq \tilde{\mu} \leq \left(1 - \frac{\gamma}{2\sqrt{p}}\right)\hat{\mu}, \tag{3.39}$$

where $\tilde{\mu} = \langle \tilde{X}, \tilde{Y} \rangle / p$.

What this tells us is that we are guaranteed to reduce the duality gap of our current iterate by at least a constant factor in each iteration, meaning that the duality gap of our iterates will converge to zero at a *linear rate*. Moreover, it follows that, under the conditions of Theorem 3.5.1, this algorithm is globally convergent in the sense that, given any $(X_0, Y_0) \in \mathcal{N}_F(\gamma)$, the infinite sequence $\{(X_r, Y_r)\}$ generated by Algorithm 3.2 when $TOL_2 = 0$ will always

converge to the unique optimal solution $(\bar{X}, \bar{Y})$. Following the commentary that appears after Theorem 8.1 in [13], we let $\varepsilon > 0$, and find that for every $r = 0, 1, 2, \ldots$ we have

$$(X_r, Y_r) \in \mathcal{N}_F(\gamma) \quad \text{and} \quad \langle X_r, Y_r \rangle \leq \left(1 - \frac{\gamma}{2\sqrt{p}}\right)^r \langle X_0, Y_0 \rangle .$$

To determine the number of iterations required in order to compute an ε-approximate solution which satisfies

$$(X_r, Y_r) \in \mathcal{L}_{++} \text{ and } \langle X_r, Y_r \rangle \leq \varepsilon \langle X_0, Y_0 \rangle ,$$

we follow the proof of a general complexity theorem in [31, p. 61–62]. Letting

$$r \geq \frac{2\sqrt{p}}{\gamma} \log \frac{1}{\varepsilon},$$

we have

$$\begin{aligned} \log \langle X_r, Y_r \rangle &\leq r \log \left(1 - \frac{\gamma}{2\sqrt{p}}\right) + \log \langle X_0, Y_0 \rangle \\ &\leq r \left(\frac{-\gamma}{2\sqrt{p}}\right) + \log \langle X_0, Y_0 \rangle \qquad \left(\text{since } \frac{-\gamma}{2\sqrt{p}} > -1\right) \\ &\leq \log \varepsilon + \log \langle X_0, Y_0 \rangle , \end{aligned}$$

which implies that $\langle X_r, Y_r \rangle \leq \varepsilon \langle X_0, Y_0 \rangle$. Therefore, we are able to compute an ε-approximate solution in $\mathcal{O}\left(\sqrt{p} \log \frac{1}{\varepsilon}\right)$ iterations, which implies that this feasible short-step algorithm of Kojima, Shindoh, and Hara is a fully polynomial-time approximation scheme.

Of course, as is stated in the concluding remarks of [13], this feasible short-step algorithm is mainly of theoretical importance. It requires that we prepare an initial point $(X_0, Y_0) \in \mathcal{N}_F(\gamma)$ with $\gamma = 0.1$, and even if we know such an initial point, it would be more computationally efficient to use a smaller centering parameter σ and a larger step length θ. Indeed, if $p = 100$, $\gamma = 0.1$, $\varepsilon = 10^{-10}$, and $TOL_2 = \varepsilon \langle X_0, Y_0 \rangle$, the above complexity result only guarantees that our algorithm will terminate in 2000 iterations, which is clearly unreasonable in practice.

Other important interior-point algorithms designed for solving the SDLCP were developed by Monteiro and Tsuchiya [19], and Kojima, Shida, and Shindoh [12]. In [19] the authors put forth two feasible algorithms, a short-step algorithm and a predictor-corrector algorithm, both based on an entire family of search directions called the KSH (Kojima-Shindo-Hara) family of directions, of which the dual-HKM direction is a member. Both of these algorithms are shown to be able to produce an ε-approximate solution of an SDLCP in at most $\mathcal{O}\left(\sqrt{p} \log \frac{1}{\varepsilon}\right)$ iterations. The algorithm considered in [12] is an infeasible predictor-corrector algorithm which uses the AHO direction. Using an infeasible wide neighbourhood similar to (3.38), but related to the equation $\frac{1}{2}(XY + YX) = \tau I$, the authors develop an algorithm with a sophisticated step length control rule that keeps all iterates within this neighbourhood. Furthermore, they prove that their algorithm is globally convergent, and, given $\varepsilon > 0$, must provide an ε-approximate solution in a finite number of iterations; under certain conditions, this number of iterations can be determined as being $\mathcal{O}\left(\frac{p^4}{\varepsilon} \log \frac{1}{\varepsilon}\right)$. However, the true strength in their algorithm lies in the fact that, under the assumption that the solution of the SDLCP satisfies the *strict complementarity* condition,

$$\bar{X} + \bar{Y} \succ \mathbf{0},$$

they are able to prove that a parameter which measures the optimality and feasibility of their iterates converges *quadratically* to zero.

This is a fast moving field and many aspects of interior-point algorithms for the SDLCP, and in particular for the SDP (2.22), continue to be studied. For example, many other search directions, and families of search directions, have been proposed which we have not mentioned here. Our goal has only been to obtain some theoretical convergence results for the interior-point methods we will be using to solve our three semidefinite constrained least squares problems. We now turn our attention toward the computational aspects of interior-point algorithms for our problems, while we keep in mind the following statement made by Todd in [27], where he observes that the intense study in the field of semidefinite optimization is partly due to "...great advances in our ability to solve such problems efficiently in theory and in practice (perhaps 'or' would be more appropriate: the most effective computational methods are not always provably efficient in theory, and *vice versa*)."

3.6 Implementation Issues

The purpose of this section is to discuss the implementation of both a standard path-following algorithm and a predictor-corrector algorithm for solving each of our three semidefinite constrained least squares problems. In each case, we have decided to use the AHO direction, but we will discuss the advantages of using the HKM, dual-HKM, and NT directions at the end of this section.

We use the following rule for computing the centering parameter σ in each iteration of our standard path-following algorithm, where (X, Y) is our current iterate.

$$\sigma := \begin{cases} \frac{1}{p^2}, & \text{if } \|Y - \mathcal{M}X - Q\|_F^2 < TOL_1 \\ 1 - \frac{1}{p^2}, & \text{otherwise} \end{cases} \tag{3.40}$$

The idea is to wait to reduce the duality gap of our iterates by choosing σ near 1 until we are within our desired tolerance of being feasible, after which we choose σ close to 0 to make rapid progress toward the solution.

Based the results reported in [2] and [29], we have decided to use the Mehrotra algorithm for our predictor-corrector algorithm. This algorithm is based on Algorithm 3.3 with an adaptive formula for determining the centering parameter σ in Step 3. First we must choose a step length parameter $c \in (0, 1)$ and compute the step length θ by equation (3.30) so that the predictor iterate,

$$(X^+, Y^+) = (X, Y) + \theta(\delta X, \delta Y),$$

satisfies $X^+, Y^+ \succ \mathbf{0}$. Letting $\hat{\mu}^+ = \langle X^+, Y^+ \rangle / p$, the Mehrotra rule for defining σ is

$$\sigma := \left(\frac{\hat{\mu}^+}{\hat{\mu}} \right)^3. \tag{3.41}$$

As is pointed out in [21] with regards to Mehrotra's algorithm for linear programming, if a lot of progress was made during the predictor step in reducing the duality gap of the current iterate, then $\hat{\mu}^+ \ll \hat{\mu}$, which will cause $\sigma \ll 1$. Having σ small will then allow for much progress to

be made during the corrector step as well. On the other hand, if not much progress is made during the predictor step, and $\hat{\mu}^+ \approx \hat{\mu}$, then we will have $\sigma \approx 1$, which will cause the corrector step to put more emphasis on centering the predictor iterate.

We have chosen to implement the step length parameter c according to a recommendation of Todd et al in [29]. They observe that letting $c = 0.98$ in each iteration r is usually a good choice, but that choosing c adaptively according to

$$c := 0.9 + 0.09 \times \theta_{r-1}, \tag{3.42}$$

where θ_{r-1} was the step length used in the previous iteration ($\theta_0 = 1$), can alleviate stagnation problems that sometimes occur when using a fixed step length parameter.

For both the standard path-following algorithm and the predictor-corrector algorithm, most of the work of each iteration is in solving a linear system of the form (3.21) for the step direction. This amounts to rewriting the linear system in the matrix-vector form using vec and the Kronecker product, or svec and the symmetric Kronecker product (see Appendix A), and then solving this fairly large system by factoring the $2p^2 \times 2p^2$ coefficient matrix. However, we will see that there are ways to take advantage of the structure present in these linear systems and reduce the bulk of the work to that of factoring a $p^2 \times p^2$ matrix. Moreover, it may be possible to avoid using vec or svec by solving a Lyapunov system directly (see [27, p. 547]), but we have not pursued this issue. Computing the predictor-corrector direction amounts to having to solve two different linear systems which have the same coefficient matrix. By retaining the matrix factorization (for example, the LU or Cholesky factorizations) from the predictor step, we are able to compute the corrector step with much less effort.

For each of our three least squares problems, we will consider both the vec and svec versions of the linear system we must solve. In each case we will propose a method that can be used to solve the resulting linear system more efficiently; there may be more efficient ways to solve these structured systems, especially if the matrices involved are large and sparse, but we have not pursued this issue. For simplicity, we will only examine the linear systems we must solve in order to obtain the AHO and dual-HKM directions for each of our least squares problems. The HKM and NT directions are similar.

3.6.1 Computing search directions for the SDLS problem

Recall that when we stated the SDLS problem (2.16) as an SDLCP (2.28), we had defined $\mathcal{M} : \mathcal{S}^n \to \mathcal{S}^n$ and $Q \in \mathcal{S}^n$ by

$$\begin{aligned} \mathcal{M}X &= \tfrac{1}{2}(A^TAX + XA^TA), \quad \text{and} \\ Q &= -\tfrac{1}{2}(A^TB + B^TA), \end{aligned}$$

so that $\Lambda = \mathcal{M}X + Q$ is the same as

$$\begin{aligned} A^T(AX - B) &= Z, \\ \tfrac{1}{2}(Z + Z^T) &= \Lambda. \end{aligned}$$

Using vec and the Kronecker product, we can write both of the linear systems for the AHO direction (3.18) and for the dual-HKM direction (3.23) in the form

Direction	E	F	D
AHO	$\frac{1}{2}(I \otimes \Lambda + \Lambda \otimes I)$	$\frac{1}{2}(I \otimes X + X \otimes I)$	$\tau I - \frac{1}{2}(X\Lambda + \Lambda X)$
dual-HKM	$\frac{1}{2}(X^{-1} \otimes \Lambda + \Lambda \otimes X^{-1})$	$I \otimes I$	$\tau X^{-1} - \Lambda$

Table 3.1: E, F, and D in equation (3.43) for the AHO and dual-HKM directions.

Direction	E	F	D
AHO	$I \otimes_s \Lambda$	$I \otimes_s X$	$\tau I - \frac{1}{2}(X\Lambda + \Lambda X)$
dual-HKM	$X^{-1} \otimes_s \Lambda$	$I \otimes_s I$	$\tau X^{-1} - \Lambda$

Table 3.2: E, F, and D in equation (3.48) for the AHO and dual-HKM directions.

$$\begin{bmatrix} \frac{1}{2}(I \otimes A^T A + A^T A \otimes I) & -I \otimes I \\ E & F \end{bmatrix} \begin{bmatrix} \mathrm{vec}(\Delta X) \\ \mathrm{vec}(\Delta\Lambda) \end{bmatrix} = \begin{bmatrix} \mathrm{vec}\left(\Lambda - \frac{1}{2}(Z + Z^T)\right) \\ \mathrm{vec}(D) \end{bmatrix}, \tag{3.43}$$

where $Z = A^T(AX - B)$, and the matrices E, F, and D are as in Table 3.1. Notice that while the coefficient matrix in the linear system (3.43) is $2n^2 \times 2n^2$, we are able to solve this system efficiently by multiplying the first block row by F and adding it to the second block row, resulting in the $n^2 \times n^2$ system

$$M\,\mathrm{vec}(\Delta X) = d \tag{3.44}$$

where

$$M = \tfrac{1}{2}F(I \otimes A^T A + A^T A \otimes I) + E, \tag{3.45}$$

$$d = F\,\mathrm{vec}\left(\Lambda - \tfrac{1}{2}(Z + Z^T)\right) + \mathrm{vec}(D), \tag{3.46}$$

and then letting

$$\Delta\Lambda := \tfrac{1}{2}(A^T A \Delta X + \Delta X A^T A) + \tfrac{1}{2}(Z + Z^T) - \Lambda. \tag{3.47}$$

If we choose the dual-HKM direction which has $F = I \otimes I$, then the corresponding M in equation (3.45) will be an $n^2 \times n^2$ symmetric and positive definite matrix (see Appendix A) and we can solve (3.44) using the Cholesky factorization of M. However, the AHO direction yields a matrix M that is, in general, nonsymmetric; this implies that an LU factorization will be necessary to solve (3.44) in this case.

We could also use svec and the symmetric Kronecker product to write both of the linear systems, (3.18) and (3.23), in the form

$$\begin{bmatrix} I \otimes_s A^T A & -I \otimes_s I \\ E & F \end{bmatrix} \begin{bmatrix} \operatorname{svec}(\Delta X) \\ \operatorname{svec}(\Delta \Lambda) \end{bmatrix} = \begin{bmatrix} \operatorname{svec}\left(\Lambda - \frac{1}{2}(Z + Z^T)\right) \\ \operatorname{svec}(D) \end{bmatrix}, \tag{3.48}$$

where the matrices are defined in Table 3.2. This time we can solve system (3.48) by solving

$$M \operatorname{svec}(\Delta X) = d \tag{3.49}$$

where

$$\begin{aligned} M &= F(I \otimes_s A^T A) + E, & (3.50) \\ d &= F \operatorname{svec}\left(\Lambda - \tfrac{1}{2}(Z + Z^T)\right) + \operatorname{svec}(D), & (3.51) \end{aligned}$$

and then letting $\Delta\Lambda$ be defined as in (3.47). Again, if we choose the dual-HKM direction, then M will be an $\bar{n} \times \bar{n}$ symmetric positive definite matrix, where $\bar{n} := \frac{1}{2}n(n+1)$; otherwise, M will be nonsymmetric for the AHO direction.

3.6.2 Computing search directions for the NS-SDLS problem

When we stated the NS-SDLS problem (2.19) as an SDLCP (2.28), we had defined $\mathcal{M} : \mathcal{S}^n \to \mathcal{S}^n$ and $Q \in \mathcal{S}^n$ by

$$\begin{aligned} \mathcal{M}\Lambda &= \tfrac{1}{2}(G^{-1}\Lambda + \Lambda G^{-1}), \text{ and} \\ Q &= \tfrac{1}{2}(G^{-1}A^T B + B^T A G^{-1}), \end{aligned}$$

where $G = A^T A \in \mathcal{S}^n_{++}$. With $\mathcal{M}$ and Q defined in this way, the equation $S = \mathcal{M}\Lambda + Q$ is equivalent to

$$\begin{aligned} A^T(AX - B) &= \Lambda, \\ \tfrac{1}{2}(X + X^T) &= S. \end{aligned}$$

Furthermore, given $X \in \mathbb{R}^{n \times n}$, $\Lambda \in \mathcal{S}^n$, and letting $S := \frac{1}{2}(X + X^T)$, it is easy to see that

$$\mathcal{M}\Delta\Lambda - \Delta S = S - \mathcal{M}\Lambda - Q \tag{3.52}$$

if and only if

$$A^T A \Delta X - \Delta\Lambda = \Lambda - A^T(AX - B) \quad \text{and} \quad \Delta S = \tfrac{1}{2}(\Delta X + \Delta X^T). \tag{3.53}$$

Indeed, to show the forward implication, we simply let

$$\Delta X := G^{-1}[(\Lambda + \Delta\Lambda) + A^T B] - X;$$

the backward implication is straightforward. Therefore, we are justified in replacing equation (3.52) with the first equation in (3.53) when solving for the search direction $(\Delta\Lambda, \Delta S)$, so long as we define $\Delta S := \frac{1}{2}(\Delta X + \Delta X^T)$ after determining ΔX.

Given the current iterate (X, Λ), where $\frac{1}{2}(X + X^T), \Lambda \succ \mathbf{0}$, we first let $S := \frac{1}{2}(X + X^T)$. Then we solve either the matrix-vector equation defined using vec,

$$\begin{bmatrix} I \otimes A^T A & -I \otimes I \\ E & F \end{bmatrix} \begin{bmatrix} \operatorname{vec}(\Delta X) \\ \operatorname{vec}(\Delta \Lambda) \end{bmatrix} = \begin{bmatrix} \operatorname{vec}\left(\Lambda - A^T(AX - B)\right) \\ \operatorname{vec}(D) \end{bmatrix}, \tag{3.54}$$

Direction	E	F	D
AHO	$\frac{1}{2}(I \otimes \Lambda + \Lambda \otimes I)V$	$\frac{1}{2}(I \otimes S + S \otimes I)$	$\tau I - \frac{1}{2}(S\Lambda + \Lambda S)$
dual-HKM	$\frac{1}{2}(S^{-1} \otimes \Lambda + \Lambda \otimes S^{-1})V$	$I \otimes I$	$\tau S^{-1} - \Lambda$

Table 3.3: E, F, and D in equation (3.54) for the AHO and dual-HKM directions.

Direction	E	F	D
AHO	$(I \otimes_s \Lambda)U^T V$	$I \otimes_s S$	$\tau I - \frac{1}{2}(S\Lambda + \Lambda S)$
dual-HKM	$(S^{-1} \otimes_s \Lambda)U^T V$	$I \otimes_s I$	$\tau S^{-1} - \Lambda$

Table 3.4: E, F, and D in equation (3.55) for the AHO and dual-HKM directions.

or the matrix-vector equation defined using svec for $\Delta\Lambda \in \mathcal{S}^n$, while still using vec for $\Delta X \in \mathbb{R}^{n\times n}$,

$$\begin{bmatrix} I \otimes A^T A & -U \\ E & F \end{bmatrix} \begin{bmatrix} \text{vec}(\Delta X) \\ \text{svec}(\Delta\Lambda) \end{bmatrix} = \begin{bmatrix} \text{vec}\left(\Lambda - A^T(AX - B)\right) \\ \text{svec}(D) \end{bmatrix}, \tag{3.55}$$

where U is defined as the matrix in equation (A.2). In the linear system (3.54), we have the matrices E, F, and D defined as in Table 3.3, while E, F, and D are defined as in Table 3.4 for the linear system (3.55). Note that V is the $n^2 \times n^2$ matrix which satisfies

$$V\,\text{vec}(\Delta X) = \text{vec}\left(\tfrac{1}{2}(\Delta X + \Delta X^T)\right);$$

thus $U^T V\,\text{vec}(\Delta X) = \text{svec}(\Delta S)$.

In order to solve (3.54) more efficiently, we can solve

$$M\,\text{vec}(\Delta X) = d, \tag{3.56}$$

where

$$M = F(I \otimes A^T A) + E, \tag{3.57}$$

$$d = F\,\text{vec}\left(\Lambda - A^T(AX - B)\right) + \text{vec}(D), \tag{3.58}$$

and let

$$\Delta\Lambda := A^T A \Delta X + A^T(AX - B) - \Lambda. \tag{3.59}$$

Alternately, we could solve (3.55) by solving system (3.56) with

$$M = I \otimes A^T A + UF^{-1}E, \tag{3.60}$$

$$d = \text{vec}\left(\Lambda - A^T(AX - B)\right) + UF^{-1}\text{svec}(D), \tag{3.61}$$

and then let $\Delta\Lambda$ be defined as in (3.59). Notice that M is an $n^2 \times n^2$ matrix in both (3.57) and (3.60), and so no advantage is obtained by solving (3.55) rather than (3.54). Furthermore, the dual-HKM direction does not appear to give a symmetric positive definite M in either (3.57) or (3.60), which means that we will have to resort to using the LU decomposition for both the AHO and dual-HKM directions.

3.6.3 Computing search directions for the LMI-LS problem

We stated the LMI-LS problem (2.21) as an SDLCP (2.28) by defining $\mathcal{M} : \mathcal{S}^k \to \mathcal{S}^k$ and $Q \in \mathcal{S}^k$ as

$$\begin{aligned} \mathcal{M}\Lambda &= \mathcal{K}G^{-1}\mathcal{K}^*\Lambda, \text{ and} \\ Q &= C - \mathcal{K}G^{-1}A^Tb, \end{aligned}$$

where $G = A^TA \in \mathcal{S}^k_{++}$ and $\mathcal{K} : \mathbb{R}^n \to \mathcal{S}^k$ and $\mathcal{K}^* : \mathcal{S}^k \to \mathbb{R}^n$ are defined by

$$\mathcal{K}x = \sum_{i=1}^n x_iK_i, \quad \mathcal{K}^*\Lambda = \begin{bmatrix} \langle K_1, \Lambda \rangle \\ \vdots \\ \langle K_n, \Lambda \rangle \end{bmatrix},$$

for some matrices $K_1, \ldots, K_n \in \mathcal{S}^k$. In this way, we have that $S = \mathcal{M}\Lambda + Q$ is equivalent to

$$\begin{aligned} A^T(Ax - b) + \mathcal{K}^*\Lambda &= \mathbf{0}, \\ \mathcal{K}x + S &= C. \end{aligned}$$

Also, as before, if we are given $x \in \mathbb{R}^n$ and $\Lambda, S \in \mathcal{S}^k$, we have that

$$\mathcal{M}\Delta\Lambda - \Delta S = S - \mathcal{M}\Lambda - Q$$

if and only if

$$\begin{aligned} A^TA\Delta x + \mathcal{K}^*\Delta\Lambda &= -\mathcal{K}^*\Lambda - A^T(Ax - b), \\ \mathcal{K}\Delta x + \Delta S &= C - \mathcal{K}x - S. \end{aligned}$$

Again, this is easy to see; the forward implication follows by defining

$$\Delta x := G^{-1}[A^Tb - \mathcal{K}^*(\Lambda + \Delta\Lambda)] - x,$$

while the backward implication is immediate.

Therefore, in order to solve for the search direction given the current iterate (x, Λ, S), where $\Lambda, S \succ \mathbf{0}$, we can solve either the linear system defined with vec,

$$\begin{bmatrix} A^TA & K^T & \mathbf{0} \\ K & \mathbf{0} & I \otimes I \\ \mathbf{0} & E & F \end{bmatrix} \begin{bmatrix} \Delta x \\ \mathrm{vec}(\Delta\Lambda) \\ \mathrm{vec}(\Delta S) \end{bmatrix} = \begin{bmatrix} -\mathcal{K}^*\Lambda - A^T(Ax - b) \\ \mathrm{vec}(C - \mathcal{K}x - S) \\ \mathrm{vec}(D) \end{bmatrix}, \tag{3.62}$$

where E, F, and D are as in Table 3.5, or we can solve the linear system defined with svec,

Direction	E	F	D
AHO	$\frac{1}{2}(I \otimes S + S \otimes I)$	$\frac{1}{2}(I \otimes \Lambda + \Lambda \otimes I)$	$\tau I - \frac{1}{2}(\Lambda S + S\Lambda)$
dual-HKM	$\frac{1}{2}(\Lambda^{-1} \otimes S + S \otimes \Lambda^{-1})$	$I \otimes I$	$\tau\Lambda^{-1} - S$

Table 3.5: E, F, and D in equation (3.62) for the AHO and dual-HKM directions.

Direction	E	F	D
AHO	$I \otimes_s S$	$I \otimes_s \Lambda$	$\tau I - \frac{1}{2}(\Lambda S + S\Lambda)$
dual-HKM	$\Lambda^{-1} \otimes_s S$	$I \otimes_s I$	$\tau\Lambda^{-1} - S$

Table 3.6: E, F, and D in equation (3.63) for the AHO and dual-HKM directions.

$$\begin{bmatrix} A^TA & K^T & \mathbf{0} \\ K & \mathbf{0} & I \otimes_s I \\ \mathbf{0} & E & F \end{bmatrix} \begin{bmatrix} \Delta x \\ \operatorname{svec}(\Delta\Lambda) \\ \operatorname{svec}(\Delta S) \end{bmatrix} = \begin{bmatrix} -\mathcal{K}^*\Lambda - A^T(Ax-b) \\ \operatorname{svec}(C - \mathcal{K}x - S) \\ \operatorname{svec}(D) \end{bmatrix}, \tag{3.63}$$

with E, F, and D as in Table 3.6. Furthermore, the K in equation (3.62) is defined as

$$K = [\operatorname{vec}(K_1) \cdots \operatorname{vec}(K_n)] \tag{3.64}$$

so that

$$\operatorname{vec}(\mathcal{K}\Delta x) = K\Delta x, \quad \text{and} \quad \mathcal{K}^*\Delta\Lambda = K^T \operatorname{vec}(\Delta\Lambda),$$

while the K in equation (3.63) is defined as

$$K = [\operatorname{svec}(K_1) \cdots \operatorname{svec}(K_n)] \tag{3.65}$$

so that

$$\operatorname{svec}(\mathcal{K}\Delta x) = K\Delta x, \quad \text{and} \quad \mathcal{K}^*\Delta\Lambda = K^T \operatorname{svec}(\Delta\Lambda).$$

Following [21, p. 408], we can simplify the linear system (3.62) by eliminating ΔS, producing the equivalent system

$$\begin{aligned} \begin{bmatrix} A^TA & K^T \\ K & -F^{-1}E \end{bmatrix} \begin{bmatrix} \Delta x \\ \operatorname{vec}(\Delta\Lambda) \end{bmatrix} &= \begin{bmatrix} r_d \\ r_p - F^{-1}\operatorname{vec}(D) \end{bmatrix}, \\ \operatorname{vec}(\Delta S) &= r_p - K\Delta x, \end{aligned}$$

where

$$\begin{aligned} r_d &= -\mathcal{K}^*\Lambda - A^T(Ax-b), \\ r_p &= \operatorname{vec}(C - \mathcal{K}x - S). \end{aligned}$$

We can further reduce this system by eliminating $\Delta\Lambda$, which gives us

$$(A^TA + K^TE^{-1}FK)\Delta x = r_d + K^TE^{-1}r_q, \tag{3.66}$$

$$\text{vec}(\Delta\Lambda) = E^{-1}(FK\Delta x - r_q), \tag{3.67}$$

$$\text{vec}(\Delta S) = r_p - K\Delta x, \tag{3.68}$$

where

$$r_q = Fr_p - \text{vec}(D).$$

We note here that these equations (3.66)-(3.68) are identical to those for SDP in [2] where they state that computing the step direction in this manner is more stable than other methods considered. A similar discussion also holds for the linear system (3.63).

At this point we could use the Cholesky factorization on the $n \times n$ matrix in equation (3.66), provided that $E^{-1}F$ is symmetric positive definite, and that K has full-column rank (i.e., the matrices $K_1, \ldots, K_n$ are linearly independent). For example, it is easy to see that the HKM direction would have $E^{-1}F$ symmetric positive definite, due to the fact that

$$E = I \otimes I, \quad \text{and} \quad F = \tfrac{1}{2}(S^{-1} \otimes \Lambda + \Lambda \otimes S^{-1}).$$

Of course, if the matrix in (3.66) is not symmetric, we can always use the LU decomposition to solve for Δx.

3.6.4 Differences in the search directions

We have already mentioned one significant difference among the different search directions we are considering in Theorem 3.4.1. This was the fact that it is possible that the AHO direction will not exist at certain interior points, although, as stated in [27, p. 547], this does not seem to cause difficulties in practice. On the other hand, by the same theorem, we also know that the dual-HKM and NT search directions always exist.

Although Theorem 3.4.1 would seem to imply that we would prefer to use the dual-HKM or NT directions over the AHO direction, this is not necessarily the case. As is noticed in both [2] and [29], performing numerical experiments with path-following algorithms for solving SDPs (2.22) using the AHO, HKM, and NT directions shows that the AHO direction gives rise to a method that

- can achieve higher accuracy in the duality gap, and
- converges in fewer iterations,

than the methods using the HKM and NT directions. We can see why the AHO direction might produce a more stable method for our problems from the fact that the linear system defining this direction is actually defined at the optimal solution, which is a feature that the HKM, dual-HKM, and NT directions do not possess. See also [2] for more stability analysis of the AHO direction.

However, as we also observed for the SDLS and LMI-LS problems, the main disadvantage of using the AHO direction for solving SDP problems is that each iteration of the method will be

about twice as expensive as each iteration of the method using either the HKM or NT directions [29, p. 790]. This is due to the fact that there is some symmetry to be taken advantage of when solving for the HKM and NT directions for an SDP problem; this symmetry does not exist when solving for the AHO direction. It turns out that, in terms of computer running time, using the HKM direction gives rise to the fastest method for reducing the duality gap by a factor of 10^{10} for SDPs (2.22) (see [29, p. 791]). This point is further reinforced by the fact that one of the more widely used codes for solving SDPs, SDPT^3 [28], has removed the option of using the AHO direction due to it being inefficient or under-used, while retaining the options of using the HKM and NT directions. Moreover, as was mentioned before, HKM is currently the direction of choice for large SDP problems [22].

All things considered, we have decided to use the AHO direction in our experiments, primarily due to the fact that we are mostly interested in solving only small scale problems. We also find the stability properties and the simplicity of implementing the AHO direction very appealing. Indeed, we see in [2] how care must be observed when implementing the HKM direction for SDP.

Chapter 4
Numerical Results

The purpose of this chapter is to obtain some information about how the standard path-following (SPF) algorithm (Figure 3.2) and the predictor-corrector (PC) algorithm (Figure 3.3) perform in practice on each of the SDLS (2.16), NS-SDLS (2.19), and LMI-LS (2.21) problems; this gives us a total of six algorithms for our numerical experiments. We will also compare Woodgate's Algorithm [34] to the PC algorithm for the SDLS problem.

All of our experiments were conducted in MATLAB [17]. A listing of the MATLAB M-files used to produce these results can be found in Appendix B. Before reporting any results, we will first explain some of the details of these codes and the methods of experimentation used.

4.1 Implementation and experimentation details

The implementation details that need to be discussed are:

- Which search direction are we using and how are we solving for it?
- How are we computing the maximum step length?
- What have we chosen for the starting points?
- What is the stopping criteria?
- How have we implemented Woodgate's Algorithm?
- How are we generating problem instances for our numerical experiments?

We will now discuss each of these questions in order.

4.1.1 Solving for the search direction

In each of our six MATLAB codes we have used the AHO direction (3.18). We have decided to solve for this direction using vec and the standard Kronecker product (Appendix A); this is due to the fact that there was too much overhead computation in our implementations using svec and the symmetric Kronecker product to give any advantage in solving a smaller linear system.

Since we are using vec to solve for the AHO direction, we use the equations (3.44–3.47) and (3.56–3.59) to solve for this direction for the SDLS and NS-SDLS problems, respectively.

However, we can improve on these equations as follows. Since we are using the AHO direction, we can write (3.45) and (3.46) as

$$M = \tfrac{1}{4}\left[(I \otimes XA^TA) + (A^TA \otimes X) + (X \otimes A^TA) + (XA^TA \otimes I)\right] + E, \quad (4.1)$$
$$d = \mathrm{vec}(\tau I - \tfrac{1}{2}(XZ + ZX)), \quad (4.2)$$

where $Z = A^T(AX - B)$. We can easily see that this is a more efficient way to compute M because the sum of four $n^2 \times n^2$ matrices is cheaper to compute than the product of two $n^2 \times n^2$ matrices. Similarly, when using the AHO direction, we can write (3.57) as

$$M = \tfrac{1}{2}\left[(I \otimes SA^TA) + (S \otimes A^TA)\right] + E, \quad (4.3)$$
$$d = \mathrm{vec}(\tau I - \tfrac{1}{2}(SR + RS + S\Lambda + \Lambda S)), \quad (4.4)$$

where $R = Z - \Lambda$ and $Z = A^T(AX - B)$. We do not simplify the expression for d in (4.4) any further due to a loss of stability that was noticed in numerical experiments. In fact, replacing (4.4) with

$$d = \mathrm{vec}(\tau I - \tfrac{1}{2}(SZ + ZS)),$$

results in a loss in feasibility of the iterates as the duality gap goes to zero. A similar phenomenon was also noticed for SDP in [2, p. 757]. For the LMI-LS problem, our implementation using equations (3.66–3.68) did not produce the efficiency we were hoping for; therefore, we chose to solve (3.62) directly for the AHO direction.

4.1.2 Computing the maximum step length

When solving the generalized eigenvalue problems in the computation of the maximum step, we have found that computing every eigenvalue and taking the maximum one is often less expensive than using an iterative method for just computing the maximum eigenvalue. In our limited experimentation with sparsity, we found that it is advantageous to use an iterative method when dealing with sparse matrices, but in the case of dense matrices, it was more efficient to just solve these generalized eigenvalue problems directly.

4.1.3 Starting points

The starting points we choose for all our experiments is as follows.

- SDLS / NS-SDLS: $X_0 = I$ and $\Lambda_0 = I$
- LMI-LS: $x_0 = \mathbf{0}$, $S_0 = I$, and $\Lambda_0 = I$.

For the SDLS and NS-SDLS problems, I is the $n \times n$ identity matrix; for the LMI-LS problem, $\mathbf{0}$ is the zero vector of length n and I is the $k \times k$ identity matrix.

4.1.4 Stopping criteria

We use the following stopping criteria:

(i) Stop when $\|\text{res}\| < TOL_1$ and $\hat{\mu} < TOL_2$.

Here we have the norm of the residual defined as

$$\|\text{res}\| := \begin{cases} \left\|\frac{1}{2}(Z+Z^T)-\Lambda\right\|_F, & \text{for SDLS,} \\ \|Z-\Lambda\|_F, & \text{for NS-SDLS,} \end{cases}$$

where

$$Z = A^T(AX-B).$$

Recall that the normalized duality gap is defined as

$$\hat{\mu} := \begin{cases} \langle X,\Lambda\rangle/n, & \text{for SDLS,} \\ \langle S,\Lambda\rangle/n, & \text{for NS-SDLS,} \end{cases}$$

where

$$S = \tfrac{1}{2}(X+X^T).$$

For the LMI-LS problem, the norm of the residual and the normalized duality gap are defined as

$$\begin{aligned} \|\text{res}\| &:= \left\| \begin{bmatrix} r_d \\ r_p \end{bmatrix} \right\|_2, \\ \hat{\mu} &:= \langle S,\Lambda\rangle/k, \end{aligned}$$

where

$$\begin{aligned} r_d &= -\mathcal{K}^*\Lambda - A^T(Ax-b), \\ r_p &= \text{vec}(C-\mathcal{K}x-S). \end{aligned}$$

Furthermore, we set

$$\begin{aligned} TOL_1 &:= \sqrt{\varepsilon}\|\text{res}_0\|, \\ TOL_2 &:= \varepsilon\hat{\mu}_0, \end{aligned}$$

where $\|\text{res}_0\|$ and $\hat{\mu}_0$ are the norm of the residual and the normalized duality gap at the initial iterate, and $\varepsilon > 0$ is a given tolerance. Notice that, given the starting points above, we will always have $\hat{\mu} = 1$, and so $TOL_2 = \varepsilon$. For the most part, we will be using $\varepsilon = 10^{-10}$ except when investigating the performance of Woodgate's Algorithm and the predictor-corrector algorithm on different tolerances.

(ii) Stop when we have exceeded the maximum number of iterations, $MaxIt$.

For all six algorithms, we have set

$$MaxIt = 100.$$

If we have reached our maximum number of iterations without meeting our desired tolerances, our algorithm will report a failure. Failed attempts will be reported but not included in our averages.

4.1.5 Woodgate's Algorithm

For the details of Woodgate's Algorithm, please refer to [34]. However, we would like to mention here the initial iterate and stopping conditions we will be using in our implementation of this algorithm.

We have decided to use the recommended starting point from [34], which is the matrix

$$X_0 = \alpha \hat{X}_{(0)},$$

where $\hat{X}$ is the solution to the symmetric unconstrained least squares problem (1.3), $\hat{X}_{(0)}$ is the result of the result of changing all of the negative eigenvalues of $\hat{X}$ to 0 in the eigenvalue decomposition of $\hat{X}$, and α is some nonnegative scalar.

Since a stopping criterion for approximate solutions was not provided in [34], we have decided to stop iterating Woodgate's Algorithm when $\|\text{res}\| < TOL_1$ and $\hat{\mu} < TOL_2$, or when we have exceeded the maximum number of iterations, $MaxIt$. This time, we have

$$\begin{aligned} \|\text{res}\| &= \|L(X)_{(0)} - L(X)\|_F, \\ \hat{\mu} &= \langle X, L(X)_{(0)} \rangle / n, \\ MaxIt &= 100, \end{aligned}$$

where

$$L(X) := A^T A X + X A^T A - A^T B - B^T A.$$

Moreover, we set the tolerances as

$$\begin{aligned} TOL_1 &:= \sqrt{\varepsilon}, \\ TOL_2 &:= \varepsilon. \end{aligned}$$

This was done in order to make a fair comparison with the predictor-corrector algorithm for the SDLS problem, given the fact that, in general, $\hat{\mu}_0 \neq 1$ for the initial iterate X_0 above.

One more difference should be noted between Woodgate's Algorithm from [34] and our implementation of this algorithm. In each iteration, it is necessary to solve a linear system which has an $n^2 \times n^2$ symmetric coefficient matrix, M, that is sometimes near singular. In order to avoid this near singularity, we modified M by adding νI, for some small positive parameter ν. We were surprised to find that this modification greatly improved the convergence of this algorithm! After some experimentation, we decided to use $\nu = 1/n^2$.

4.1.6 Generating problem instances

For all of our experiments, we have chosen to report each result averaged over 10 randomly generated instances of our problems. We randomly generate the matrices A and B for the SDLS and NS-SDLS problems, and the vector b and the matrices A, $K_1, \ldots, K_n$, and C for the LMI-LS problem by choosing their entries uniformly from the interval $[-1, 1]$. We have chosen the dimensions of our problems to satisfy $m > n$ and $n > \frac{1}{2}k(k+1)$ so that A has full column rank in all three problems, and $\text{span}\{K_1, \ldots, K_n\} = \mathcal{S}^k$ in the LMI-LS problem (see Corollary 2.4.10).

4.2 Numerical comparison of the algorithms

Our experimentation comes in three parts. First of all, we compare the SPF algorithm with the PC algorithm for each of our three semidefinite constrained least squares problems. We then compare the PC algorithm for solving the SDLS problem to Woodgate's Algorithm. Finally, we look at each of these algorithms in action on a typical instance of each problem.

4.2.1 The SPF / PC experiment

We compare the performance of the SPF algorithm with the PC algorithm for the SDLS, NS-SDLS, and LMI-LS problems. This is done by determining how the average number of iterations and amount of CPU time required to compute an ε-approximate solution, where $\varepsilon = 10^{-10}$, and the resulting average feasibility of this solution – measured by the norm of the residual of the solution – each depend on the size of the problem. These averages are based on 10 randomly generated problem instances for each problem size. The results of this experiment are shown in Table 4.1. We have plotted the results from Table 4.1 in Figure 4.1 for the SDLS problem, in Figure 4.2 for the NS-SDLS problem, and in Figure 4.3 for the LMI-LS problem.

Based on these results, we make the following observations.

1. The PC algorithm is the clear winner in each of the three problems when we consider the CPU time required to reduce the duality gap by a factor of 10^{10}. Even though each iteration of the PC algorithm is more expensive than each iteration of the SPF algorithm, we find that the overall CPU time for the PC algorithm is much less that of the SPF algorithm. This is due to the fact that, on average, the PC algorithm converges in far fewer iterations than the SPF algorithm.

2. On the other hand, we notice that the SPF algorithm is better suited to producing an approximate solution that is closer to being feasible. This is partly due to the strategy we have used for choosing the centering parameter σ (3.40) in the SPF algorithm. In fact, we find that while the PC algorithm tends to reduce the norm of the residual gradually, the SPF algorithm achieves feasibility once we are able to take a step length of $\theta = 1$.

3. We observe that the NS-SDLS algorithms seem to require much more CPU time than their corresponding SDLS algorithms. This difference can be explained by the need to multiplying an two $n^2 \times n^2$ matrices each iteration in order to compute the matrix $E = \frac{1}{2}(I \otimes \Lambda + \Lambda \otimes I)V$.

4. Although we were successful in approximating the solution the SDLS and NS-SDLS problems with a tolerance of 10^{-10} in each attempt (in fact, we were able to perform a Cholesky factorization on these approximations, proving that they were numerically positive definite), we encountered a few higher order problems for which the SPF algorithm for the LMI-LS problem stagnated (we exceeded our maximum number of iterations). However, as the results show, the PC algorithm for the LMI-LS problem was successful in each attempt.

5. The rapid growth in the CPU time required to solve these problems is to be expected due to the fact that in each iteration we are performing an LU decomposition of an $n^2 \times n^2$

(or $(n + 2k^2) \times (n + 2k^2)$ for the LMI-LS problem) matrix. The CPU time required for this LU decomposition is itself proportional to n^6 (or $(n+2k^2)^3$ for the LMI-LS problem).

4.2.2 The Woodgate / PC experiment

Now we would like to compare Woodgate's Algorithm to the PC algorithm for solving the SDLS problem. In order to do this, we have chosen to compare their ability to achieve a variety of tolerances on both small instances ($n = 5$) and large instances ($n = 30$). The results averaged over 10 random instances for $n = 5$ and over 10 random instances for $n = 30$ are shown in Table 4.2; the corresponding plots are presented in Figure 4.4.

From this experiment we reach the same conclusion as in [34] that Woodgate's Algorithm is able to converge very quickly to an approximate solution of the SDLS problem for small instances with limited tolerances. However, for larger problems and smaller tolerances, Woodgate's Algorithm is much slower to converge than the PC algorithm, both in terms of number of iterations and CPU time required. In fact, as we see for $n = 30$ and $\varepsilon \leq 10^{-4}$, Woodgate's Algorithm sometimes exceeds the maximum number of iterations, 100, before being able to achieve the desired tolerance.

4.2.3 A final comparison

The purpose of this section is to take a typical medium sized random instance of each of the SDLS, NS-SDLS, and LMI-LS problems, and investigate the convergence properties of each of the algorithms considered in this thesis. We can see how each algorithm converges to the optimal solution by watching how the duality gap goes to zero in the duality gap convergence curves shown in Figure 4.5.

These convergence curves again indicate the rapid convergence of the PC algorithm for each problem. At the same time we see how the SPF algorithm has a fairly level convergence curve while the centering parameter σ is still near 1, and then converges fairly rapidly for the SDLS and NS-SDLS problems when σ is chosen near 0. However, we also witness the ability for the SPF algorithm to stagnate slightly when solving an LMI-LS instance. Woodgate's Algorithm is seen to make slow but steady progress toward the solution.

4.3 Estimating the compliance matrix

We now return to the application that was mentioned in Chapter 1 about estimating the compliance matrix at a certain location on a deformable object. Given a number of displacement measurements, $u^1 \cdots u^l \in \mathbb{R}^3$, corresponding to some forces, $p^1 \cdots p^l \in \mathbb{R}^3$, our aim is to find a matrix $\bar{X}$ which is the least squares solution of the equation

$$[p^1 \cdots p^l]^T X^T = [u^1 \cdots u^l]^T \tag{4.5}$$

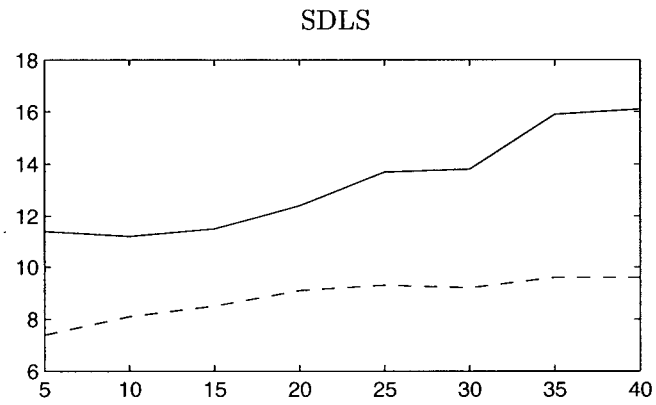
subject to the constraint that the real parts of the eigenvalues of X are nonnegative; i.e., that $\frac{1}{2}(X + X^T) \succeq \mathbf{0}$. Notice that we can write (4.5) as $AX^T = B$.

SDLS		# iterations		CPU time (sec)		$\log_{10}(\|res\|)$	
m	n	SPF	PC	SPF	PC	SPF	PC
20	5	11.4	7.4	0.02	0.02	-15.2	-10.6
40	10	11.2	8.1	0.06	0.05	-14.6	-10.8
60	15	11.5	8.5	0.39	0.31	-14.4	-10.1
80	20	12.4	9.1	1.38	1.05	-14.2	-10.1
100	25	13.7	9.3	4.06	2.84	-14.0	-10.7
120	30	13.8	9.2	9.61	6.57	-13.8	-10.3
140	35	15.9	9.6	23.85	14.70	-13.7	-10.4
160	40	16.1	9.6	47.95	28.95	-13.6	-10.1

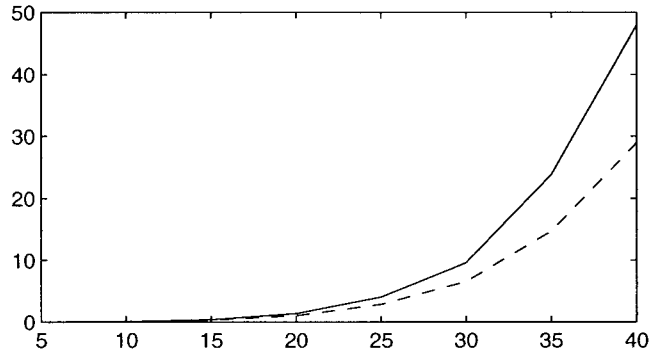
NS-SDLS		# iterations		CPU time (sec)		$\log_{10}(\|res\|)$	
m	n	SPF	PC	SPF	PC	SPF	PC
20	5	11.2	7.2	0.02	0.01	-15.0	-10.4
40	10	11.4	8.4	0.07	0.06	-12.5	-10.9
60	15	12.2	8.9	0.50	0.39	-12.2	-10.4
80	20	12.4	9.1	2.06	1.57	-11.0	-10.1
100	25	12.7	9.1	6.72	4.94	-11.0	-10.4
120	30	13.4	9.1	18.91	13.15	-10.7	-10.1
140	35	13.9	9.5	47.09	32.84	-9.8	-9.8
160	40	14.8	9.6	106.99	70.61	-9.0	-9.4

LMI-LS			% success		# iterations		CPU time (sec)		$\log_{10}(\|res\|)$	
m	n	k	SPF	PC	SPF	PC	SPF	PC	SPF	PC
40	20	5	100	100	11.3	7.7	0.03	0.02	-14.3	-10.3
120	60	10	100	100	13.5	8.3	0.30	0.20	-13.6	-10.2
300	150	15	100	100	17.3	8.3	2.97	1.45	-13.0	-9.5
500	250	20	100	100	26.3	8.7	19.38	6.54	-12.7	-9.8
700	350	25	90	100	32.1	8.9	75.85	20.96	-12.4	-9.6
1000	500	30	70	100	45.1	9.2	289.44	60.86	-12.2	-10.3

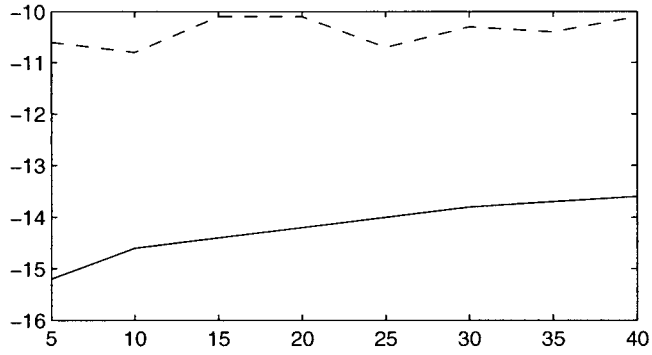
Table 4.1: Results from computing approximate solutions of the SDLS, NS-SDLS, and LMI-LS problems with $\hat{\mu} < 10^{-10}$, where $\hat{\mu}$ is the normalized duality gap. The SPF algorithm and the PC algorithm are used. Results averaged over 10 randomly generated problems for each problem size. Failures are not included in the averages.



(a) Average # iterations vs problem size n

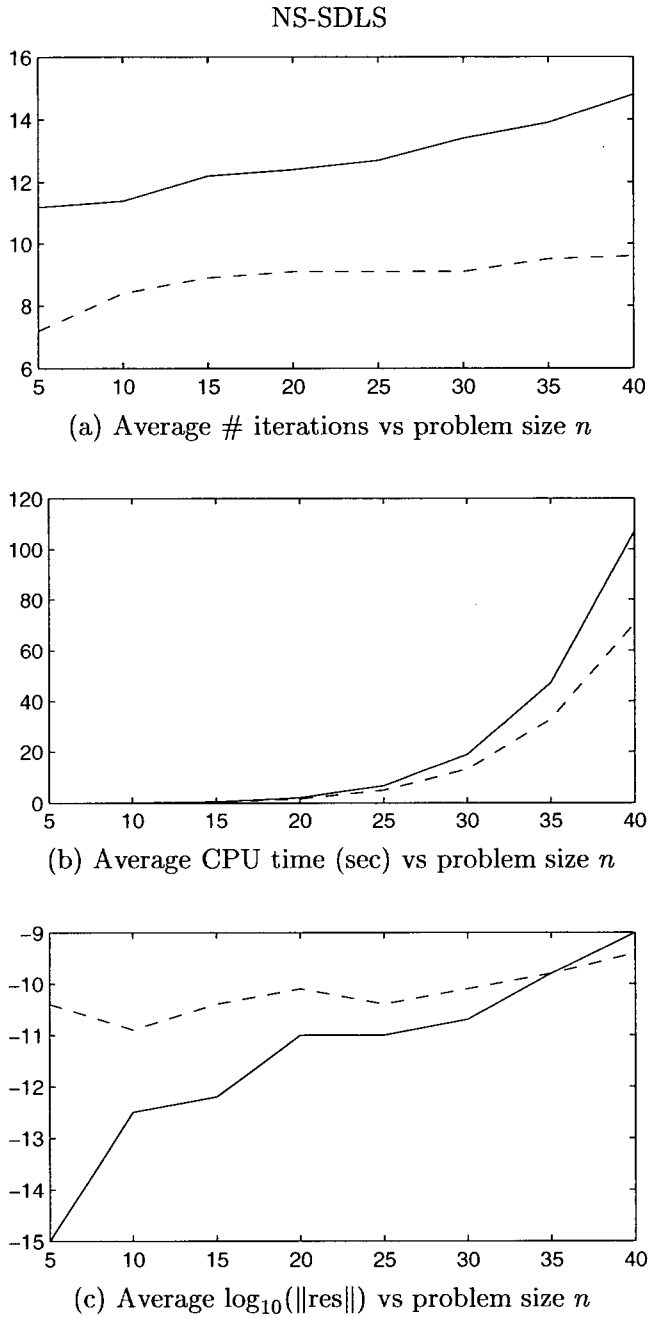


(b) Average CPU time (sec) vs problem size n



(c) Average $\log_{10}(\|\text{res}\|)$ vs problem size n

Figure 4.1: The plots of the data from Table 4.1 for the SDLS problem. The solid line corresponds to the SPF algorithm, while the dashed line corresponds to the PC algorithm.

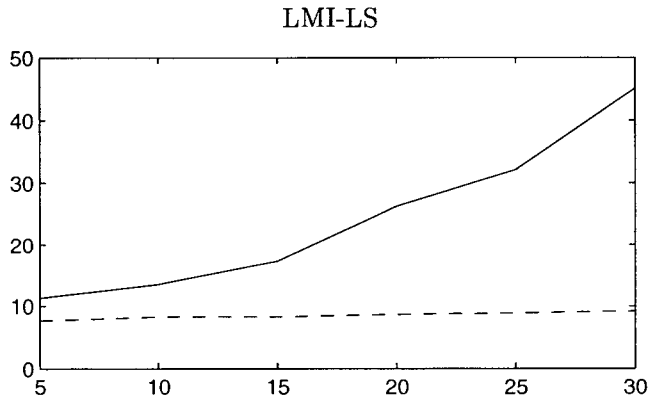


(a) Average # iterations vs problem size n

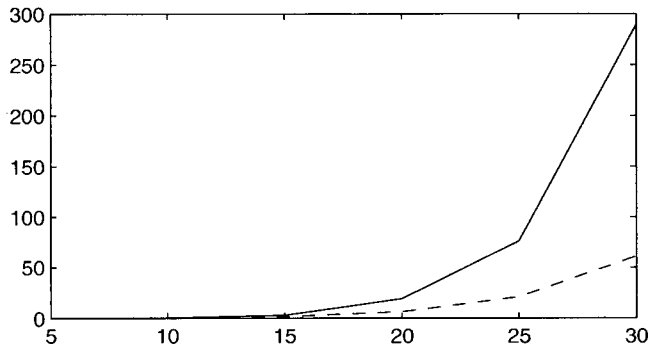
(b) Average CPU time (sec) vs problem size n

(c) Average $\log_{10}(\|\text{res}\|)$ vs problem size n

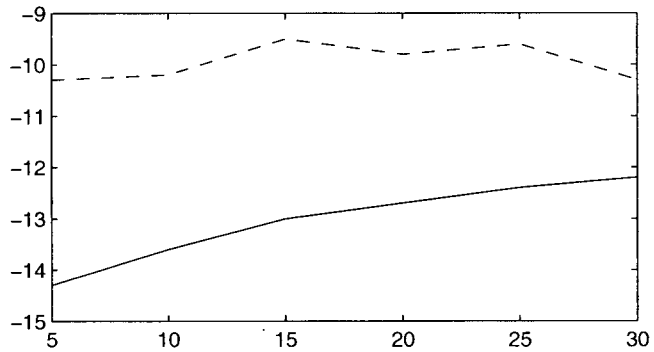
Figure 4.2: The plots of the data from Table 4.1 for the NS-SDLS problem. The solid line corresponds to the SPF algorithm, while the dashed line corresponds to the PC algorithm.



(a) Average # iterations vs problem size k



(b) Average CPU time (sec) vs problem size k



(c) Average $\log_{10}(\|\text{res}\|)$ vs problem size k

Figure 4.3: The plots of the data from Table 4.1 for the LMI-LS problem. The solid line corresponds to the SPF algorithm, while the dashed line corresponds to the PC algorithm.

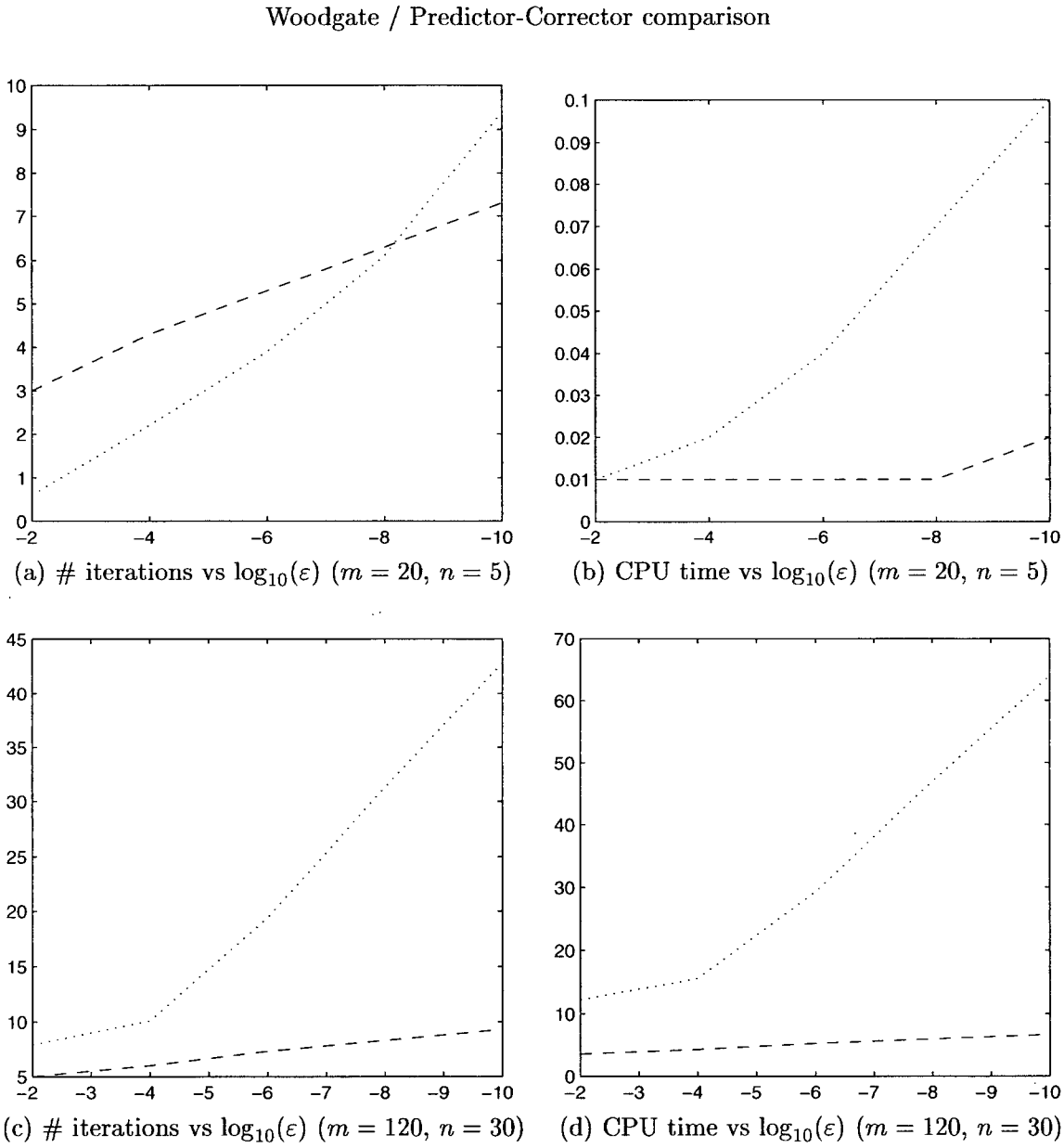
SDLS	% success		# iterations		CPU time (sec)	
$\log_{10}(\varepsilon)$	WG	PC	WG	PC	WG	PC
-2	100	100	0.6	3.0	0.01	0.01
-4	100	100	2.2	4.3	0.02	0.01
-6	100	100	3.9	5.3	0.04	0.01
-8	100	100	6.1	6.3	0.07	0.01
-10	100	100	9.4	7.3	0.10	0.02

(a) $m = 20$, $n = 5$

SDLS	% success		# iterations		CPU time (sec)	
$\log_{10}(\varepsilon)$	WG	PC	WG	PC	WG	PC
-2	100	100	7.9	5.0	12.21	3.57
-4	80	100	10.1	6.0	15.52	4.29
-6	80	100	19.4	7.3	29.27	5.22
-8	80	100	31.3	8.3	46.91	5.94
-10	80	100	42.8	9.3	64.02	6.66

(b) $m = 120$, $n = 30$

Table 4.2: Results from computing approximate solutions of the SDLS problem with normalized duality gap $\hat{\mu} < \varepsilon$, for $\varepsilon = 10^{-2}, \ldots, 10^{-10}$. Woodgate's Algorithm (WG) [34] and the PC algorithm are used. Results averaged over 10 randomly generated problems. Failures are not included in the averages.



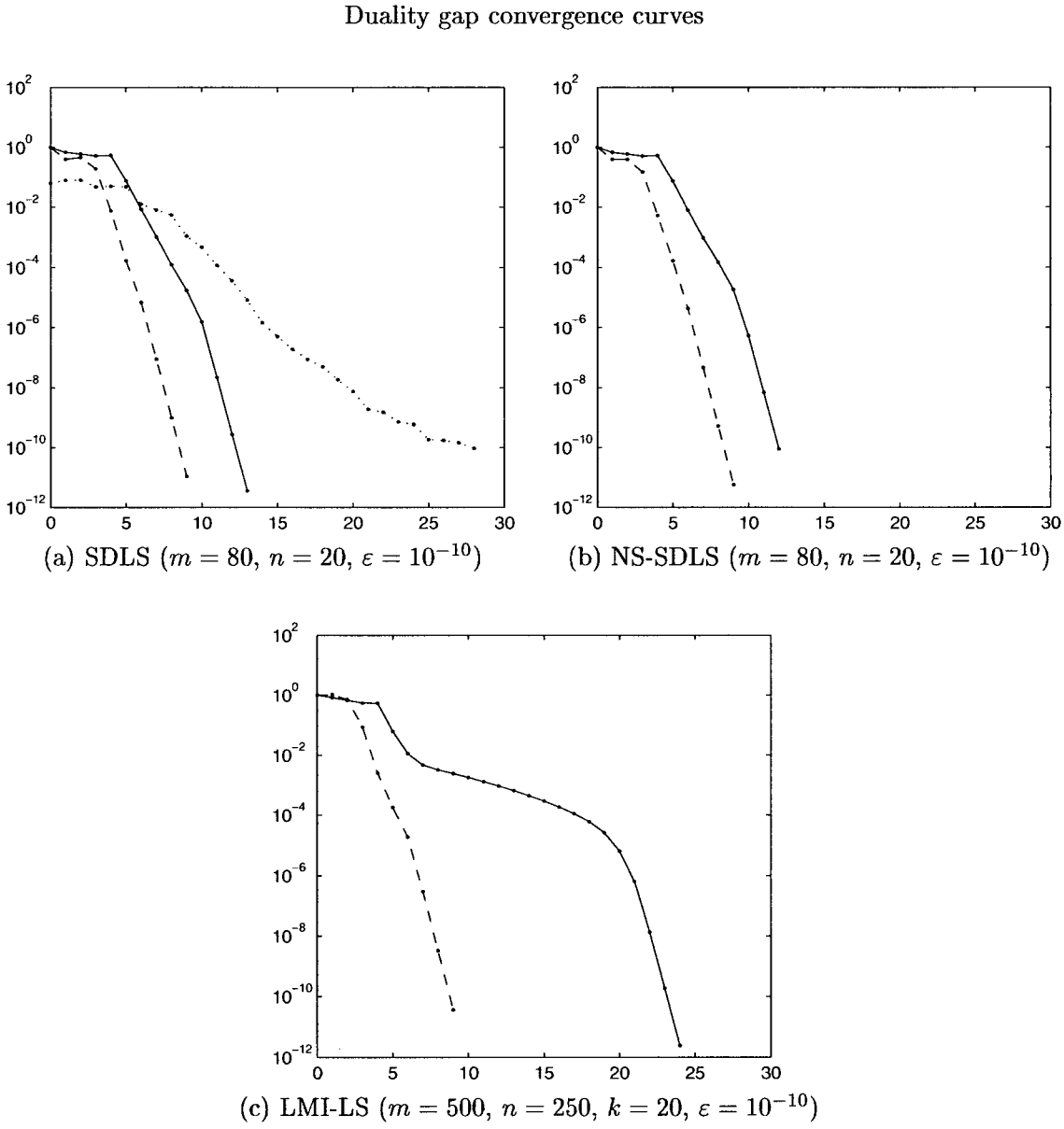
(a) # iterations vs $\log_{10}(\varepsilon)$ ($m = 20$, $n = 5$)

(b) CPU time vs $\log_{10}(\varepsilon)$ ($m = 20$, $n = 5$)

(c) # iterations vs $\log_{10}(\varepsilon)$ ($m = 120$, $n = 30$)

(d) CPU time vs $\log_{10}(\varepsilon)$ ($m = 120$, $n = 30$)

Figure 4.4: The plots of the data from Table 4.2. The dashed line corresponds to the PC algorithm, while the dotted line corresponds to Woodgate's Algorithm [34].



(a) SDLS ($m = 80$, $n = 20$, $\varepsilon = 10^{-10}$)

(b) NS-SDLS ($m = 80$, $n = 20$, $\varepsilon = 10^{-10}$)

(c) LMI-LS ($m = 500$, $n = 250$, $k = 20$, $\varepsilon = 10^{-10}$)

Figure 4.5: Duality gap convergence curves of typical random SDLS, NS-SDLS, and LMI-LS problems. The solid line corresponds to the SPF algorithm, the dashed line corresponds to the PC algorithm, and the dotted line corresponds to Woodgate's Algorithm [34].

In [14], the following ten force and displacement measurements were made from the nose of a stuffed toy tiger.

```
>> A

A =
   -0.3157    0.0330    0.0603
   -0.3274   -0.0158    0.0625
   -0.3569    0.0787    0.0563
   -0.2994    0.0301    0.0496
   -0.3243   -0.0048    0.0715
   -0.3447    0.0736    0.0545
   -0.2417    0.0709    0.0522
   -0.2063   -0.0099    0.0233
   -0.3285    0.1585    0.0979
   -0.2484    0.0878    0.0622
   -0.2196    0.0023    0.0280
   -0.3148    0.1506    0.0922

>> B

B =
   -1.4257    0.1528   -0.4398
   -1.4024   -0.3092   -0.4187
   -1.3766    0.4366   -0.4197
   -1.4274    0.1424   -0.4353
   -1.3994   -0.3095   -0.4206
   -1.3716    0.4285   -0.4193
   -1.4269    0.1581   -0.4335
   -1.4015   -0.3229   -0.4214
   -1.3767    0.4189   -0.4333
   -1.4257    0.1515   -0.4358
   -1.3989   -0.3276   -0.4217
   -1.3724    0.4154   -0.4356
```

As we can see, the regular unconstrained least squares solution, $\hat{X}$, which minimizes $\|AX^T - B\|_F$ over all possible 3×3 matrices X, does not satisfy our positive semidefinite requirement.

```
>> Xhat = (A\B)'

Xhat =
    5.1595    0.3075    2.3185
```

```
   -0.8348    6.2621   -8.1377
    1.5400   -0.0070    0.6169

>> norm(A*Xhat'-B,'fro')

ans =
    0.9805

>> eig(Xhat)

ans =
  -0.1683
   6.1034 + 0.8875i
   6.1034 - 0.8875i

>> eig((Xhat+Xhat')/2)

ans =
   -1.8795
    5.1456
    8.7725
```

Using the PC algorithm to compute an ε-approximate solution to the NS-SDLS problem, with $\varepsilon = 10^{-16}$, we obtain a matrix $\bar{X}$ which satisfies $\frac{1}{2}(\bar{X}+\bar{X}^T) \succeq \mathbf{0}$ and minimizes $\|AX^T-B\|_F$ over all 3×3 matrices X which satisfy $\frac{1}{2}(X + X^T) \succeq \mathbf{0}$.

```
>> [X,Y] = ns_sdls_precorr(A,B,[],[],1e-16,1);  Xbar = X'
```

r	sigma	theta	norm(res)	<S,Y>/n
0			5.4503e+00	1.0000e+00
1	1.1534e-01	8.0257e-01	1.0761e+00	1.1770e+00
2	9.2476e-02	8.8364e-01	1.2521e-01	1.7550e-01
3	1.4385e-01	9.0385e-01	1.2039e-02	3.7142e-02
4	2.7082e-02	1.0000e+00	8.9038e-16	3.1443e-03
5	1.5178e-02	1.0000e+00	6.4673e-16	2.1483e-04
6	5.1198e-04	9.9198e-01	8.9622e-16	2.4720e-06
7	1.5787e-06	9.8928e-01	8.0999e-16	2.6492e-08
8	1.3218e-06	9.8904e-01	4.0358e-16	2.9050e-10
9	1.3263e-06	9.8901e-01	5.1891e-16	3.1919e-12
10	1.3270e-06	9.8901e-01	8.8842e-16	3.5076e-14
11	1.3460e-06	9.8895e-01	6.6974e-16	3.8598e-16
12	4.1771e-06	9.8768e-01	5.5102e-16	6.2161e-18

```
Average iteration time: 0.0030 seconds
Total time:             0.0486 seconds

Xbar =
    5.0392    0.4423    1.5978
   -0.6207    6.0223   -6.8559
    1.8979   -0.4079    2.7600

>> norm(A*Xbar'-B,'fro')

ans =
    0.9859

>> eig(Xbar)

ans =
   1.1481
   6.3367 + 0.7865i
   6.3367 - 0.7865i

>> eig((Xbar+Xbar')/2)

ans =
    0.0000
    5.1401
    8.6813
```

As we see, the eigenvalues of $\bar{X}$ have nonnegative real parts and the symmetric part of $\bar{X}$, $\frac{1}{2}(\bar{X} + \bar{X}^T)$, is positive semidefinite. Furthermore, the fact that neither $\|\bar{X} - \hat{X}\|_F$ nor $\|A\bar{X}^T - B\|_F - \|A\hat{X}^T - B\|_F$ is large indicates that the unconstrained least squares solution, $\hat{X}$, was near the semidefinite constrained least squares solution, $\bar{X}$; this result is to be expected due to the physical nature of this problem.

```
>> norm(Xbar-Xhat,'fro')

ans =
    2.6795

>> norm(A*Xbar'-B,'fro') - norm(A*Xhat'-B,'fro')

ans =
    0.0055
```

Chapter 5
Conclusions and Future Work

5.1 Conclusions

The purpose of this thesis was to develop numerical methods for solving three semidefinite constrained least squares problems: the symmetric semidefinite least squares (SDLS) problem (1.3), the nonsymmetric semidefinite least squares (NS-SDLS) problem (1.4), and the linear matrix inequality least squares (LMI-LS) problem (1.5). Although the SDLS problem had been previously studied, we found that neither the NS-SDLS nor the LMI-LS problems appear to have been proposed before. Our inspiration for studying these problems came from the need to solve the NS-SDLS problem for the purpose of compliance matrix estimation in the modeling of deformable objects. The numerical methods we chose to use to solve each of these problems were the standard path-following (SPF) and predictor-corrector (PC) interior-point methods which have been used to solve the related semidefinite programming (SDP) and nonnegative least squares (NNLS) problems. While algorithms have already been proposed for the numerical solution of the SDLS problem, namely Woodgate's Algorithm [34], this appears to be the first time these interior-point methods have been used for this problem.

We presented a uniform discussion of the KKT conditions which characterize the solutions of each of the three semidefinite constrained least squares problems by developing these conditions for self-dual cone constrained convex programming problems in the general setting of Euclidean spaces. Using an important theorem due to Weierstrass, we were able to show that if the coefficient matrix A has full column rank then the SDLS and NS-SDLS problems have unique optimal solutions, and, under the additional assumption that the set of feasible points is nonempty, that the same holds true for the LMI-LS problem.

The key discovery made in this thesis was that each of the three problems under consideration are actually instances of a more general problem, the semidefinite linear complementarity problem (SDLCP). This connection to the SDLCP then assisted us in providing a uniform discussion of the interior-point methods we were considering. After discussing some known theoretical convergence results for interior-point methods applied to the SDLCP, we went on to describe techniques used for the numerical implementation of these interior-point methods for each of our three least squares problems.

Based on the numerical experiments performed in MATLAB with the SPF and PC algorithms using the AHO search direction, it was determined that the PC algorithm was the most efficient for all three problems. Moreover, the PC algorithm for the SDLS problem was seen to be an

improvement over the best current algorithm for solving this problem, Woodgate's Algorithm. We also saw how the PC algorithm for the NS-SDLS problem was able to efficiently compute an accurate estimate of the compliance matrix at a certain location on a deformable object that satisfied the required positive semidefinite constraint.

In conclusion, this thesis has made three important contributions. The first is the introduction and study of two new semidefinite constrained least squares problems, the NS-SDLS problem and the LMI-LS problem. The second is the proposed use of interior-point methods for the solution of the SDLS, NS-SDLS, and LMI-LS problems. Included in this second point are the resulting efficient method for compliance matrix estimation and an improved algorithm for solving the SDLS problem. The final contribution, while related to the first two, is the development of the KKT conditions for each of the three problems and the explicit connection of these conditions to the SDLCP.

5.2 Future Work

In this section we summarize the many promising ideas for possible future work which arose during the course of research for this thesis.

- When considering the existence of the central path in Section 3.3, the question as to when $\mathcal{L}_{++} \neq \emptyset$ holds for the LMI-LS problem remained undecided. This is an important question that should be the topic of future work in this area.
- More research could be done to propose more efficient ways to solve for the search directions in these interior-point methods. For example, it may be possible and preferable to solve a Lyapunov system directly rather than using the functions vec or svec in order to solve these large linear systems.
- Similarly, if the matrix A is very ill-conditioned, we could find that the methods mentioned here for solving for the search directions would not be successful due to the presence of the matrix $A^T A$ whose condition number is known to be the square of the condition number of A. It may be possible to determine an equivalent system which could be solved in a more stable manner using a QR factorization (see, for example, [23]).
- Another topic for future consideration is to develop methods for solving large sparse semidefinite constrained least squares problems.
- Finally, it would be interesting to determine if any of the other search directions mentioned here would result in more efficient algorithms for solving our three problems than the algorithms we have implemented here with the AHO search direction.

Bibliography

[1] M. Adlers. Sparse least squares problems with box constraints. *Linköping Studies in Science and Technology*, Linköping, 1998.

[2] F. Alizadeh, J. A. Haeberly, and M. Overton. Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results. *SIAM J. Optimization*, 8:746–768, 1998.

[3] J. C. Allwright. Positive semidefinite matrices: Characterization via conical hulls and least-squares solution of a matrix equation. *SIAM J. Control and Optimization*, 26:537–555, 1988.

[4] J. C. Allwright and K. G. Woodgate. Errata and addendum: Positive semidefinite matrices: Characterization via conical hulls and least-squares solution of a matrix equation. *SIAM J. Control and Optimization*, 28:250–251, 1990.

[5] Å. Björck. *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia, 1996.

[6] J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples.* Springer-Verlag, New York, 2000.

[7] J. E. Brock. *Optimal Matrices Describing Linear Systems.* AIAA Journal, 6:1292–1296, 1968.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* McGraw-Hill, New York, 1990.

[9] R. Escalante and M. Raydan. Dykstra's algorithm for a constrained least-squares matrix problem. *Numerical Linear Algebra with Applications*, 3:459–471, 1996.

[10] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, Cambridge, 1991.

[11] H. Hu. Positive definite constrained least-squares estimation of matrices. *Linear Algebra and Its Applications*, 229:167–174, 1995.

[12] M. Kojima, M. Shida, and S. Shindoh. A predictor-corrector interior-point algorithm for the semidefinite linear complementarity problem using the Alizadeh-Haeberly-Overton search direction. *SIAM J. Optimization*, 9:444–465, 1999.

[13] M. Kojima, S. Shindoh, and S. Hara. Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices. *SIAM J. Optimization*, 7:86–125, 1997.

[14] N. Krislock, J. Lang, J. Varah, D. K. Pai, and H.-P. Seidel. Local compliance estimation with a positive semi-definite constraint. Work in progress, 2003.

[15] J. Lang, D. K. Pai, and R. J. Woodham. Robotic acquisition of deformable models. *Proceedings of the 2002 IEEE International Conference on Robotics & Automation*, 1:933–938, 2002.

[16] A. Liao. On the least squares problem of a matrix equation. *Journal of Computational Mathematics*, 17:589–594, 1999.

[17] THE MATHWORKS, INC. MATLAB *Reference Guide.* The MathWorks, Inc., Natick, MA, 1992.

[18] R. D. C. Monteiro. Polynomial convergence of primal-dual algorithms for semidefinite programming based on the Monteiro and Zhang family of directions. *SIAM J. Optimization*, 8:797–812, 1998.

[19] R. D. C. Monteiro and T. Tsuchiya. Polynomiality of primal-dual algorithms for semidefinite linear complementarity problems based on the Kojima-Shindoh-Hara family of directions. *Math. Programming*, 85:51–80, 1999.

[20] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM, Philadelphia, 1994.

[21] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer-Verlag, New York, 2000.

[22] M. L. Overton. Private communication, 2003.

[23] L. F. Portugal, J. J. Júdice, and L. N. Vicente. A comparison of block pivoting and interior-point algorithms for linear least squares problems with nonnegative variables. *Mathematics of Computation*, 63:625–643, 1994.

[24] J. Renegar. *A Mathematical View of Interior-Point Methods in Convex Optimization.* SIAM, Philadelphia, 2001.

[25] R. T. Rockafellar. *Convex Analysis.* Princeton University Press, Princeton, New Jersey, 1970.

[26] M. Shida, S. Shindoh, and M. Kojima. Existence and uniqueness of search directions in interior-point algorithms for the SDP and the monotone SDLCP. *SIAM J. Optimization*, 8:387–396, 1998.

[27] M. J. Todd. Semidefinite optimization. *Acta Numerica*, 10:515–560, 2001.

[28] M. J. Todd, K. C. Toh, and R. H. Tütüncü. SDPT^3 - a MATLAB software package for semidefinite-quadratic-linear programming. http://www.math.cmu.edu/~reha/sdpt3.html, 2003.

[29] M. J. Todd, K. C. Toh, and R. H. Tütüncü. On the Nesterov-Todd direction in semidefinite programming. *SIAM J. Optimization*, 8:769–796, 1998.

[30] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

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

[32] K. G. Woodgate. *Optimization over positive semi-definite symmetric matrices with applications to quasi-Newton algorithms.* Ph.D. thesis, Dept. of Electrical Engineering, Imperial College, London, England, 1987.

[33] K. G. Woodgate. Least-squares solution of $F = PG$ over positive semidefinite symmetric P. *Linear Algebra and Its Applications*, 245:171–190, 1996.

[34] K. G. Woodgate. Efficient stiffness matrix estimation for elastic structures. *Computers and Structures*, 69:79–84, 1998.

[35] D.-X. Xie. Least-squares solution of inverse eigenpair problem of nonnegative definite matrices. *Computers and Mathematics with Applications*, 40:1241–1251, 2000.

[36] Y. Zhang. On extending some primal-dual interior-point algorithms from linear programming to semidefinite programming. *SIAM J. Optimization*, 8:365–386, 1998.

Appendix A
Kronecker products

At many times in our discussion, we must solve a system of linear equations, but the variables we are solving for reside in a matrix rather than a vector. Here we follow the development in [2] and [29] by noticing that an easy way to deal with this situation is to rewrite the linear system by putting the entries of the unknown matrix into a vector. We do this by defining the function vec : $\mathbb{R}^{n\times n} \to \mathbb{R}^{n^2}$ as

$$\text{vec}(X) = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_n \end{bmatrix},$$

where $X = [\bar{x}_1\, \bar{x}_2 \cdots \bar{x}_n]$. In other words, the function vec stacks the columns of a matrix into a vector. In fact, vec is an isometry between $\mathbb{R}^{n\times n}$ and $\mathbb{R}^{n^2}$ in that it is a linear bijective function that preserves the inner-product: $\langle X, Y\rangle = \langle \text{vec}(X), \text{vec}(Y)\rangle$, for all $X, Y \in \mathbb{R}^{n\times n}$. We denote the inverse of vec as the function mat : $\mathbb{R}^{n^2} \to \mathbb{R}^{n\times n}$, which is defined in the obvious way.

Using vec to rewrite the linear matrix equation

$$AXB^T = C,$$

where $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{q\times n}$, $X \in \mathbb{R}^{n\times n}$, and $C \in \mathbb{R}^{p\times q}$, as the linear equation

$$M\text{vec}(X) = \text{vec}(C),$$

we find that the matrix $M \in \mathbb{R}^{pq\times n^2}$ is the *Kronecker product* of B and A, which is defined as

$$B \otimes A := \begin{bmatrix} b_{11}A & \cdots & b_{1n}A \\ \vdots & \ddots & \vdots \\ b_{q1}A & \cdots & b_{qn}A \end{bmatrix}. \tag{A.1}$$

The properties of the Kronecker product (see [29, p. 791, 794]) can be summarized as follows:

1. $(B \otimes A)\text{vec}(X) = \text{vec}(AXB^T)$.
2. $(B \otimes A)^T = B^T \otimes A^T$.

3. $(B \otimes A)^{-1} = B^{-1} \otimes A^{-1}$.
4. $(B \otimes A)(D \otimes C) = BD \otimes AC$.
5. If $\sigma(A) = \{\lambda_i\}$ and $\sigma(B) = \{\mu_j\}$, then $\sigma(B \otimes A) = \{\lambda_i \mu_j\}$.
6. If A and B are symmetric and positive definite, then so is $B \otimes A$.

We can also define an isometry between real symmetric matrices and vectors. We denote $\mathrm{svec} : \mathcal{S}^n \to \mathbb{R}^{\bar{n}}$, where $\bar{n} := \frac{1}{2}n(n+1)$, as the function defined by

$$\mathrm{svec}(X) = [x_{11}, \sqrt{2}x_{21}, \ldots, \sqrt{2}x_{n1}, x_{22}, \sqrt{2}x_{32}, \ldots, \sqrt{2}x_{n2}, \ldots\ldots, x_{nn}]^T,$$

where $X = [x_{ij}]$. The reason for scaling the off-diagonal entries of X by $\sqrt{2}$ is so that the inner-product is preserved: $\langle X, Y \rangle = \langle \mathrm{svec}(X), \mathrm{svec}(Y) \rangle$, for all $X, Y \in \mathcal{S}^n$. The inverse of svec is denoted as the function $\mathrm{smat} : \mathbb{R}^{\bar{n}} \to \mathcal{S}^n$.

Now suppose we would like to solve the linear matrix equation

$$\tfrac{1}{2}(AXB^T + BXA^T) = C$$

for $X \in \mathcal{S}^n$, where $A, B \in \mathbb{R}^{n \times n}$ and $C \in \mathcal{S}^n$. We define the *symmetric Kronecker product* of B and A, denoted as $B \otimes_s A$, as the matrix in $\mathbb{R}^{\bar{n} \times \bar{n}}$ which satisfies

$$(B \otimes_s A)\mathrm{svec}(X) = \mathrm{svec}\left(\tfrac{1}{2}(AXB^T + BXA^T)\right), \text{ for all } X \in \mathcal{S}^n.$$

In fact, we can compute $B \otimes_s A$ as

$$B \otimes_s A = \tfrac{1}{2}U^T(B \otimes A + A \otimes B)U, \tag{A.2}$$

where U is the $n^2 \times \bar{n}$ matrix that satisfies

$$U\mathrm{svec}(X) = \mathrm{vec}(X), \text{ for all } X \in \mathcal{S}^n.$$

To compute U, let $\bar{e}_i$ be the i^{th} unit vector in $\mathbb{R}^{\bar{n}}$, for $i = 1, \ldots, \bar{n}$, and let $E_i = \mathrm{smat}(\bar{e}_i) \in \mathcal{S}^n$. Then E_i is of the form

$$E_i = \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & 0 & \cdots & 1/\sqrt{2} & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & 1/\sqrt{2} & \cdots & 0 & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \quad \text{or} \quad E_i = \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & 1 & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \end{bmatrix},$$

and the i^{th} column of U is then

$$\bar{u}_i = U\bar{e}_i = U\mathrm{svec}(E_i) = \mathrm{vec}(E_i) = \mathrm{vec}(\mathrm{smat}(\bar{e}_i)).$$

Therefore, we find that

$$U = [\mathrm{vec}(\mathrm{smat}(\bar{e}_1)) \cdots \mathrm{vec}(\mathrm{smat}(\bar{e}_{\bar{n}}))],$$

and since

$$\bar{u}_i^T \bar{u}_j = \langle E_i, E_j \rangle = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j, \end{cases}$$

we also find that $U^T U = I \in \mathbb{R}^{\bar{n} \times \bar{n}}$. Thus, we have

$$U^T \mathrm{vec} X = \mathrm{svec}(X), \text{ for all } X \in \mathcal{S}^n,$$

and identity (A.2) follows.

We now summarize the properties of the symmetric Kronecker product (see [29, p. 794]):

1. $(B \otimes_s A)\mathrm{svec}(X) = \mathrm{svec}\left(\frac{1}{2}(AXB^T + BXA^T)\right)$.
2. $(B \otimes_s A)^T = B^T \otimes_s A^T$.
3. $B \otimes_s A = A \otimes_s B$.
4. $(B \otimes_s A)(D \otimes_s C) = \frac{1}{2}(BD \otimes_s AC + BC \otimes_s AD)$.
5. If $A, B \in \mathcal{S}^n$ commute, $\sigma(A) = \{\lambda_i\}$, and $\sigma(B) = \{\mu_j\}$, then
$$\sigma(B \otimes_s A) = \left\{\tfrac{1}{2}(\lambda_i \mu_j + \lambda_j \mu_i)\right\}.$$
6. If A and B are symmetric and positive definite, then so is $A \otimes_s B$.

Appendix B
Matlab M-files

B.1 SDLS M-files

B.1.1 sdls.m

```
function [X,Y,norm_res,muv,tt,iter,fail]  = sdls(A,B,X0,Y0,tol,verbose)
% [X,Y,norm_res,muv,tt,iter,fail] = SDLS(A,B,X0,Y0,tol,verbose)
%
% Uses a stanadard path-following interior-point method based on the
% AHO search direction to solve the symmetric semidefinite constrained
% least squares problem:
%
%   min  norm(A*X-B,'fro')
%   s.t. X symm. pos. semidef.
%
% where A and B are real m-by-n matrices, and X is a real n-by-n matrix.
%
% X0 and Y0 are n-by-n initial strictly feasible matrices, which means
% that X0 and Y0 are symmetric positive definite.
% Set as [] for the default value of eye(n).
%
% tol is the zero tolerance described below.
% Set as [] for the default value of 1e-10.
%
% Set verbose = 1 for screen output during algorithm execution,
% otherwise set vebose = 0 for no output.
%
% SDLS returns approximate optimal solutions to the above primal
% problem and its associated dual problem so that
%
%   norm(res,'fro')  <=  sqrt(tol)*norm(res0,'fro')
%        trace(X*Y)  <=  tol*trace(X0*Y0)
%
% where res = (Z+Z')/2-Y, Z =  A'*(A*X-B), and res0 is res evaluated
```

```
% at X0, Y0.
%
% SDLS optionally returns:
%
%  norm_res : norm(res,'fro') at the final iterate,
%  muv   : a vector of the duality gaps for each iteration
%  tt    : the total running time of the algorithm
%  iter : the number of iterations required
%  fail : fail = 1 if the algorithm failed to achieve the desired
%         tolerances within the maximum number of iterations allowed;
%         otherwise fail = 0

% Nathan Krislock, University of British Columbia, 2003.
%
% N. Krislock. Numerical solution of semidefinite constrained least
% squares problems.  M.Sc. thesis, University of British Columbia,
% Vancouver, British Columbia, Canada, 2003.


tic;  % Start the preprocessing timer

MaxIt = 100;  % max iteration

[m,n] = size(A);
AA = A'*A;  AB = A'*B;  I = eye(n);

if isempty(X0), X = I; else, X = X0; end
if isempty(Y0), Y = I; else, Y = Y0; end
if isempty(tol), tol = 1e-10; end

XAA = X*AA;  XY = X*Y;
Z = XAA' - AB;  Z = (Z+Z')/2;  R = Z - Y;

norm_res = norm(R,'fro');  mu = trace(XY)/n;  muv = mu;

tol1 = sqrt(tol)*norm_res;  tol2 = tol*mu;

r = 0;  theta = 0;

if verbose==1
    disp(' ');
    disp([' r      sigma         theta        norm(res)     <X,Y>/n']);
    disp([' ---   ----------    ----------    ----------    ----------']);

    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('                              ') ];
```

```
    ol = [ ol, sprintf('  %12.4e',[norm_res, mu]) ];
    disp(ol);
end

pretime = toc;  % End the preprocessing timer


while ( norm_res > tol1 | mu > tol2 ) & r < MaxIt

    tic;  % Start the iteration timer


    % Compute sigma and tau

    if norm_res < tol1
        sigma = 1/n^2;
    else
        sigma = 1-1/n^2;
    end

    tau = sigma*mu;


    % Compute the AHO search direction (dX,dY)

    E = (kron(I,Y)+kron(Y,I))/2;  % F = (kron(I,X)+kron(X,I))/2;

    XZ = X*Z;

    M = (kron(I,XAA)+kron(AA,X)+kron(X,AA)+kron(XAA,I))/4 + E;
        % M = F*kronAA + E;
    d = vec(tau*I - (XZ+XZ')/2);
        % d = F*vec(-R) + vec(tau*I-(X*Y+Y*X)/2);

    [L,U,P] = lu(M);
    dx = U\(L\(P*d));

    dX = mat(dx);  dX = (dX+dX')/2;  AAdX = AA*dX;
    dY = (AAdX+AAdX')/2 + R;  dY = (dY+dY')/2;


    % Compute the step length theta

    c = 0.9 + 0.09*theta;

    theta1 = max_step(X,dX);  theta2 = max_step(Y,dY);
```

```
    theta_max = min([theta1, theta2]);

    theta = min([c*theta_max,1]);


    % Update

    X = X + theta*dX;
    Y = Y + theta*dY;

    XAA = X*AA;  XY = X*Y;
    Z = XAA' - AB;  Z = (Z+Z')/2;  R = Z - Y;

    norm_res = norm(R,'fro');  mu = trace(XY)/n;  muv(r+2) = mu;

    r = r + 1;

    if verbose==1
        ol = sprintf('%3d',r);
        ol = [ ol, sprintf('  %12.4e',[sigma, theta, norm_res, mu]) ];
        disp(ol);
    end

    t(r) = toc;  % End the iteration timer

end


if r==MaxIt & ( norm_res > tol1 | mu > tol2 )
    fail = 1;
else
    fail = 0;
end

avt = mean(t);  tt = sum(t) + pretime;  iter = r;

if verbose==1
    if fail==1,
        disp(' ');
        disp('Failed to reach desired tolerance.');
    end
    disp(' ');
    disp(sprintf('Average iteration time:\t%2.4f seconds', avt));
    disp(sprintf('Total time:\t\t%2.4f seconds', tt));
    disp(' ');
end
```

B.1.2 sdls_precorr.m

```
function [X,Y,norm_res,muv,tt,iter,fail] = sdls_precorr(A,B,X0,Y0,tol,verbose)
% [X,Y,norm_res,muv,tt,iter,fail] = SDLS_PRECORR(A,B,X0,Y0,tol,verbose)
%
% Uses a predictor-corrector interior-point method to solve the
% symmetric semidefinite constrained least squares problem:
%
%   min  norm(A*X-B,'fro')
%   s.t. X symm. pos. semidef.
%
% where A and B are real m-by-n matrices, and X is a real n-by-n matrix.
%
% X0 and Y0 are n-by-n initial strictly feasible matrices, which means
% that X0 and Y0 are symmetric positive definite.
% Set as [] for the default value of eye(n).
%
% tol is the zero tolerance described below.
% Set as [] for the default value of 1e-10.
%
% Set verbose = 1 for screen output during algorithm execution,
% otherwise set vebose = 0 for no output.
%
% SDLS_PRECORR returns approximate optimal solutions to the above
% primal problem and its associated dual problem so that
%
%   norm(res,'fro')  <=  sqrt(tol)*norm(res0,'fro')
%        trace(X*Y)  <=  tol*trace(X0*Y0)
%
% where res = (Z+Z')/2-Y, Z =  A'*(A*X-B), and res0 is res evaluated
% at X0, Y0.
%
% SDLS_PRECORR optionally returns:
%
%  norm_res : norm(res,'fro') at the final iterate,
%  muv  : a vector of the duality gaps for each iteration
%  tt   : the total running time of the algorithm
%  iter : the number of iterations required
%  fail : fail = 1 if the algorithm failed to achieve the desired
%         tolerances within the maximum number of iterations allowed;
%         otherwise fail = 0

% Nathan Krislock, University of British Columbia, 2003.
%
% N. Krislock. Numerical solution of semidefinite constrained least
% squares problems.  M.Sc. thesis, University of British Columbia,
```

```
% Vancouver, British Columbia, Canada, 2003.


tic;  % Start the preprocessing timer

MaxIt = 100;  % max iteration

[m,n] = size(A);
AA = A'*A;  AB = A'*B;  I = eye(n);

if isempty(X0), X = I; else, X = X0; end
if isempty(Y0), Y = I; else, Y = Y0; end
if isempty(tol), tol = 1e-10; end

XAA = X*AA;  XY = X*Y;
Z = XAA' - AB;  Z = (Z+Z')/2;  R = Z - Y;

norm_res = norm(R,'fro');  mu = trace(XY)/n;  muv = mu;

tol1 = sqrt(tol)*norm_res;  tol2 = tol*mu;

r = 0;  theta = 0;

if verbose==1
    disp(' ');
    disp([' r      sigma         theta        norm(res)     <X,Y>/n']);
    disp([' ---   ----------    ----------    ----------    ----------']);

    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('                              ') ];
    ol = [ ol, sprintf('  %12.4e',[norm_res, mu]) ];
    disp(ol);
end

pretime = toc;  % End the preprocessing timer


while ( norm_res > tol1 | mu > tol2 ) & r < MaxIt

    tic;  % Start the iteration timer


    % Compute the predictor directions dXp, dYp

    E = (kron(I,Y)+kron(Y,I))/2;  % F = (kron(I,X)+kron(X,I))/2;
```

```
XZ = X*Z;

M = (kron(I,XAA)+kron(AA,X)+kron(X,AA)+kron(XAA,I))/4 + E;
    % M = F*kronAA + E;
d = -vec((XZ+XZ')/2);
    % d = F*vec(-R) + vec(-(X*Y+Y*X)/2);

[L,U,P] = lu(M);
dxp = U\(L\(P*d));

dXp = mat(dxp);  dXp = (dXp+dXp')/2;  AAdXp = AA*dXp;
dYp = (AAdXp+AAdXp')/2 + R;  dYp = (dYp+dYp')/2;


% Compute predictor step length ptheta

c = 0.9 + 0.09*theta;

ptheta1 = max_step(X,dXp);  ptheta2 = max_step(Y,dYp);
ptheta_max = min([ptheta1, ptheta2]);

ptheta = min([c*ptheta_max,1]);


% Compute the Mehrotra sigma and parameter tau

muhat = sum(sum((X+ptheta*dXp).*(Y+ptheta*dYp)))/n;

sigma = (muhat/mu)^3;
tau = sigma*mu;


% Compute the predictor-corrector directions dX, dY

dYpdXp = dYp*dXp;
d = d + vec(tau*I-(dYpdXp+dYpdXp')/2);

dx = U\(L\(P*d));

dX = mat(dx);  dX = (dX+dX')/2;  AAdX = AA*dX;
dY = (AAdX+AAdX')/2 + R;  dY = (dY+dY')/2;


% Compute the step length theta

theta1 = max_step(X,dX);  theta2 = max_step(Y,dY);
```

```
    theta_max = min([theta1, theta2]);

    theta = min([c*theta_max,1]);


    % Update

    X = X + theta*dX;
    Y = Y + theta*dY;

    XAA = X*AA;  XY = X*Y;
    Z = XAA' - AB;  Z = (Z+Z')/2;  R = Z - Y;

    norm_res = norm(R,'fro');  mu = trace(XY)/n;  muv(r+2) = mu;

    r = r + 1;

    if verbose==1
        ol = sprintf('%3d',r);
        ol = [ ol, sprintf('  %12.4e',[sigma, theta, norm_res, mu]) ];
        disp(ol);
    end

    t(r) = toc;  % End the iteration timer

end


if r==MaxIt & ( norm_res > tol1 | mu > tol2 )
    fail = 1;
else
    fail = 0;
end

avt = mean(t);  tt = sum(t) + pretime;  iter = r;

if verbose==1
    if fail==1,
        disp(' ');
        disp('Failed to reach desired tolerance.');
    end
    disp(' ');
    disp(sprintf('Average iteration time:\t%2.4f seconds', avt));
    disp(sprintf('Total time:\t\t%2.4f seconds', tt));
    disp(' ');
end
```

B.2 NS-SDLS M-files

B.2.1 ns_sdls.m

```
function [X,Y,norm_res,muv,tt,iter,fail] = ns_sdls(A,B,X0,Y0,tol,verbose)
% [X,Y,norm_res,muv,tt,iter,fail] = NS_SDLS(A,B,X0,Y0,tol,verbose)
%
% Uses a stanadard path-following interior-point method to solve the
% nonsymmetric semidefinite constrained least squares problem:
%
%   min  norm(A*X-B,'fro')
%   s.t. (X+X')/2 pos. semidef.
%
% where A and B are real m-by-n matrices, and X is a real n-by-n matrix.
%
% X0 and Y0 are n-by-n initial strictly feasible matrices, which means
% that (X0+X0')/2 and Y0 are symmetric positive definite.
% Set as [] for the default value of eye(n).
%
% tol is the zero tolerance described below.
% Set as [] for the default value of 1e-10.
%
% Set verbose = 1 for screen output during algorithm execution,
% otherwise set vebose = 0 for no output.
%
% NS_SDLS returns approximate optimal solutions to the above primal
% problem and its associated dual problem so that
%
%   norm(res,'fro')  <=  sqrt(tol)*norm(res0,'fro')
%        trace(S*Y)  <=  tol*trace(S0*Y0)
%
% where res = A'*(A*X-B)-Y, S = (X+X')/2 and res0 and S0 are res
% and S evaluated at X0, Y0.
%
% NS_SDLS optionally returns:
%
%  norm_res : norm(res,'fro') at the final iterate,
%  muv   : a vector of the duality gaps for each iteration
%  tt    : the total running time of the algorithm
%  iter  : the number of iterations required
%  fail  : fail = 1 if the algorithm failed to achieve the desired
%          tolerances within the maximum number of iterations allowed;
%          otherwise fail = 0

% Nathan Krislock, University of British Columbia, 2003.
%
```

```
% N. Krislock. Numerical solution of semidefinite constrained least
% squares problems.  M.Sc. thesis, University of British Columbia,
% Vancouver, British Columbia, Canada, 2003.


tic;  % Start the preprocessing timer

MaxIt = 100;  % max iteration

[m,n] = size(A);
AA = A'*A;  AB = A'*B;  I = eye(n);

V = vecV(n);  % vec((X+X')/2) = V*vec(X)

if isempty(X0), X = I; else, X = X0; end,  S = (X+X')/2;
if isempty(Y0), Y = I; else, Y = Y0; end
if isempty(tol), tol = 1e-10; end

Z = AA*X - AB;  R = Z - Y;  SY = S*Y;

norm_res = norm(R,'fro');  mu = trace(SY)/n;  muv = mu;

tol1 = sqrt(tol)*norm_res;  tol2 = tol*mu;

r = 0;  theta = 0;

if verbose==1
    disp(' ');
    disp([' r       sigma        theta        norm(res)     <S,Y>/n']);
    disp([' ---   ----------    ----------    ----------    ----------']);

    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('                                ') ];
    ol = [ ol, sprintf('  %12.4e',[norm_res, mu]) ];
    disp(ol);
end

pretime = toc;  % End the preprocessing timer


while ( norm_res > tol1 | mu > tol2 ) & r < MaxIt

    tic;  % Start the iteration timer


    % Compute sigma and tau
```

```
if norm_res < tol1,
    sigma = 1/n^2;
else
    sigma = 1-1/n^2;
end

tau = sigma*mu;


% Find suitable directions dX, dS, dY

E = (kron(I,Y)+kron(Y,I))*V/2;  % F = (kron(I,S)+kron(S,I))/2;

M = (kron(I,S*AA)+kron(S,AA))/2 + E;  % M = F*kron(I,AA) + E;
d = vec(tau*I - (S*R+R*S+SY+SY'))/2);  % d = F*vec(-R)
                                       %      + vec(tau*I-(S*Y+Y*S)/2);
[L,U,P] = lu(M);
dx =  U\(L\(P*d));

dX = mat(dx);    dS = (dX+dX')/2;
dY = AA*dX + R;  dY = (dY+dY')/2;


% Find step length theta

c = 0.9 + 0.09*theta;

theta1 = max_step(S,dS);  theta2 = max_step(Y,dY);
theta_max = min([theta1, theta2]);

theta = min([c*theta_max,1]);


% Update

X = X + theta*dX;  S = (X+X')/2;
Y = Y + theta*dY;

Z = AA*X - AB;  R = Z - Y;  SY = S*Y;

norm_res = norm(R,'fro');  mu = trace(SY)/n;  muv(r+2) = mu;

r = r + 1;

if verbose==1
```

```
        ol = sprintf('%3d',r);
        ol = [ ol, sprintf('  %12.4e',[sigma, theta, norm_res, mu]) ];
        disp(ol);
    end

    t(r) = toc;  % End the iteration timer

end


if r==MaxIt & ( norm_res > tol1 | mu > tol2 )
    fail = 1;
else
    fail = 0;
end

avt = mean(t);  tt = sum(t) + pretime;  iter = r;

if verbose==1
    if fail==1,
        disp(' ');
        disp('Failed to reach desired tolerance.');
    end
    disp(' ');
    disp(sprintf('Average iteration time:\t%2.4f seconds', avt));
    disp(sprintf('Total time:\t\t%2.4f seconds', tt));
    disp(' ');
end
```

B.2.2 ns_slds_precorr.m

```
function [X,Y,norm_res,muv,tt,iter,fail]
                        = ns_sdls_precorr(A,B,X0,Y0,tol,verbose)
% [X,Y,norm_res,muv,tt,iter,fail] = NS_SDLS_PRECORR(A,B,X0,Y0,tol,verbose)
%
% Uses a predictor-corrector interior-point method to solve the
% nonsymmetric semidefinite constrained least squares problem:
%
%   min  norm(A*X-B,'fro')
%   s.t. (X+X')/2 pos. semidef.
%
% where A and B are real m-by-n matrices, and X is a real n-by-n matrix.
%
% X0 and Y0 are n-by-n initial strictly feasible matrices, which means
% that (X0+X0')/2 and Y0 are symmetric positive definite.
% Set as [] for the default value of eye(n).
%
% tol is the zero tolerance described below.
% Set as [] for the default value of 1e-10.
%
% Set verbose = 1 for screen output during algorithm execution,
% otherwise set vebose = 0 for no output.
%
% NS_SDLS_PRECORR returns approximate optimal solutions to the above
% primal problem and its associated dual problem so that
%
%   norm(res,'fro')  <=  sqrt(tol)*norm(res0,'fro')
%        trace(S*Y)  <=  tol*trace(S0*Y0)
%
% where res = A'*(A*X-B)-Y, S = (X+X')/2 and res0 and S0 are res
% and S evaluated at X0, Y0.
%
% NS_SDLS_PRECORR optionally returns:
%
%  norm_res : norm(res,'fro') at the final iterate,
%  muv  : a vector of the duality gaps for each iteration
%  tt   : the total running time of the algorithm
%  iter : the number of iterations required
%  fail : fail = 1 if the algorithm failed to achieve the desired
%         tolerances within the maximum number of iterations allowed;
%         otherwise fail = 0

% Nathan Krislock, University of British Columbia, 2003.
%
% N. Krislock. Numerical solution of semidefinite constrained least
```

```
% squares problems.  M.Sc. thesis, University of British Columbia,
% Vancouver, British Columbia, Canada, 2003.


tic;  % Start the preprocessing timer

MaxIt = 100;  % max iteration

[m,n] = size(A);
AA = A'*A;  AB = A'*B;  I = eye(n);

V = vecV(n);  % vec((X+X')/2) = V*vec(X)

if isempty(X0), X = I; else, X = X0; end,  S = (X+X')/2;
if isempty(Y0), Y = I; else, Y = Y0; end
if isempty(tol), tol = 1e-10; end

Z = AA*X - AB;  R = Z - Y;  SY = S*Y;

norm_res = norm(R,'fro');  mu = trace(SY)/n;  muv = mu;

tol1 = sqrt(tol)*norm_res;  tol2 = tol*mu;

r = 0;  theta = 0;

if verbose==1
    disp(' ');
    disp([' r       sigma        theta        norm(res)     <S,Y>/n']);
    disp([' ---   ----------    ----------    ----------    ----------']);

    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('                              ') ];
    ol = [ ol, sprintf('  %12.4e',[norm_res, mu]) ];
    disp(ol);
end

pretime = toc;  % End the preprocessing timer


while ( norm_res > tol1 | mu > tol2 ) & r < MaxIt

    tic;  % Start the iteration timer


    % Compute the predictor directions dXp, dSp, dYp
```

```
E = (kron(I,Y)+kron(Y,I))*V/2;  % F = (kron(I,S)+kron(S,I))/2;

M = (kron(I,S*AA)+kron(S,AA))/2 + E;  % M = F*kronAA + E;
d = -vec((S*R+R*S+SY+SY')/2);  % d = F*vec(-R) + vec(-(S*Y+Y*S)/2);

[L,U,P] = lu(M);
dxp =  U\(L\(P*d));

dXp = mat(dxp);     dSp = (dXp+dXp')/2;
dYp = AA*dXp + R;   dYp = (dYp+dYp')/2;


% Compute predictor step length ptheta

c = 0.9 + 0.09*theta;

ptheta1 = max_step(S,dSp);  ptheta2 = max_step(Y,dYp);
ptheta_max = min([ptheta1, ptheta2]);

ptheta = min([c*ptheta_max,1]);


% Compute the Mehrotra sigma and parameter tau

muhat = sum(sum((S+ptheta*dSp).*(Y+ptheta*dYp)))/n;

sigma = (muhat/mu)^3;
tau = sigma*mu;


% Compute the predictor-corrector directions dX, dS, dY

dYpdSp = dYp*dSp;

d = d + vec(tau*I-(dYpdSp+dYpdSp')/2);

dx = U\(L\(P*d));

dX = mat(dx);     dS = (dX+dX')/2;
dY = AA*dX + R;   dY = (dY+dY')/2;


% Compute the step length theta

theta1 = max_step(S,dS);  theta2 = max_step(Y,dY);
```

```
    theta_max = min([theta1, theta2]);
    theta = min([c*theta_max,1]);


    % Update

    X = X + theta*dX;  S = (X+X')/2;
    Y = Y + theta*dY;

    Z = AA*X - AB;  R = Z - Y;  SY = S*Y;

    norm_res = norm(R,'fro');  mu = trace(SY)/n;  muv(r+2) = mu;

    r = r + 1;

    if verbose==1
        ol = sprintf('%3d',r);
        ol = [ ol, sprintf('  %12.4e',[sigma, theta, norm_res, mu]) ];
        disp(ol);
    end

    t(r) = toc;  % End the iteration timer

end


if r==MaxIt & ( norm_res > tol1 | mu > tol2 )
    fail = 1;
else
    fail = 0;
end

avt = mean(t);  tt = sum(t) + pretime;  iter = r;

if verbose==1
    if fail==1,
        disp(' ');
        disp('Failed to reach desired tolerance.');
    end
    disp(' ');
    disp(sprintf('Average iteration time:\t%2.4f seconds', avt));
    disp(sprintf('Total time:\t\t%2.4f seconds', tt));
    disp(' ');
end
```

B.3 LMI-LS M-files

B.3.1 lmi_ls.m

```
function [x,S,Y,norm_res,muv,tt,iter,fail]
                        = lmi_ls(A,b,KK,C,x0,S0,Y0,tol,verbose)
% [x,S,Y,norm_res,muv,tt,iter,fail] = LMI_LS(A,b,KK,C,x0,S0,Y0,tol,verbose)
%
% Uses a stanadard path-following interior-point method to solve the
% linear matrix inequality least squares problem:
%
%   min  norm(A*x-b)
%   s.t. K(x) + S = C
%        S is symm. pos. semidef.
%
% where A is a real m-by-n matrix, x is a real n-by-1 vector,
% b is a real m-by-1 vector, S and C are real n-by-n symmetric
% matrices, and K(x) = sum( x(i)*K_i, i=1:n ) where KK = [K_1,...,K_n]
% and K_i is a real k-by-k symmetric matrix for i=1:n.
%
% S0 and Y0 are n-by-n initial strictly feasible matrices, which means
% that S0 and Y0 are symmetric positive definite.
% Set as [] for the default value of eye(n).
%
% tol is the zero tolerance described below.
% Set as [] for the default value of 1e-10.
%
% Set verbose = 1 for screen output during algorithm execution,
% otherwise set vebose = 0 for no output.
%
% LMI_LS returns approximate optimal solutions to the above primal
% problem and its associated dual problem so that
%
%   norm(res,'fro')  <=  sqrt(tol)*norm(res0,'fro')
%        trace(S*Y)  <=  tol*trace(S0*Y0)
%
% where res = [rp; rd] is the primal-dual residual vector
% and res0 is res evaluated at x0, S0, Y0.
%
% LMI_LS optionally returns:
%
%  norm_res : norm(res,'fro') at the final iterate,
%  muv  : a vector of the duality gaps for each iteration
%  tt   : the total running time of the algorithm
%  iter : the number of iterations required
%  fail : fail = 1 if the algorithm failed to achieve the desired
```

```
%         tolerances within the maximum number of iterations allowed;
%         otherwise fail = 0

% Nathan Krislock, University of British Columbia, 2003.
%
% N. Krislock. Numerical solution of semidefinite constrained least
% squares problems.  M.Sc. thesis, University of British Columbia,
% Vancouver, British Columbia, Canada, 2003.


tic;  % Start the preprocessing timer

MaxIt = 100;  % max iteration

[m,n] = size(A);  k = length(C);
AA = A'*A;  Ab = A'*b;  I = eye(k);  Ik2 = eye(k^2);
O = zeros(k^2,n); Ok2 = zeros(k^2,k^2);

K = vecK(KK);  % vec(K(x)) = K*x,  K^{*}(Y) = K'*vec(Y)

if isempty(Y0), Y = I; else, Y = Y0; end
if isempty(x0), x = zeros(n,1); else, x = x0; end
if isempty(S0), S = I; else, S = S0; end
if isempty(tol), tol = 1e-10; end

Kx = mat(K*x);  y = vec(Y);  KY = K'*y;  SY = S*Y;

z = Ab - AA*x;  rd = z - KY;
Rp = C - Kx - S;  rp = vec(Rp);

norm_res = norm([rp;rd]);  mu = trace(SY)/k;  muv = mu;

tol1 = sqrt(tol)*norm_res;  tol2 = tol*mu;

r = 0;  theta = 0;

if verbose==1
    disp(' ');
    disp([' r      sigma         theta        norm(res)     <S,Y>/n']);
    disp([' ---   ----------    ----------    ----------    ----------']);

    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('                              ') ];
    ol = [ ol, sprintf('  %12.4e',[norm_res, mu]) ];
    disp(ol);
end
```

```
pretime = toc;  % End the preprocessing timer


while ( norm_res > tol1 | mu > tol2 ) & r < MaxIt

    tic;  % Start the iteration timer


    % Compute sigma and tau

    if norm_res < tol1,
        sigma = 1/k^2;
    else
        sigma = 1-1/k^2;
    end

    tau = sigma*mu;


    % Find suitable directions dx, dS, dY

    E = (kron(I,S)+kron(S,I))/2;  F = (kron(I,Y)+kron(Y,I))/2;
    D = tau*I-(SY+SY')/2;  rc = vec(D);

    M = [ AA,  K',  0';
          K, 0k2, Ik2;
          0,  E ,  F   ];

    d = [ rd; rp; rc ];

    w = M\d;

    dx = w(1:n);
    dy = w((n+1):(k^2+n));
    ds = w((k^2+n+1):(2*k^2+n));

    dY = mat(dy);  dY = (dY+dY')/2;
    dS = mat(ds);  dS = (dS+dS')/2;


    % Find step length theta

    c = 0.9 + 0.09*theta;

    theta1 = max_step(S,dS);  theta2 = max_step(Y,dY);
```

```
    theta_max = min([theta1, theta2]);

    theta = min([c*theta_max,1]);


    % Update

    Y = Y + theta*dY;
    x = x + theta*dx;
    S = S + theta*dS;

    Kx = mat(K*x);  y = vec(Y);  KY = K'*y;  SY = S*Y;

    z = Ab - AA*x;  rd = z - KY;
    Rp = C - Kx - S;  rp = vec(Rp);

    norm_res = norm([rp;rd]);  mu = trace(SY)/k;  muv(r+2) = mu;

    r = r + 1;

    if verbose==1
        ol = sprintf('%3d',r);
        ol = [ ol, sprintf('  %12.4e',[sigma, theta, norm_res, mu]) ];
        disp(ol);
    end

    t(r) = toc;  % End the iteration timer

end


if r==MaxIt & ( norm_res > tol1 | mu > tol2 )
    fail = 1;
else
    fail = 0;
end

avt = mean(t);  tt = sum(t) + pretime;  iter = r;

if verbose==1
    if fail==1,
        disp(' ');
        disp('Failed to reach desired tolerance.');
    end
    disp(' ');
    disp(sprintf('Average iteration time:\t%2.4f seconds', avt));
```

```
    disp(sprintf('Total time:\t\t%2.4f seconds', tt));
    disp(' ');
end
```

B.3.2 lmi_ls_precorr.m

```
function [x,S,Y,norm_res,muv,tt,iter,fail]
                    = lmi_ls_precorr(A,b,KK,C,x0,S0,Y0,tol,verbose)
% [x,S,Y,norm_res,muv,tt,iter,fail]
%                    = LMI_LS_PRECORR(A,b,KK,C,x0,S0,Y0,tol,verbose)
%
% Uses a stanadard path-following interior-point method to solve the
% linear matrix inequality least squares problem:
%
%   min  norm(A*x-b)
%   s.t. K(x) + S = C
%        S is symm. pos. semidef.
%
% where A is a real m-by-n matrix, x is a real n-by-1 vector,
% b is a real m-by-1 vector, S and C are real n-by-n symmetric
% matrices, and K(x) = sum( x(i)*K_i, i=1:n ) where KK = [K_1,...,K_n]
% and K_i is a real k-by-k symmetric matrix for i=1:n.
%
% S0 and Y0 are n-by-n initial strictly feasible matrices, which means
% that S0 and Y0 are symmetric positive definite.
% Set as [] for the default value of eye(n).
%
% tol is the zero tolerance described below.
% Set as [] for the default value of 1e-10.
%
% Set verbose = 1 for screen output during algorithm execution,
% otherwise set vebose = 0 for no output.
%
% LMI_LS_PRECORR returns approximate optimal solutions to the above primal
% problem and its associated dual problem so that
%
%   norm(res,'fro')  <=  sqrt(tol)*norm(res0,'fro')
%        trace(S*Y)  <=  tol*trace(S0*Y0)
%
% where res = [rp; rd] is the primal-dual residual vector
% and res0 is res evaluated at x0, S0, Y0.
%
% LMI_LS_PRECORR optionally returns:
%
%  norm_res : norm(res,'fro') at the final iterate,
%  muv  : a vector of the duality gaps for each iteration
%  tt   : the total running time of the algorithm
%  iter : the number of iterations required
%  fail : fail = 1 if the algorithm failed to achieve the desired
%         tolerances within the maximum number of iterations allowed;
```

```
%         otherwise fail = 0

% Nathan Krislock, University of British Columbia, 2003.
%
% N. Krislock. Numerical solution of semidefinite constrained least
% squares problems.  M.Sc. thesis, University of British Columbia,
% Vancouver, British Columbia, Canada, 2003.


tic;  % Start the preprocessing timer

MaxIt = 100;  % max iteration

[m,n] = size(A);  k = length(C);
AA = A'*A;  Ab = A'*b;  I = eye(k);  Ik2 = eye(k^2);
O = zeros(k^2,n); Ok2 = zeros(k^2,k^2);

K = vecK(KK);  % vec(K(x)) = K*x,  K^{*}(Y) = K'*vec(Y)

if isempty(Y0), Y = I; else, Y = Y0; end
if isempty(x0), x = zeros(n,1); else, x = x0; end
if isempty(S0), S = I; else, S = S0; end
if isempty(tol), tol = 1e-10; end

Kx = mat(K*x);  y = vec(Y);  KY = K'*y;  SY = S*Y;

z = Ab - AA*x;  rd = z - KY;
Rp = C - Kx - S;  rp = vec(Rp);

norm_res = norm([rp;rd]);  mu = trace(SY)/k;  muv = mu;

tol1 = sqrt(tol)*norm_res;  tol2 = tol*mu;

r = 0;  theta = 0;

if verbose==1
    disp(' ');
    disp([' r      sigma         theta        norm(res)     <S,Y>/n']);
    disp([' ---   ----------    ----------    ----------    ----------']);

    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('                              ') ];
    ol = [ ol, sprintf('  %12.4e',[norm_res, mu]) ];
    disp(ol);
end
```

```
pretime = toc;  % End the preprocessing timer


while ( norm_res > tol1 | mu > tol2 ) & r < MaxIt

    tic;  % Start the iteration timer


    % Compute the predictor directions dxp, dSp, dYp

    E = (kron(I,S)+kron(S,I))/2;  F = (kron(I,Y)+kron(Y,I))/2;
    D = -(SY+SY')/2;  rcp = vec(D);

    M = [ AA,  K',  O';
          K, Ok2, Ik2;
          O,  E ,  F   ];

    d = [ rd; rp; rcp ];

    [L,U,P] = lu(M);
    w = U\(L\(P*d));

    dxp = w(1:n);
    dyp = w((n+1):(k^2+n));
    dsp = w((k^2+n+1):(2*k^2+n));

    dYp = mat(dyp);  dYp = (dYp+dYp')/2;
    dSp = mat(dsp);  dSp = (dSp+dSp')/2;


    % Compute predictor step length ptheta

    c = 0.9 + 0.09*theta;

    ptheta1 = max_step(S,dSp);  ptheta2 = max_step(Y,dYp);
    ptheta_max = min([ptheta1, ptheta2]);

    ptheta = min([c*ptheta_max,1]);


    % Compute the Mehrotra sigma and parameter tau

    muhat = sum(sum((S+ptheta*dSp).*(Y+ptheta*dYp)))/k;

    sigma = (muhat/mu)^3;
    tau = sigma*mu;
```

```
% Compute the predictor-corrector directions dx, dS, dY

dSpdYp = dSp*dYp;
D = tau*I-(dSpdYp+dSpdYp')/2;  rc = vec(D);

d = [ rd; rp; rcp + rc ];

w = U\(L\(P*d));

dx = w(1:n);
dy = w((n+1):(k^2+n));
ds = w((k^2+n+1):(2*k^2+n));

dY = mat(dy);  dY = (dY+dY')/2;
dS = mat(ds);  dS = (dS+dS')/2;


% Compute the step length theta

theta1 = max_step(S,dS);  theta2 = max_step(Y,dY);
theta_max = min([theta1, theta2]);

theta = min([c*theta_max,1]);


% Update

Y = Y + theta*dY;
x = x + theta*dx;
S = S + theta*dS;

Kx = mat(K*x);  y = vec(Y);  KY = K'*y;  SY = S*Y;

z = Ab - AA*x;  rd = z - KY;
Rp = C - Kx - S;  rp = vec(Rp);

norm_res = norm([rp;rd]);  mu = trace(SY)/k;  muv(r+2) = mu;

r = r + 1;

if verbose==1
    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('  %12.4e',[sigma, theta, norm_res, mu]) ];
    disp(ol);
```

```
    end

    t(r) = toc;  % End the iteration timer

end


if r==MaxIt & ( norm_res > tol1 | mu > tol2 )
    fail = 1;
else
    fail = 0;
end

avt = mean(t);  tt = sum(t) + pretime;  iter = r;

if verbose==1
    if fail==1,
        disp(' ');
        disp('Failed to reach desired tolerance.');
    end
    disp(' ');
    disp(sprintf('Average iteration time:\t%2.4f seconds', avt));
    disp(sprintf('Total time:\t\t%2.4f seconds', tt));
    disp(' ');
end
```

B.4 Woodgate's Algorithm M-files

B.4.1 wg_sdls.m

```
function [K,norm_res,dgv,tt,iter,fail] = wg_sdls(A,B,K0,tol,verbose);
% [K,norm_res,dgv,tt,iter,fail] = WG_SDLS(A,B,K0,tol,verbose);
%
% Uses Woodgate's Algorithm to solve the symmetric semidefinite
% constrained least squares problem:
%
%   min  norm(A*K-B,'fro')
%   s.t. K symm. pos. semidef.
%
% where A and B are real m-by-n matrices, and K is a real n-by-n matrix.
%
% K0 is an n-by-n initial symmetric matrix.
% Set as [] for the default value of K0 = aleph(Kbar+), where Kbar is the
% solution of the symmetric least squares problem,
%
%   min  norm(A*K-B,'fro')
%   s.t. K = K'
%
% Kbar+ is the result of setting any negative eigenvalue in the
% eigenvalue decomposition of Kbar to zero, and
%
%   aleph(Kbar+) = alpha*Kbar+.
%
% tol is the zero tolerance described below.
% Set as [] for the default value of 1e-10.
%
% Set verbose = 1 for screen output during algorithm execution,
% otherwise set vebose = 0 for no output.
%
% WG_SDLS returns approximate optimal solutions to the above problem
% so that
%              norm(res,'fro') <= sqrt(tol)
%               trace(K*L(K))  <=  tol
%
% where L(K) = eigshift(A'*(A*K-B) + (A*K-B)'*A, 0) and
% res = L(K) - (A'*(A*K-B) + (A*K-B)'*A).
%
% WG_SDLS optionally returns:
%
%  norm_res : norm(res,'fro') at the final iterate,
%  dgv  : a vector of the duality gaps for each iteration
%  tt   : the total running time of the algorithm
```

```
%  iter : the number of iterations required
%  fail : fail = 1 if the algorithm failed to achieve the desired
%         tolerances within the maximum number of iterations allowed;
%         otherwise fail = 0

% Nathan Krislock, University of British Columbia, 2003.
%
% Based on an algorithm described in
% K. G. Woodgate.  "Efficient stiffness matrix estimation for elastic
% structures"  Computers & Structures, 69:79-84, 1998.


tic;  % Start the preprocessing timer

ItMax = 100;  % max iteration

[m,n] = size(A);
X = A';  F = B';  I = eye(n);  In2 = eye(n^2);  In2divn2 = (1/n^2)*In2;
XX = X*X';  FF = F*F';  FX = F*X';  Q = FX + FX';

% vec(E') = V*vec(E)
V = [];
for i=1:n
    W = [];
    for j=1:n
        Eij = zeros(n);
        Eij(i,j) = 1;
        W = [W; Eij];
    end
    V = [V, W];
end

if isempty(K0),
    K = lyap(XX,-Q);
    K = eigshift(K,0);
    trKKXX = trace(K*K*XX);
    if trKKXX==0
        alpha = 0;
    else
        alpha = max([0,trace(K*Q)/(2*trKKXX)]);
    end
    K = alpha*K;
else
    K = K0;
end
```

```
if isempty(tol), tol = 1e-10; end

E = real(sqrtm(K + 1e-14*I));

K = E'*E;  KXX = K*XX;
LK = KXX + KXX' - Q;

LK = eigshift(LK,0);

res = LK - KXX - KXX' + Q;  norm_res = norm(res,'fro');
dual_gap = abs(trace(K*LK)/n);  dgv = dual_gap;

tol1 = sqrt(tol);  tol2 = tol;

r = 0; t = 0;

if verbose==1
    disp(' ');
    disp(['  r      omega          alpha         norm(res)     dual_gap']);
    disp([' ---   ----------    ----------    ----------    ----------']);

    ol = sprintf('%3d',r);
    ol = [ ol, sprintf('                                ') ];
    ol = [ ol, sprintf('  %12.4e',norm_res,dual_gap) ];
    disp(ol);
end

pretime = toc;  % End the preprocessing timer


while ( norm_res > tol1 | dual_gap > tol2 ) &  r < ItMax

    tic;  % Start the iteration timer


    % Determine the direction D

    kronEXXEV = (kron(E*XX,E') + kron(E,XX*E'))*V;
    Z = kron(E*XX*E',I) + kron(K,XX) + kronEXXEV;

    M = Z + kron(I,LK);  % This is the matrix used in Woodgate's paper

    M = M + In2divn2;  % We have modified this matrix by adding
                       % (1/n^2)*kron(I,I) to avoid near sigularity
                       % and discovered it has greatly improved the
                       % convergence of this algorithm.
```

```
    w = (kronEXXEV - kron(I,Q))*vec(E');

    [L,U,P] = lu(M);
    d = U\(L\(P*w));  D = mat(d)';


    % Determine the step length omega

    OPTIONS  = optimset('fminbnd');  OPTIONS.TolX = 1e-4;
    omega = fminbnd(@wg_f,0,1,OPTIONS,FF,E,D,XX,Q);


    % Update E to aleph(E-omega*D)

    E = E - omega*D;

    K = E'*E;
    trKKXX = trace(K*K*XX);
    if trKKXX==0
        alpha = 0;
    else
        alpha = sqrt(max([0,trace(K*Q)/(2*trKKXX)]));
    end
    E = alpha*E;

    K = E'*E;  KXX = K*XX;
    LK = KXX + KXX' - Q;

    LK = eigshift(LK,0);

    res = LK - KXX - KXX' + Q;  norm_res = norm(res,'fro');
    dual_gap = abs(trace(K*LK)/n);  dgv(r+2) = dual_gap;

    r = r + 1;

    if verbose==1
        ol = sprintf('%3d',r);
        ol = [ ol, sprintf('  %12.4e',[omega, alpha, norm_res, dual_gap]) ];
        disp(ol);
    end


    t(r) = toc;  % End the iteration timer

end
```

```
if r==ItMax & ( norm_res > tol1 | dual_gap > tol2 )
    fail = 1;
else
    fail = 0;
end

avt = mean(t);  tt = sum(t) + pretime;  iter = r;

if verbose==1
    if fail==1,
        disp(' ');
        disp('Failed to reach desired tolerance.');
    end
    disp(' ');
    disp(sprintf('Average iteration time:\t%2.4f seconds', avt));
    disp(sprintf('Total time:\t\t%2.4f seconds', tt));
    disp(' ');
end
```

B.5 Miscellaneous M-files

B.5.1 vec.m

```
function v = vec(V)
% v = vec(V)
%
% Given an m-by-n matrix V, returns the corresponding
% m*n column vector v.

[m,n] = size(V);

v = [];
for i=1:n
    v = [v; V(:,i)];
end
```

B.5.2 mat.m

```
function V = mat(v,n)
% V = mat(v,n)
%
% Given an m*n column vector v, returns the corresponding
% m-by-n matrix V.  If n is not given, it is assumed that
% v is an n^2 column vector.

if nargin == 1
    n2 = length(v);
    n = sqrt(n2);
    m = n;  mn = m*n;
else
    mn = length(v);
    m = mn/n;
end

k = 1;
for i=1:m:mn
    V(:,k) = v(i:(i+m-1));
    k = k+1;
end
```

B.5.3 vecV.m

```
function V = vecV(n)
% V = vecV(n)
%
% Calculates the n^2-by-n^2 matrix V which satisfies
%
%   vec((X+X')/2) = V*vec(X)
%
% for all n-by-n matrices X.

V = [];
for i=1:n
    W = [];
    for j=1:n
        E = zeros(n);
        E(i,j) = 1;
        W = [W; E];
    end
    V = [V, W];
end
V = (eye(n^2)+V)/2;
```

B.5.4 vecK.m

```
function K = vecK(KK)
% K = vecK(KK)
%
% Calculates the k^2-by-n matrix K which satisfies
%
%   vec(K(x)) = K*x
%
% where K(x) = x(1)*K_1 + ... + x(n)*K_n,
% KK = [K_1,...,K_n], and each K_i is a k-by-k matrix.
%
% This matrix K can be described as
%
%   K = [vec(K_1),...,vec(K_n)].

[k, nk] = size(KK);  n = nk/k;

K = [];
for i=1:n
    K = [K, vec(KK(:,((i-1)*k+1):(i*k)))];
end
```

B.5.5 max_step.m

```
function theta_max = max_step(X,dX)
% theta = MAX_STEP(X,dX)
%
% Given n-by-n symmetric matrices, X and dX, where X is positive definite,
% MAX_STEP returns the largest theta_max > 0 such that
%
%   X + theta*dX
%
% is positive definite for all 0 < theta < theta_max. If X + theta*dX is
% positive definite for all theta, then theta_max = Inf.

x = max(eig(-dX,X,'chol'));

if x > 0, theta_max = 1/x; else theta_max = Inf; end
```

B.5.6 eigshift.m

```
function M = eigshift(X,lam_min)
% M = eigshift(X,lam_min)
%
% Returns the result of setting all the eigenvalues less than lam_min
% in the spectral form of X (or (X+X')/2 if X is not symmetric) to lam_min.

X = (X+X')/2;  n = length(X);

[V,D] = eig(X);

for i=1:n
    if D(i,i) < lam_min
        D(i,i) = lam_min;
    end
end

M = V*D*V';
```

B.5.7 wg_f.m

```
function f = wg_f(t,FF,E,D,XX,Q)
% f = wg_f(t,FF,E,D,XX,Q)
%
% Required for computing the omega which minimizes WG_F in WG_SDLS.

f = .5*trace(FF+((E-t*D)'*(E-t*D))^2*XX-(E-t*D)'*(E-t*D)*Q);
```