Solving Large-Scale Least Squares Semidefinite Programming by Alternating Direction Methods

He, Bingsheng; Xu, Minghua; Yuan, Xiaoming

Published in:
SIAM Journal on Matrix Analysis and Applications

DOI:
10.1137/090768813

Published: 08/02/2011

Document Version:
Publisher's PDF, also known as Version of record

SIAM J. MATRIX ANAL. APPL.
Vol. 32, No. 1, pp. 136–152

SOLVING LARGE-SCALE LEAST SQUARES SEMIDEFINITE PROGRAMMING BY ALTERNATING DIRECTION METHODS*

BINGSHENG HE†, MINGHUA XU‡, AND XIAOMING YUAN§

Abstract. The well-known least squares semidefinite programming (LSSDP) problem seeks the nearest adjustment of a given symmetric matrix in the intersection of the cone of positive semidefinite matrices and a set of linear constraints, and it captures many applications in diversing fields. The task of solving large-scale LSSDP with many linear constraints, however, is numerically challenging. This paper mainly shows the applicability of the classical alternating direction method (ADM) for solving LSSDP and convinces the efficiency of the ADM approach. We compare the ADM approach with some other existing approaches numerically, and we show the superiority of ADM for solving large-scale LSSDP.

Key words. least squares semidefinite matrix, alternating direction method, variational inequality, large-scale

AMS subject classifications. 49M29, 90C06, 90C25, 90C22

DOI. 10.1137/090768813

1. Introduction.

Let S^n be the space of $n \times n$ symmetric matrices and S^n_+ be the cone of positive semidefinite matrices in S^n. The least squares semidefinite programming (LSSDP) problem is

$$\begin{array}{ll} \min & \frac{1}{2}\|X - C\|^2 \\ \text{subject to (s.t.)} & \langle A_i, X\rangle = b_i, \quad i = 1, 2, \dots, p, \\ & \langle G_j, X\rangle \le d_j, \quad j = 1, 2, \dots, m, \\ & X \in S^n_+, \end{array} \tag{1.1}$$

where C, A_i, $i = 1, 2, \dots, p$, and G_j, $j = 1, 2, \dots, m$, are given matrices in S^n, $b = (b_1, b_2, \dots, b_p)^T \in \mathbb{R}^p$, $d = (d_1, d_2, \dots, d_m)^T \in \mathbb{R}^m$, $\langle \cdot, \cdot\rangle$ denotes the standard trace inner product in S^n where $\langle A, B\rangle = \text{trace}(A^T B)$, and $\|\cdot\|$ is the induced Frobenius norm. We refer to, e.g., [2, 13, 21, 22, 27] and references cited therein for applications of LSSDP. Note that by defining

$$S_B := \{X \in S^n | \langle A_i, X\rangle = b_i, i = 1, 2, \dots, p, \langle G_j, X\rangle \le d_j, j = 1, 2, \dots, m\}, \tag{1.2}$$

the LSSDP can be viewed as seeking the nearest adjustment of C in the intersection $S^n_+ \cap S_B$ or finding the projection of C onto the intersection $S^n_+ \cap S_B$ (the projection is well defined since both S_B and S^n_+ are convex). Throughout, we assume that the solution set of (1.1) is not empty.

The LSSDP arising in applications are usually large-scale in the sense that the dimension of variables n is large (say, $n > 1,000$) and the number of constraints $p+m$ is also large (say, $p + m > 5,000$). Then it is very challenging to develop efficient

*Received by the editors August 21, 2009; accepted for publication (in revised form) by P.-A. Absil December 28, 2010; published electronically February 8, 2011. This work was supported by NSFC grant 10971095, Hong Kong General Research Fund 203009, and Jiangsu NSF grant BK2008255.
http://www.siam.org/journals/simax/32-1/76881.html

†Department of Mathematics, Nanjing University, Nanjing, 210093, China (hebma@nju.edu.cn).
‡School of Mathematics and Physics, Changzhou University, Changzhou, 213164, China (xuminghua@cczu.edu.cn).
§Corresponding author. Department of Mathematics, Hong Kong Baptist University, Hong Kong, China (xmyuan@hkbu.edu.hk).

algorithms for solving large-scale LSSDP (1.1), and many authors have been devoted to this goal. In particular Higham [23] applied Dysktra's alternating projection method to solve the weighted nearest correlation matrix problem, which is a special case of LSSDP, and this celebrated work immediately motivated lots of work, e.g., [2, 5, 13, 26, 27, 31]. Note that LSSDP (1.1) can be reformulated into the following semidefinite programming (SDP) problem (see, e.g., [13]):

$$\begin{aligned} \min\ & t \\ \text{s.t.}\ & \langle A_i, X\rangle = b_i,\ i = 1, 2, \ldots, p, \\ & \langle G_j, X\rangle \leq d_j,\ j = 1, 2, \ldots, m, \\ & t \geq \|X - C\|, \\ & X \in S^n_+. \end{aligned}$$

Hence, popular solvers based on interior point methods such as SeDuMi [32] and SDPT3 [38] are applicable to LSSDP (see, e.g., [10, 35, 36]) for the interior point approach for solving the LSSDP problem. However, as analyzed in [2, 13], the interior point approach is not efficient for solving even medium sized LSSDP problems (e.g., $80 < n < 200$) or the number of constraints is large. The reason for this difficulty is apparent since a system of linear equations with a dense Schur complement matrix of the size $[(p + m) + 1 + \frac{1}{2}n(n + 1)] \times [(p + m) + 1 + \frac{1}{2}n(n + 1)]$ is generated and is required to solve at each iteration; see [1, 13]. In [27], the author studied LSSDP without inequality constraints by the Lagrangian dual approach and applied the BFGS method to solve the reduced convex differentiable dual problem. This Lagrangian dual approach [27] was extended to a more general LSSDP problem with both equality and inequality constraints in [2], and a projected gradient method was designed to solve the reduced dual problem therein (a Newton method was proposed to solve a special LSSDP problem). We mention that a particular plausible feature of [2] is that the optimal solution is a low rank adjustment of the given matrix C in many cases. Most recently, the authors in [13] reformulated the LSSDP problem into a system of semismooth equations with two level metric projection operators, and it was solved by an innovative inexact smoothing Newton method (ISNM). The numerical results reported in [13] reveal the promising competitiveness of the semismooth approach to solving large-scale LSSDP with simple constraints. Note that the computational cost of each single iteration of the semismooth approach in [13] mainly consists of computing a projection onto S^n_+ and solving a smoothing Newton linear system (see equation (27) in [13]) whose size is determined by the dimension of variables n and the number of constraints $p + m$.

Therefore, for solving large-scale LSSDP it is of particular desire to retain both the advantages of gradient-type methods (the complexity of one single iteration is relatively lower) and Newton-type methods (the total number of iterations is significantly smaller). In this paper we achieve this goal to some extents via implementing the classical alternating direction method (ADM) proposed originally in [12]. Note that ADM has received extensive attention in many areas such as convex programming [4, 6, 9, 25], variational inequalities [19, 37], partial differential equations [11, 12, 14], SDP [39], and image processing [7, 29]. In particular, the ADM was applied to solve the convex quadratically constrained quadratic SDP in [34] recently. As the authors of [34], we will also focus on the original ADM. However, our work differs from [34] in mainly two points: (a) along with the original ADM, we will also concentrate on the ADM proposed in [40] and not on an inexact ADM as the authors do in [34]; (b) we will show more numerical evidence of the efficiency of the ADM-type methods.

In this paper we show that ADM-type methods are readily applicable to a simple reformulation of (1.1), and the resulting ADM approach turns out to be a very simple and efficient approach to solving large-scale LSSDP. More specifically, for the ADM approach to LSSDP, the total number of iterations is moderate (much smaller than existing gradient-type methods), and its dominant computation at each iteration is to compute a projection onto S^n_+ (no system of linear equations is required to solve). We compare the ADM approach numerically with some existing approaches and show the efficiency of the ADM approach to large-scale LSSDP. The main reason of the efficiency of the ADM approach for solving the LSSDP problem is that the high-level separable structure of the ADM-oriented reformulation of the LSSDP problem (see (2.1)) is fully exploited.

Since the ADM is originated from the classical augmented Lagrangian method (ALM), we refer to some recent impressive augmented Lagrangian-based methods in the literature of SDP; e.g., [3, 28, 33, 41].

The rest of the paper is organized as follows. In section 2 we present a simple reformulation of (1.1) and then develop some ADM-type methods for solving the reformulation of (1.1). In section 3 we prove the convergence of the proposed ADM-type methods for solving (1.1). Then in section 4 we implement the ADM-type methods to solve some LSSDP tested in the literature, and we compare them numerically with some existing methods. Finally, some conclusions and some issues for future research are discussed in section 5.

2. The ADM to LSSDP.

2.1. An ADM-oriented reformulation. The LSSDP (1.1) has the following reformulation:

$$
\begin{array}{cl}
\min\limits_{X,Y} & \frac{1}{2}\|X-C\|^2 + \frac{1}{2}\|Y-C\|^2 \\
\text{s.t} & X - Y = 0, \\
 & X \in S^n_+,\ Y \in S_B,
\end{array} \tag{2.1}
$$

where S_B is defined in (1.2). Because of the convexity, (2.1) amounts to solving the following variational inequalities: find $W^* = (X^*, Y^*, Z^*) \subset \Omega = S^n_+ \times S_B \times S^n$ such that

$$
\left\{
\begin{array}{l}
\langle X - X^*,\ X^* - C - Z^* \rangle \geq 0 \ \ \forall X \in S^n_+, \\
\langle Y - Y^*,\ Y^* - C + Z^* \rangle \geq 0 \ \ \forall Y \in S_B, \\
\langle Z - Z^*,\ X^* - Y^* \rangle \geq 0 \ \ \forall Z \in S^n.
\end{array}
\right. \tag{2.2}
$$

The reformulation (2.1) is well structured and ADM-type methods are readily applicable to the reformulation of (1.1). Note that the augmented Lagrangian function of (2.1) is

$$
AL(X, Y, Z) = \frac{1}{2}\|X-C\|^2 + \frac{1}{2}\|Y-C\|^2 - \langle Z, X-Y \rangle + \frac{\beta}{2}\|X-Y\|^2,
$$

where $\beta > 0$ is the penalty parameter for the violation of constraints and $Z \in S^n$ is the Lagrangian multiplier. By applying the classical ALM (see, e.g., [20, 30]) directly to (2.1), the iterative scheme for generating the new iterate is

$$
\left\{
\begin{array}{l}
(X^{k+1}, Y^{k+1}) \in \operatorname{argmin}_{X \in S^n_+, Y \in S_B} \{AL(X, Y, Z^k)\}, \\
Z^{k+1} = Z^k - \beta(X^{k+1} - Y^{k+1}),
\end{array}
\right. \tag{2.3}
$$

where (X^k, Y^k, Z^k) is the given triple of the iterate.

Note that the objective function of (2.1) is of separable structure in the sense that it is the sum of two terms without crossed terms, while the ALM scheme (2.3) applied for (2.1) ignores this separable structure and performs the minimization task over X and Y simultaneously at each iteration. On the other hand, by splitting the minimization task in (2.3) into two smaller subproblems, the well-known ADM (see, e.g., [11, 12]) executes the minimization tasks of X and Y separably in the alternating manner:

$$\left\{ \begin{array}{l} X^{k+1} = \text{argmin}_{X \in S^n_+}\{AL(X, Y^k, Z^k)\}, \\ Y^{k+1} = \text{argmin}_{Y \in S_B}\{AL(X^{k+1}, Y, Z^k)\}, \\ Z^{k+1} = Z^k - \beta(X^{k+1} - Y^{k+1}). \end{array} \right. \tag{2.4}$$

In this sense, ADM is virtually a practical version of ALM for solving linearly constrained convex programming problems with separable objective functions such as (2.1).

2.2. Solve the subproblems of ADM. Because of the convexity of (2.4), by deriving the optimal conditions, we can see that the iterate $(X^{k+1}, Y^{k+1}, Z^{k+1})$ generated by (2.4) is characterized by the following variational inequalities and equation:

$$\left\{ \begin{array}{l} \langle X - X^{k+1},\ X^{k+1} - C - [Z^k - \beta(X^{k+1} - Y^k)]\rangle \geq 0 \ \ \forall X \in S^n_+, \\ \langle Y - Y^{k+1},\ Y^{k+1} - C + [Z^k - \beta(X^{k+1} - Y^{k+1})]\rangle \geq 0 \ \ \forall Y \in S_B, \\ Z^{k+1} = Z^k - \beta(X^{k+1} - Y^{k+1}). \end{array} \right. \tag{2.5}$$

According to the Theorem 3.1.1 in [24], the involved variational inequalities in (2.5) have analytic solutions that can be computed as follows:

$$X^{k+1} = P_{S^n_+}\left[\frac{1}{1+\beta}(\beta Y^k + Z^k + C)\right], \tag{2.6}$$

$$Y^{k+1} = P_{S_B}\left[\frac{1}{1+\beta}(\beta X^{k+1} - Z^k + C)\right], \tag{2.7}$$

where $P_{S^n_+}$ and P_{S_B} denote the projection onto S^n_+ and P_{S_B}, respectively.

Note that problem (2.5) is equivalent to the problem (2.4); thus the main computational cost for executing one iteration of ADM-type methods consists of the computation of $P_{S^n_+}$ and P_{S_B}. We now elaborate on computing these projections. First, we consider $P_{S^n_+}(X)$. Let

$$Q^T X Q = \Lambda \quad \text{with} \quad \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n) \tag{2.8}$$

be the symmetric Schur decomposition of X; $Q = (q_1, \ldots, q_n)$ be an orthogonal matrix whose column vector $q_i, i = 1, \ldots, n$, are eigenvectors of X; and $\lambda_i, i = 1, \ldots, n$, be the corresponding eigenvalues. Then it is known (see, e.g., [15, 21]) that

$$P_{S^n_+}(X) = Q\tilde{\Lambda}Q^T, \tag{2.9}$$

where

$$\tilde{\Lambda} = \text{diag}(\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n) \quad \text{with} \quad \tilde{\lambda}_i = \max(\lambda_i, 0). \tag{2.10}$$

Recall that the complexity of implementing the symmetric QR algorithm (Algorithm 8.3.3 in [16]) to compute an approximate symmetric Schur decomposition is about $9n^3$ flops.

For computing the projection $P_{S_B}(X)$, it requires solving the following standard quadratic programming problem:

$$\begin{aligned} \min_Z \ & \frac{1}{2}\|Z - X\|^2 \\ \text{s.t. } & \langle A_i, Z\rangle = b_i, i = 1, 2, \ldots, p, \\ & \langle G_j, Z\rangle \le d_j, j = 1, 2, \ldots, m. \end{aligned} \tag{2.11}$$

Many existing approaches in the literature of quadratic programming are applicable for solving (2.11); see, e.g., [8, 24]. As we will see later, for many cases of the LSSDP problem, including the cases tested in [13], the projection onto S_B can be computed easily with much lower complexity than $P_{S^n_+}$.

Hence, the computational cost of each iteration of the ADM is dominated by the computation of $P_{S^n_+}$.

2.3. ADMs to be implemented. In this paper we will implement the original ADM [11, 12] and the ADM proposed in [40] for solving the reformulation (2.1) of LSSDP. For notational convenience, from now on, we denote by $\tilde{W}^k = (\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$ the iterate generated by solving (2.4) from the given iterate $W^k = (X^k, Y^k, Z^k)$. Then we consider the following schemes to generate the new triple $W^{k+1} = (X^{k+1}, Y^{k+1}, Z^{k+1})$:

The original ADM in [11, 12].

$$\begin{cases} X^{k+1} = \tilde{X}^k, \\ Y^{k+1} = \tilde{Y}^k, \\ Z^{k+1} = \tilde{Z}^k. \end{cases} \tag{2.12}$$

The ADM proposed in [40].

$$\begin{cases} X^{k+1} & = & \tilde{X}^k, \\ \begin{pmatrix} Y^{k+1} \\ Z^{k+1} \end{pmatrix} & = & \begin{pmatrix} Y^k \\ Z^k \end{pmatrix} - \gamma\alpha_k^* \begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix}, \end{cases} \tag{2.13}$$

where $\gamma \in (0,\ 2)$ and

$$\alpha_k^* = \frac{\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 - \langle Y^k - \tilde{Y}^k,\ Z^k - \tilde{Z}^k\rangle}{\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2}. \tag{2.14}$$

Remark 1. In [14], it was proved that the step of updating the multiplier Z^k in (2.12) can be generalized into

$$Z^{k+1} = Z^k - \tau\beta(X^{k+1} - Y^{k+1}),$$

where $\tau \in (0, \frac{1+\sqrt{5}}{2})$.

2.4. Stopping criterion. Let $(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k) \in \Omega$ be the iterate generated by the basic ADM scheme (2.4). Recall that $(X^{k+1}, Y^{k+1}, Z^{k+1})$ in (2.5) is now labeled by $(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$. Thus, replacing $(X^{k+1}, Y^{k+1}, Z^{k+1})$ by $(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$, we can easily have the following reformulation of (2.5):

$$\begin{cases} \langle X - \tilde{X}^k,\ \tilde{X}^k - \tilde{Z}^k - C - \beta(Y^k - \tilde{Y}^k)\rangle \ge 0 \ \ \forall X \in S^n_+, \\ \langle Y - \tilde{Y}^k,\ \tilde{Y}^k + \tilde{Z}^k - C\rangle \ge 0 \ \ \forall Y \in S_B, \\ \langle Z - \tilde{Z}^k,\ \tilde{X}^k - \tilde{Y}^k + \frac{1}{\beta}(\tilde{Z}^k - Z^k)\rangle \ge 0 \ \ \forall Z \in S^n. \end{cases} \tag{2.15}$$

Therefore, it is clear that $(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$ is a solution of (2.2) if and only if $Y^k = \tilde{Y}^k$ and $Z^k = \tilde{Z}^k$. Hence, to implement ADM-type methods, it is reasonable to use the following stopping criterion:

$$\max\left\{\|Y^k - Y^{k+1}\|_{\max}, \|Z^k - Z^{k+1}\|_{\max}\right\} \le \varepsilon, \tag{2.16}$$

where $\varepsilon > 0$.

3. Convergence.

In this section we investigate the convergence of the proposed ADM approach to solve the LSSDP problem. Note that the convergence of the original ADM (2.12) is a straightforward conclusion of Theorem 1 in [18]. Thus, we need only to prove the convergence of the ADM proposed in [40] ((2.13)–(2.14) in this article). Since the proof of the ADM proposed in [40] was established with a general setting in the vector space $\mathcal{R}^n$, the specification of the convergence of the iterative scheme (2.13) for LSSDP is not very evident, and it deserves delineation. For completeness, we provide the proof of (2.13) as follows.

LEMMA 3.1. *Let $W^* = (X^*, Y^*, Z^*)$ be the solution of (2.2) and $\tilde{W}^k = (\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$ be generated by the ADM scheme (2.4) from the given iterate (X^k, Y^k, Z^k). Then we have*

$$\left\langle \begin{pmatrix} \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, G\begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix} \right\rangle \ge -\langle Y^k - \tilde{Y}^k, Z^k - \tilde{Z}^k \rangle, \tag{3.1}$$

where

$$G = \begin{pmatrix} \beta I_n & 0 \\ 0 & \frac{1}{\beta} I_n \end{pmatrix}. \tag{3.2}$$

Proof. Let $\Omega := S_+^n \times S_B \times S^n$ and $W := (X, Y, Z) \in \Omega$. Then it is easy to see that (2.15) can be rewritten into the following more compact form:

$$\left\langle \begin{pmatrix} X - \tilde{X}^k \\ Y - \tilde{Y}^k \\ Z - \tilde{Z}^k \end{pmatrix}, \begin{pmatrix} \tilde{X}^k - \tilde{Z}^k - C \\ \tilde{Y}^k + \tilde{Z}^k - C \\ \tilde{X}^k - \tilde{Y}^k \end{pmatrix} + \begin{pmatrix} -\beta(Y^k - \tilde{Y}^k) \\ \beta(Y^k - \tilde{Y}^k) \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ \beta(\tilde{Y}^k - Y^k) \\ \frac{1}{\beta}(\tilde{Z}^k - Z^k) \end{pmatrix} \right\rangle \ge 0$$

(3.3) $\forall W \in \Omega$.

In particular, taking $(X, Y, Z) = (X^*, Y^*, Z^*)$ in (3.3) and using the definition of G (3.2), we have

$$\left\langle \begin{pmatrix} \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, G\begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix} \right\rangle \ge$$

$$\left\langle \begin{pmatrix} \tilde{X}^k - X^* \\ \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, \begin{pmatrix} \tilde{X}^k - \tilde{Z}^k - C \\ \tilde{Y}^k + \tilde{Z}^k - C \\ \tilde{X}^k - \tilde{Y}^k \end{pmatrix} + \begin{pmatrix} -\beta(Y^k - \tilde{Y}^k) \\ \beta(Y^k - \tilde{Y}^k) \\ 0 \end{pmatrix} \right\rangle. \tag{3.4}$$

In addition, since

$$\left\langle \begin{pmatrix} \tilde{X}^k - X^* \\ \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, \begin{pmatrix} \tilde{X}^k - \tilde{Z}^k - C \\ \tilde{Y}^k + \tilde{Z}^k - C \\ \tilde{X}^k - \tilde{Y}^k \end{pmatrix} \right\rangle - \left\langle \begin{pmatrix} \tilde{X}^k - X^* \\ \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, \begin{pmatrix} X^* - Z^* - C \\ Y^* + Z^* - C \\ X^* - Y^* \end{pmatrix} \right\rangle$$

$$= \left\langle \begin{pmatrix} \tilde{X}^k - X^* \\ \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, \begin{pmatrix} \tilde{X}^k - X^* - \tilde{Z}^k + Z^* \\ \tilde{Y}^k - Y^* + \tilde{Z}^k - Z^* \\ \tilde{X}^k - X^* - \tilde{Y}^k + Y^* \end{pmatrix} \right\rangle$$

$$= \left\langle \tilde{X}^k - X^*, \tilde{X}^k - X^* \right\rangle + \left\langle \tilde{Y}^k - Y^*, \tilde{Y}^k - Y^* \right\rangle \ge 0,$$

we have

$$\left\langle \begin{pmatrix} \tilde{X}^k - X^* \\ \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, \begin{pmatrix} \tilde{X}^k - \tilde{Z}^k - C \\ \tilde{Y}^k + \tilde{Z}^k - C \\ \tilde{X}^k - \tilde{Y}^k \end{pmatrix} \right\rangle \geq \left\langle \begin{pmatrix} \tilde{X}^k - X^* \\ \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, \begin{pmatrix} X^* - Z^* - C \\ Y^* + Z^* - C \\ X^* - Y^* \end{pmatrix} \right\rangle \geq 0, \tag{3.5}$$

where the second inequality comes from (2.2) by taking $(X, Y, Z) = (\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$.

Moreover, using $X^* - Y^* = 0$ and $\tilde{Z}^k = Z^k - \beta(\tilde{X}^k - \tilde{Y}^k)$, we have

$$\begin{aligned} \left\langle \begin{pmatrix} \tilde{X}^k - X^* \\ \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, \begin{pmatrix} -\beta(Y^k - \tilde{Y}^k) \\ \beta(Y^k - \tilde{Y}^k) \\ 0 \end{pmatrix} \right\rangle &= -\beta\langle Y^k - \tilde{Y}^k, \tilde{X}^k - \tilde{Y}^k \rangle \\ &= -\langle Y^k - \tilde{Y}^k, Z^k - \tilde{Z}^k \rangle. \end{aligned} \tag{3.6}$$

Substituting (3.5) and (3.6) in (3.4), we obtain the assertion (3.1). ∎

The next lemma implies the rationale of making use of the directions $Y^k - \tilde{Y}^k$ and $Z^k - \tilde{Z}^k$ in the step (2.13).

LEMMA 3.2. *Let* $W^* = (X^*, Y^*, Z^*)$ *be the solution of* (2.2), $\tilde{W}^k = (\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$ *be generated by the ADM scheme* (2.4) *from the given iterate* (X^k, Y^k, Z^k), *and* G *be defined by* (3.2). *Then we have*

$$\begin{aligned} \left\langle \begin{pmatrix} Y^k - Y^* \\ Z^k - Z^* \end{pmatrix}, G \begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix} \right\rangle &\geq \beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 \\ &\quad -\langle Y^k - \tilde{Y}^k,\, Z^k - \tilde{Z}^k \rangle. \end{aligned} \tag{3.7}$$

Proof. Based on (3.1), we have

$$\begin{aligned} &\left\langle \begin{pmatrix} Y^k - Y^* \\ Z^k - Z^* \end{pmatrix}, G \begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix} \right\rangle = \left\langle \begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix} + \begin{pmatrix} \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, G \begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix} \right\rangle \\ &= \beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 + \left\langle \begin{pmatrix} \tilde{Y}^k - Y^* \\ \tilde{Z}^k - Z^* \end{pmatrix}, G \begin{pmatrix} Y^k - \tilde{Y}^k \\ Z^k - \tilde{Z}^k \end{pmatrix} \right\rangle \\ &\geq \beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 - \langle Y^k - \tilde{Y}^k,\, Z^k - \tilde{Z}^k \rangle, \end{aligned}$$

which is the assertion (3.7). Thus, the proof is complete. ∎

LEMMA 3.3. *We have the identity*

$$\begin{aligned} &\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 - \langle Y^k - \tilde{Y}^k,\, Z^k - \tilde{Z}^k \rangle \\ &\quad = \frac{1}{2}\left(\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 + \left\| \sqrt{\beta}(Y^k - \tilde{Y}^k) - \frac{1}{\sqrt{\beta}}(Z^k - \tilde{Z}^k) \right\|^2 \right). \end{aligned} \tag{3.8}$$

Proof. It is an obvious fact. In fact, we have

$$\begin{aligned}
&\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2-\langle Y^k-\tilde{Y}^k,\, Z^k-\tilde{Z}^k\rangle\\
&\quad=\frac{1}{2}\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2\right)\\
&\qquad+\frac{1}{2}\left(\beta\|Y^k-\tilde{Y}^k\|^2-2\langle Y^k-\tilde{Y}^k, Z^k-\tilde{Z}^k\rangle+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2\right)\\
&\quad=\frac{1}{2}\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2+\left\|\sqrt{\beta}(Y^k-\tilde{Y}^k)-\frac{1}{\sqrt{\beta}}(Z^k-\tilde{Z}^k)\right\|^2\right),
\end{aligned}$$

which is the assertion (3.8). ∎

The following theorem shows the contractive property of the sequence $\{Y^k, Z^k\}$ generated by the ADM (2.13)–(2.14), which plays a pivotal role in the convergence of this method.

THEOREM 3.4. *Let* $W^*=(X^*,Y^*,Z^*)$ *be the solution of* (2.2), $\tilde{W}^k=(\tilde{X}^k,\tilde{Y}^k,\tilde{Z}^k)$ *be generated by the ADM scheme* (2.4), $(X^{k+1},Y^{k+1},Z^{k+1})$ *be generated by the ADM* (2.13)–(2.14) *from the given iterate* (X^k,Y^k,Z^k), *and* G *be defined by* (3.2). *Then we have*

$$\begin{aligned}
\beta\|Y^{k+1}-Y^*\|^2+\frac{1}{\beta}\|Z^{k+1}-Z^*\|^2 &\le \beta\|Y^k-Y^*\|^2+\frac{1}{\beta}\|Z^k-Z^*\|^2-\frac{1}{2}\gamma(2-\gamma)\alpha_k^*\\
&\quad\times\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2\right.\\
&\quad\left.+\left\|\sqrt{\beta}(Y^k-\tilde{Y}^k)-\frac{1}{\sqrt{\beta}}(Z^k-\tilde{Z}^k)\right\|^2\right). \qquad (3.9)
\end{aligned}$$

Proof. It follows from the iterative scheme (2.13) that

$$\begin{aligned}
&\beta\|Y^{k+1}-Y^*\|^2+\frac{1}{\beta}\|Z^{k+1}-Z^*\|^2\\
&\quad=\beta\|Y^k-Y^*-\gamma\alpha_k^*(Y^k-\tilde{Y}^k)\|^2+\frac{1}{\beta}\|Z^k-Z^*-\gamma\alpha_k^*(Z^k-\tilde{Z}^k)\|^2\\
&\quad=\beta\|Y^k-Y^*\|^2+\frac{1}{\beta}\|Z^k-Z^*\|^2-2\gamma\alpha_k^*\left\langle\begin{pmatrix} Y^k-Y^*\\ Z^k-Z^*\end{pmatrix}, G\begin{pmatrix} Y^k-\tilde{Y}^k\\ Z^k-\tilde{Z}^k\end{pmatrix}\right\rangle+\\
&\qquad\gamma^2(\alpha_k^*)^2\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2\right)\\
&\quad\le\beta\|Y^k-Y^*\|^2+\frac{1}{\beta}\|Z^k-Z^*\|^2-2\gamma\alpha_k^*\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2-\right.\\
&\qquad\left.\langle Y^k-\tilde{Y}^k,\, Z^k-\tilde{Z}^k\rangle\right)+\gamma^2(\alpha_k^*)^2\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2\right)\\
&\quad=\beta\|Y^k-Y^*\|^2+\frac{1}{\beta}\|Z^k-Z^*\|^2-\gamma(2-\gamma)\alpha_k^*\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2-\right.\\
&\qquad\left.\langle Y^k-\tilde{Y}^k,\, Z^k-\tilde{Z}^k\rangle\right)\\
&\quad=\beta\|Y^k-Y^*\|^2+\frac{1}{\beta}\|Z^k-Z^*\|^2-\frac{1}{2}\gamma(2-\gamma)\alpha_k^*\left(\beta\|Y^k-\tilde{Y}^k\|^2+\frac{1}{\beta}\|Z^k-\tilde{Z}^k\|^2+\right.\\
&\qquad\left.\left\|\sqrt{\beta}(Y^k-\tilde{Y}^k)-\frac{1}{\sqrt{\beta}}(Z^k-\tilde{Z}^k)\right\|^2\right),
\end{aligned}$$

where the inequality is because of (3.7), the second to last equality follows from the definition of α_k^* (see (2.14)), and the last equivalence is due to (3.8). ∎

COROLLARY 3.5. *Let* $W^* = (X^*, Y^*, Z^*)$ *be the solution of* (2.2), $\tilde{W}^k = (\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$ *be generated by the ADM scheme* (2.4), *and* $(X^{k+1}, Y^{k+1}, Z^{k+1})$ *be generated by the ADM* (2.13)–(2.14) *from the given iterate* (X^k, Y^k, Z^k). *Then we have*

$$\begin{aligned}\beta\|Y^{k+1} - Y^*\|^2 + \frac{1}{\beta}\|Z^{k+1} - Z^*\|^2 \le \beta\|Y^k - Y^*\|^2 + \frac{1}{\beta}\|Z^k - Z^*\|^2 - \frac{1}{4}\gamma(2-\gamma)\\ \times\left(\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2\right.\\ \left. + \left\|\sqrt{\beta}(Y^k - \tilde{Y}^k) - \frac{1}{\sqrt{\beta}}(Z^k - \tilde{Z}^k)\right\|^2\right).\end{aligned} \tag{3.10}$$

Proof. From Lemma 3.3, we have

$$\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 - \langle Y^k - \tilde{Y}^k,\, Z^k - \tilde{Z}^k\rangle \ge \frac{1}{2}\left(\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2\right).$$

We thus conclude that the step size α_k^* defined by (2.14) is bounded below:

$$\alpha_k^* \ge \frac{1}{2}.$$

Hence, the assertion (3.10) follows immediately from (3.9) and the above inequality. ∎

COROLLARY 3.6. *Let* $W^* = (X^*, Y^*, Z^*)$ *be the solution of* (2.2), $\tilde{W}^k = (\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)$ *be generated by the ADM scheme* (2.4), *and* $(X^{k+1}, Y^{k+1}, Z^{k+1})$ *be generated by the ADM* (2.13)–(2.14) *from the given iterate* (X^k, Y^k, Z^k). *Then we have*

(1) *The sequence* $\{(X^{k+1}, Y^{k+1}, Z^{k+1})\}$ *is bounded.*
(2) *The sequence* $\{\beta\|Y^k - Y^*\|^2 + \frac{1}{\beta}\|Z^k - Z^*\|^2\}$ *is nonincreasing.*
(3) $\lim_{k\to\infty}(\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2) = 0.$
(4) *The sequence* $\{(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)\}$ *is bounded.*

Proof. (1) and (2) are straightforward conclusions of (3.10).

Now we prove (3). It again follows from (3.10) that

$$\begin{aligned}\beta\|Y^{k+1} - Y^*\|^2 + \frac{1}{\beta}\|Z^{k+1} - Z^*\|^2 \le \beta\|Y^0 - Y^*\|^2 + \frac{1}{\beta}\|Z^0 - Z^*\|^2 - \sum_{j=0}^{k}\frac{1}{4}\gamma(2-\gamma)\\ \times\left(\beta\|Y^j - \tilde{Y}^j\|^2 + \frac{1}{\beta}\|Z^j - \tilde{Z}^j\|^2\right).\end{aligned}$$

Therefore, we get

$$\sum_{j=0}^{k}\frac{1}{4}\gamma(2-\gamma)\left(\beta\|Y^j - \tilde{Y}^j\|^2 + \frac{1}{\beta}\|Z^j - \tilde{Z}^j\|^2\right) < \beta\|Y^0 - Y^*\|^2 + \frac{1}{\beta}\|Z^0 - Z^*\|^2 \quad \forall k > 0.$$

Thus, for any k, we have that

$$\sum_{j=0}^{k}\frac{1}{4}\gamma(2-\gamma)\left(\beta\|Y^j - \tilde{Y}^j\|^2 + \frac{1}{\beta}\|Z^j - \tilde{Z}^j\|^2\right)$$

is bounded by a constant independent of k. This means

$$\sum_{j=0}^{\infty} \frac{1}{4}\gamma(2-\gamma)\left(\beta\|Y^j - \tilde{Y}^j\|^2 + \frac{1}{\beta}\|Z^j - \tilde{Z}^j\|^2\right) < +\infty.$$

Hence, we have

$$\lim_{k\to\infty} \left(\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2\right) = 0. \tag{3.11}$$

Finally, (4) is a direct conclusion of (1) and (3.11). ∎

Now we are at the stage to prove the convergence of the ADM (2.13)–(2.14) for solving LSSDP.

THEOREM 3.7. *Let* $\{(X^{k+1}, Y^{k+1}, Z^{k+1})\}$ *be the sequence generated by the ADM* (2.13) *and* $\{(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)\}$ *be generated by the ADM scheme* (2.4)*. Then the sequence* $\{(X^{k+1}, Y^{k+1}, Z^{k+1})\}$ *converges to a solution of* (2.2)*. Hence, the corresponding sequence* $\{(X^{k+1}, Y^{k+1})\}$ *converges to the solution of* (2.1).

Proof. Since bounded (see (4) of Corollary 3.6), the sequence $\{(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)\}$ has at least one cluster point. Let $\{(X^\infty, Y^\infty, Z^\infty)\}$ be a cluster point of $\{(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)\}$ and $\{(\tilde{X}^{k_j}, \tilde{Y}^{k_j}, \tilde{Z}^{k_j})\}$ be the subsequence converging to $\{(X^\infty, Y^\infty, Z^\infty)\}$. It follows from (2.15) and (3.11) that

$$\begin{cases} \lim\limits_{j\to\infty} \langle X - \tilde{X}^{k_j}, \tilde{X}^{k_j} - C - \tilde{Z}^{k_j}\rangle \geq 0 \ \forall X \in S_+^n, \\ \lim\limits_{j\to\infty} \langle Y - \tilde{Y}^{k_j}, \tilde{Y}^{k_j} - C + \tilde{Z}^{k_j}\rangle \geq 0 \ \forall Y \in S_B, \\ \lim\limits_{j\to\infty} \langle Z - \tilde{Z}^{k_j}, \tilde{X}^{k_j} - \tilde{Y}^{k_j}\rangle \geq 0 \ \ \forall Z \in S^n. \end{cases}$$

Consequently,

$$\begin{cases} \langle X - X^\infty, X^\infty - C - Z^\infty\rangle \geq 0 \ \forall X \in S_+^n, \\ \langle Y - Y^\infty, Y^\infty - C + Z^\infty\rangle \geq 0 \ \forall Y \in S_B, \\ \langle Z - Z^\infty, X^\infty - Y^\infty\rangle \geq 0 \ \ \forall Z \in S^n. \end{cases}$$

Recall (2.2). Then $(X^\infty, Y^\infty, Z^\infty)$ is a solution of (2.2). Thus, any cluster point of $\{(\tilde{X}^k, \tilde{Y}^k, \tilde{Z}^k)\}$ is a solution of (2.2).

Furthermore, using the inequality (3.11), we have that for any given $\varepsilon > 0$, there exists an integer $k_0 > 0$ such that

$$\beta\|Y^k - \tilde{Y}^k\|^2 + \frac{1}{\beta}\|Z^k - \tilde{Z}^k\|^2 < \frac{\varepsilon}{4} \ \ \forall k \geq k_0.$$

Since $\{(\tilde{Y}^{k_j}, \tilde{Z}^{k_j})\}$ converges to $\{(Y^\infty, Z^\infty)\}$, for the ε given above, there exists $l > 0$ such that

$$\beta\|\tilde{Y}^{k_j} - Y^\infty\|^2 + \frac{1}{\beta}\|\tilde{Z}^{k_j} - Z^\infty\|^2 < \frac{\varepsilon}{4} \ \ \forall k_j \geq k_l.$$

Then it follows from (3.9) (here we use (Y^∞, Z^∞) rather than (Y^*, Z^*)) that

$$\begin{aligned} &\beta\|Y^k - Y^\infty\|^2 + \frac{1}{\beta}\|Z^k - Z^\infty\|^2 \leq \beta\|Y^{k_j} - Y^\infty\|^2 + \frac{1}{\beta}\|Z^{k_j} - Z^\infty\|^2 \\ &\quad \leq 2\left(\beta\|\tilde{Y}^{k_j} - Y^\infty\|^2 + \frac{1}{\beta}\|\tilde{Z}^{k_j} - Z^\infty\|^2\right) + 2\left(\beta\|Y^{k_j} - \tilde{Y}^{k_j}\|^2 + \frac{1}{\beta}\|Z^{k_j} - \tilde{Z}^{k_j}\|^2\right) \\ &\quad < 2\left(\frac{\varepsilon}{4} + \frac{\varepsilon}{4}\right) = \varepsilon \ \forall k > k_j \geq \max\{k_0, k_l\}. \end{aligned}$$

This implies that the sequence $\{(Y^k, Z^k)\}$ converges to $\{(Y^\infty, Z^\infty)\}$.

Note that $P_{S_+^n}$ is continuous and

$$X^{k+1} = \tilde{X}^k = P_{S_+^n}\left[\frac{1}{1+\beta}(\beta Y^k + Z^k + C)\right].$$

Thus, we have that

$$\begin{aligned}\lim_{k\to\infty} X^{k+1} &= \lim_{k\to\infty} P_{S_+^n}\left[\frac{1}{1+\beta}(\beta Y^k + Z^k + C)\right]\\ &= P_{S_+^n}\left[\frac{1}{1+\beta}(\beta Y^\infty + Z^\infty + C)\right].\end{aligned}$$

Therefore, $\{X^k\}$ is convergent. As we have shown, the subsequence $\{\tilde{X}^{k_j}\}$ of $\{\tilde{X}^k\}$ converges to X^∞ and $X^{k+1} = \tilde{X}^k$. Thus,

$$\lim_{k\to\infty} X^{k+1} = X^\infty.$$

Note the monotonicity of (2.2) and the convexity of (2.1). The proof is complete. ☐

4. Numerical experiments. In this section we implement the ADM to solve some scenarios of the LSSDP problem. We will numerically compare the ADM approach with the well-known Lagrangian approach [2], which is a gradient type-method, and the efficient semismooth approach [13], which is a Newton-type method. For implementing the methods in [2, 13], we used the codes that were downloaded from the authors' personal web sites. For the ADM-type methods all the codes were written in MATLAB 7.1. All the codes were run on a Dell Poweredge 1950 dual processor server equipped with quad core Xeon 3.0 GHz CPU and 16 GB RAM running Fedora 8 Linux.

Since the initial iterate for implementing ADM-type methods can be chosen arbitrarily, we take the initial iterate as $(X^0 = I_n, Y^0 = I_n, Z^0 = 0)$ to implement ADM-type methods. For the stopping criterion (2.16), although normalization is usually required in practical implementation (see, e.g., [23]), in our numerical experiments we use (2.16) directly without any normalization since we find by empirical experiments that the stopping criterion (2.16) with or without normalization performs equally for our tested examples.

4.1. Comparison with the Lagrangian dual approach in [2]. In this subsection we apply the original ADM (OADM) ((2.12) with $\beta = 1$) to solve the LSSDP problem tested in[2], and we compare the OADM numerically with the dual projected gradient algorithm (DPGA) developed in [2].

Recall that for the tested LSSDP problem in [2], $n = 10^4$ and the matrix C is a sparse positive semidefinite matrix with around 10^6 nonzero entries. In particular, C is generalized by $C = RR^T$, where $R \in \mathbb{R}^{n\times r}$ is a random sparse matrix with unit Gaussian nonzero entries and $r \le n$. The coefficient matrices A_i and G_j are all rank one matrices generated by $A_i = a_i a_i^T, i = 1, 2, \ldots, p$, and $G_j = g_j g_j^T, j = 1, 2, \ldots, m$, respectively, where a_i and g_j are normalized (i.e., $\|a_i\| = \|g_i\| = 1$) and normally distributed random vectors. The coefficients b_i and d_j are chosen by the manner $b_i = \eta_i a_i^T C a_i, i = 1, 2, \ldots, p$, and $d_j = \eta_j g_j^T C g_j, j = 1, 2, \ldots, m$, where η_i and η_j are random with uniform distribution on [0.5, 1.5]. In this example $p = m = 50$.

We focus on the version of the DPGA whose step size along the gradient direction at the kth direction is chosen as $\alpha_k = 4/(1 + k/100)$, as this version was reported to perform better than others with constant step sizes in [2]. To compute the projection onto P_{S_B} we implement the method in [17] to solve the variational inequality reformulation of the dual of (2.11). The stopping criterion of the DPGA is the same as in [2]; i.e.,

$$\left(\sum_{i=1}^{p}(\langle A_i, X\rangle - b_i)^2 + \sum_{j=1}^{m}(\max\{0, \langle G_j, X\rangle - d_j\})^2\right)^{1/2} \leq \varepsilon. \tag{4.1}$$

In addition to the scenario tested in [2], i.e., $n = 10^4$ and $r/n = 0.01$, we also test other scenarios, as stated in Table 1. The numbers of iterations (It.) and computing time in seconds (CT(s)) of these two methods for $\varepsilon = 10^{-5}$ are reported in Table 1.

TABLE 1
Comparison of the DPGA and the OADM.

		DPGA		OADM	
n	r	It.	CT(s)	It.	CT(s)
500	50	130	135.9	18	6.0
500	100	150	143.4	18	6.0
1,000	50	143	194.6	19	39.5
1,000	100	144	186.8	18	37.5
2,000	50	505	1,610.9	18	218.9
2,000	100	142	691.0	19	236.2
10,000	50	—	—	19	17,950.0
10,000	100	4,860	104,387.1	19	18,120.0

In the above table "—" means that the left-hand side of (4.1) is still greater than 10^{-4} when the total number of iteration exceeds $2,000$ and the CT(s) is more than 10 hours. As shown in Table 1, compared to the DPGA in [2], the OADM reduces both iterative numbers and computing time significantly for large-scale LSSDP. Moreover, for the OADM, the number of iterations is not sensitive to the size of the LSSDP.

4.2. Comparison with the semismooth approach in [13]. In this subsection we compare numerically the ADM approach with the semismooth approach for LSSDP. In particular, we will compare the numerical results of the OADM (2.12), the ADM in [40] (2.13) (YY-ADM), and the ISNM in [13] for solving some particular cases of the LSSDP problem which have been tested in [13].

The tested LSSDP problem in [13] is

$$\begin{array}{ll} \min & \frac{1}{2}\|X - C\|^2 \\ \text{s.t.} & X_{ij} = e_{ij}, (i,j) \in \mathcal{B}_e, \\ & X_{ij} \geq l_{ij}, (i,j) \in \mathcal{B}_l, \\ & X_{ij} \leq u_{ij}, (i,j) \in \mathcal{B}_u, \\ & X \in S_+^n, \end{array} \tag{4.2}$$

where $\mathcal{B}_e, \mathcal{B}_l, \mathcal{B}_u$ are subsets of $\{(i,j)|1 \leq i,j \leq n\}$ denoting the indexes of such entries of X that are constrained by equality, lower bounds, and upper bounds, respectively. As in [13], we assume that these index sets satisfy $\mathcal{B}_e \cap \mathcal{B}_l = \emptyset, \mathcal{B}_e \cap \mathcal{B}_u = \emptyset$, and $l_{ij} < u_{ij}$ for any $(i,j) \in \mathcal{B}_l \cap \mathcal{B}_u$. Considering that the LSSDP captures the concrete application of the so-called covariance matrix calibration problem (e.g., [2, 23]), the index set $\mathcal{B}_e$ usually includes the subset $\{(i,i)|1 \leq i \leq n\}$ with $X_{ii} = 1$. Thus we can denote by $\mathcal{B}_e = \{(i,i)|1 \leq i \leq n\} \cup \mathcal{B}_h$, where $\mathcal{B}_h = \{(i,j)|1 \leq i < j \leq n\}$.

Since the matrix C under consideration is symmetric, we can further assume that $\mathcal{B}_h, \mathcal{B}_l, \mathcal{B}_u$ are subsets of $\{(i,j)|1 \le i < j \le n\}$. According to the rows of X, we can partition the sets $\mathcal{B}_h, \mathcal{B}_l, \mathcal{B}_u$ into n disjoined subsets: $\mathcal{B}_h = \cup_{k=1}^n \mathcal{B}_h^k$, $\mathcal{B}_l = \cup_{k=1}^n \mathcal{B}_l^k$, $\mathcal{B}_u = \cup_{k=1}^n \mathcal{B}_u^k$, where the subsets $\mathcal{B}_h^k, \mathcal{B}_l^k, \mathcal{B}_u^k$ represent the indexes of such right off-diagonal entries that are constrained by equality, lower bounds, and upper bounds, respectively. We here reiterate the methodology in [13] to determine the sets $\mathcal{B}_h, \mathcal{B}_l, \mathcal{B}_u$ in the following numerical experiments. Let X be an n-dimensional matrix and h, l, u be given nonnegative integer numbers. For the kth row of X, we determine the cardinalities of $\mathcal{B}_h^k, \mathcal{B}_l^k, \mathcal{B}_u^k$ by $\min(h, n-k)$, $\min(l, n-k)$, and $\min(u, n-k)$, respectively; the indexes $(k,j) \in \mathcal{B}_h^k, \mathcal{B}_l^k, \mathcal{B}_u^k$ are determined by the same random manner as in [13].

Since X is constrained componentwise only by box constraints, the projections onto S_B arising in the second steps of ADMs are trivial. In fact, we have

$$(P_{S_B}[X])_{ij} = \begin{cases} e_{ij}, & (i,j) \in \mathcal{B}_e, \\ \max\{X_{ij}, l_{ij}\}, & (i,j) \in \mathcal{B}_l, (i,j) \notin \mathcal{B}_u, \\ \min\{X_{ij}, u_{ij}\}, & (i,j) \in \mathcal{B}_u, (i,j) \notin \mathcal{B}_l, \\ \max\{\min\{X_{ij}, u_{ij}\}, l_{ij}\}, & (i,j) \in \mathcal{B}_l \cap \mathcal{B}_u, \\ X_{ij}, & \text{otherwise.} \end{cases}$$

For this example we take $\beta = 15$ in the implementation of both OADM and YY-ADM and $\gamma = 1.5$ in the implementation of YY-ADM. For the implementation of OADM, YY-ADM, and ISNM, an MEX interface (in the C programming language) provided by the authors of [13] is used to accelerate the eigenvalue decomposition (which is faster than the MATLAB built-in function "eig.m" by about two times when $n \ge 1000$).

We first test Example 5.4 in [13]. Let $\bar{C}$ be the 387×387 1-day correlation matrix $\bar{C}$ (as of June 15, 2006) from the lagged datasets of RiskMetrics. Then the matrix C is a perturbation of $\bar{C}$: $C = (1-\alpha)\bar{C} + \alpha R$, where $\alpha \in (0,1)$ and R is a randomly generated symmetric matrix with entries in $[-1,\ 1]$. We test the scenario where $\alpha = 0.1$. Let $e_{ii} = 1$ for $i = 1, 2, \ldots, n$, $e_{ij} = 0$ for $(i,j) \in \mathcal{B}_h$, $l_{ij} = -0.1$ for $(i,j) \in \mathcal{B}_l$, and $u_{ij} = 0.1$ for $(i,j) \in \mathcal{B}_u$. For different values of ε, n, h, l, and u we report the numbers of iterations (It.) and CT(s) of ISNM, OADM, and YY-ADM in Table 2. Since the main computation of each iteration of ISNM consists of a projection onto S_+^n and solving a smoothing Newton linear system by the BiCGStab iterative solver, we report separately the computing time of these two tasks (denoted by t_1 and t_2, respectively) in addition to the total time (CT(s)). Similarly, as the complexity of each iteration of ADM-type methods is dominated by a projection onto S_+^n, we also report the aggregate time of computing $P_{S_+^n}$ for OADM and YY-ADM (denoted by t_1 in the corresponding columns) in Table 2.

We then test Example 5.6 in [13]. More specifically, in (4.2), the symmetric matrix C is generated by $C_{ii} = 1$ $(i = 1, 2, \cdots n)$ and C_{ij} $(i \ne j \in \{1, 2, \cdots, n\})$ are randomly generated in $[-1,\ 1]$. The constraints are determined exactly as the last example. We test the scenarios of $n = 500$ and $1,000$. In addition to the cases in [13], that is, $h = 0$, $l = u = 1, 5$, and 10, we also tested the scenarios stated in Tables 3–4.

When the eigenvalue decomposition (2.8) is completed, we can easily identify the zero diagonal entries of the diagonal matrix $\tilde{\Lambda}$. Then the total number of multiplications in (2.9) can be reduced accordingly, as only those columns of Q corresponding to the positive diagonal entries of $\tilde{\Lambda}$ need to be multiplied. For this example, we adopt this technique in the implementation of ADM-type methods, and we find by empirical experiments that this technique saves the computation time of ADM-type methods

Table 2
Comparison of ISNM, OADM, and YY-ADM for Example 5.4 *in* [13].

Method			ISNM			OADM		YY-ADM	
ε	h	$l = u$	It.	CT(s)	(t_1, t_2)	It.	CT(s) (t_1)	It.	CT(s) (t_1)
1e-4	0	1	7	3.5	(0.5 , 2.4)	103	8.7 (7.5)	68	6.2 (5.0)
	0	2	8	4.3	(0.6 , 3.1)	106	9.3 (8.0)	77	7.4 (5.7)
	0	5	9	4.5	(0.7 , 3.2)	104	9.4 (8.2)	69	6.9 (5.4)
	0	10	9	4.4	(0.7 , 3.1)	90	8.2 (7.1)	66	6.1 (4.9)
	0	20	9	5.7	(0.7 , 4.3)	82	7.3 (6.2)	60	6.0 (4.6)
	0	50	11	11.4	(1.0 , 9.3)	58	5.4 (4.5)	44	4.4 (3.4)
	0	100	11	20.6	(1.0 , 18.0)	46	4.9 (3.9)	32	3.8 (2.7)
	0	200	13	30.5	(1.6 , 25.9)	45	5.0 (3.7)	30	3.6 (2.4)
	0	300	10	16.6	(0.9 , 13.5)	43	4.6 (3.3)	29	3.4 (2.2)
	0	350	10	24.4	(0.9 , 21.4)	47	4.9 (3.5)	32	3.6 (2.4)
	5	350	15	48.8	(3.0 , 39.5)	44	4.9 (3.4)	30	3.5 (2.3)
	10	350	17	70.4	(4.4 , 57.4)	43	4.7 (3.3)	29	3.5 (2.3)
	20	350	18	89.2	(4.6 , 75.2)	43	4.8 (3.4)	29	3.5 (2.3)
	30	350	19	110.4	(4.6 , 96.1)	61	6.8 (4.8)	42	5.0 (3.3)
1e-6	0	1	8	4.0	(0.6 , 2.9)	165	13.8 (12.0)	106	9.6 (7.7)
	0	2	9	4.9	(0.7 , 3.6)	168	14.7 (12.0)	112	10.7 (8.3)
	0	5	9	4.5	(0.7 , 3.2)	151	13.6 (11.8)	101	10.1 (7.9)
	0	10	10	5.1	(0.8 , 3.7)	140	12.7 (11.0)	97	8.9 (7.2)
	0	20	10	6.7	(0.8 , 5.1)	122	10.9 (9.2)	88	8.8 (6.8)
	0	50	12	12.8	(1.0 , 10.6)	90	8.4 (7.0)	68	6.8 (5.2)
	0	100	12	23.5	(1.1 , 20.6)	83	8.9 (7.1)	53	6.2 (4.5)
	0	200	14	34.8	(1.7 , 30.0)	80	8.9 (6.6)	51	6.1 (4.2)
	0	300	11	18.6	(1.0 , 15.2)	79	8.6 (6.1)	51	5.9 (4.0)
	0	350	11	25.9	(1.0 , 22.7)	84	8.8 (6.3)	54	6.1 (4.1)
	5	350	16	50.7	(3.1 , 41.2)	79	8.7 (6.1)	51	6.1 (4.0)
	10	350	18	72.9	(4.4 , 59.6)	79	8.6 (6.1)	51	6.5 (4.2)
	20	350	19	91.8	(4.7 , 77.6)	77	8.6 (6.1)	50	6.1 (4.0)
	30	350	19	110.4	(4.6 , 96.1)	140	15.6 (11.1)	93	11.3 (7.4)

slightly by about $0.5\% \sim 3\%$. Note that this technique is not used for ISNM, as we want to run the original code written by the authors of ISNM.

As in Table 2, in the column "ISNM" t_1 and t_2 represent the aggregate time of computing $P_{S_+^n}$ and solving the smoothing Newton linear systems by the BiCGStab iterative solver, respectively, while t_1 in the columns of "OADM" and "YY-ADM" represents the aggregate time of computing $P_{S_+^n}$.

Tables 2–4 show that both the semismooth approach [13] and the proposed ADM approach are very efficient for solving large-scale LSSDP problems with simple constraints. For the cases with few constraints, the semismooth approach is faster than the ADM approach, while for the cases with many constraints, the ADM approach is faster. As we can see in Tables 2–4, when the number of constraints is large, the computational cost of each iteration of the semismooth approach is dominated by the time for solving the involved system of linear equations (i.e., t_2), while the time for the involved SVD is significantly less (i.e., t_1). On the opposite, at each iteration, the main computation of the ADM approach is only for executing one SVD. Thus, for the cases with many constraints, the total iteration time of the ADM approach is much less than that of the semismooth approach even if its numbers of iterations are larger. In addition, another interesting observation is that the ADM approach seems not sensitive to the numbers of constraints. Even if the number of constraints is increased greatly, the iterative time varies very slightly. This feature also makes the ADM approach promising in practice.

Table 3
Comparison of ISNM, OADM, and YY-ADM for Example 5.6 *in* [13], $n = 500$.

Method			ISNM			OADM		YY-ADM	
ε	h	$l = u$	It.	CT(s)	(t_1, t_2)	It.	CT(s) (t_1)	It.	CT(s) (t_1)
1e-4	0	1	7	2.6	(1.1 , 0.8)	49	7.5 (6.5)	45	7.5(5.9)
	0	5	9	3.6	(1.4 , 1.3)	48	7.3 (6.4)	40	6.7(5.3)
	0	10	10	5.0	(1.8 , 2.0)	46	8.8 (7.7)	40	8.9(6.9)
	10	20	11	7.5	(1.8 , 4.5)	43	6.8 (5.8)	32	5.6(4.3)
	10	50	12	12.9	(2.2 , 8.9)	40	6.6 (5.6)	28	5.0(3.9)
	50	50	12	29.2	(3.5 , 22.8)	34	6.9 (5.8)	24	5.8(4.5)
	100	100	12	55.4	(2.8 , 48.4)	34	8.0 (6.6)	24	5.7(4.3)
	100	200	13	93.8	(4.1 , 83.1)	35	8.3 (6.6)	24	6.3(4.6)
	100	300	17	125.2	(5.2 , 110.3)	35	8.4 (6.5)	24	6.2(4.5)
	100	395	17	119.7	(4.3 , 106.3)	36	7.7 (5.6)	25	6.0(4.0)
1e-6	0	1	8	2.9	(1.2 , 0.9)	85	12.9(11.2)	68	11.3(8.9)
	0	5	10	3.9	(1.6 , 1.4)	84	12.8(11.1)	62	10.3(8.1)
	0	10	11	5.5	(2.0 , 2.2)	83	15.7(13.6)	61	13.4(10.5)
	10	20	11	7.5	(1.8 , 4.5)	78	12.3(10.5)	50	8.7(6.7)
	10	50	13	14.0	(2.4 , 9.7)	74	12.1(10.2)	47	8.3(6.5)
	50	50	13	32.0	(3.6 , 25.2)	64	13.5(11.4)	41	9.8(7.6)
	100	100	13	61.1	(2.9 , 54.2)	60	14.3(11.8)	40	9.7(7.3)
	100	200	14	101.8	(4.3 , 90.6)	64	15.5(12.4)	42	10.8(7.9)
	100	300	18	133.5	(5.4 , 118.1)	65	16.1(12.5)	43	11.2(8.0)
	100	395	17	119.7	(4.3 , 106.3)	68	14.6(10.7)	45	10.6(7.1)

Table 4
Comparison of ISNM, OADM, and YY-ADM for Example 5.6 *in* [13], $n = 1,000$.

Method			ISNM			OADM		YY-ADM	
ε	h	$l = u$	It.	CT(s)	(t_1, t_2)	It.	CT(s) (t_1)	It.	CT(s) (t_1)
1e-4	0	1	9	17.5	(9.4 , 4.8)	54	47.8 (43.7)	50	49.7 (41.6)
	0	5	10	25.3	(12.4 , 7.7)	47	51.5 (47.2)	50	57.4 (47.9)
	0	10	10	24.9	(12.0 , 7.6)	53	65.6 (60.7)	50	66.4 (55.6)
	10	20	11	30.4	(11.2 , 14.3)	44	38.2 (34.7)	38	35.3 (30.0)
	10	50	12	43.7	(13.7 , 23.8)	43	40.4 (36.5)	36	36.0 (30.2)
	50	50	12	72.2	(13.9 , 51.2)	38	36.1 (32.2)	30	30.5 (25.5)
	100	100	14	164.7	(17.9 , 135.6)	38	38.9 (34.2)	26	26.9 (22.1)
	100	200	18	290.2	(27.1 , 242.6)	38	37.8 (32.6)	26	27.4 (22.2)
	100	500	19	631.2	(40.7 , 545.8)	39	57.2 (48.1)	27	40.4 (31.2)
	100	800	21	795.6	(47.2 , 686.9)	40	58.9 (48.6)	27	41.4 (31.4)
1e-6	0	1	10	19.1	(9.7 , 5.3)	84	74.2 (67.8)	76	74.6 (62.5)
	0	5	10	25.3	(12.4 , 7.7)	83	93.3 (85.7)	75	89.2 (74.4)
	0	10	10	24.9	(12.0 , 7.6)	83	101.8 (94.2)	75	97.6 (81.5)
	10	20	12	32.5	(12.0 , 15.4)	79	68.2 (61.9)	60	56.1 (47.4)
	10	50	13	46.7	(14.5 , 25.6)	76	70.3 (63.6)	50	49.8 (41.7)
	50	50	13	81.1	(14.8 , 58.8)	67	63.4 (56.6)	43	43.6 (36.5)
	100	100	15	179.2	(18.6 , 148.8)	65	65.2 (57.3)	44	45.4 (37.3)
	100	200	19	308.0	(27.9 , 258.8)	67	66.5 (57.4)	44	46.5 (37.6)
	100	500	20	672.1	(41.8 , 584.2)	71	104.3 (87.8)	47	70.7 (54.4)
	100	800	22	843.4	(48.3 , 731.9)	73	105.4 (87.0)	48	74.1 (56.2)

Nevertheless, the proposed ADM approach falls into the category of first-order methods, and it might be less efficient than second-order methods when the iterate enters into a sufficiently small neighborhood of the solution. As shown in Tables 2–4, for the cases with many constraints, the superiority of the ADM approach to the semismooth approach is less apparent when the accuracy is increased. An interesting work is that we can employ the proposed ADM approach to generate a good approximation with low or medium accuracy and then switch to the semismooth approach [13] to achieve a solution of higher accuracy. This hybrid strategy is expected to solve the LSSDP problem very efficiently.

5. Conclusions. This paper mainly studies the application of the ADMs for solving LSSDP problems. It is shown that the ADM is a simple and efficient method for solving large-scale LSSDP problems. As a gradient-type method, the ADM usually requires more iterations than Newton-type methods to achieve the same accuracy. The significantly lower complexity of each iteration, however, enables the ADM to be very promising for solving large-scale LSSDP problems. The essential reasons of the efficiency of the ADM for LSSDP are that the high-level separable structure of a reformulation of LSSDP is fully exploited and that the generated subproblems of each iteration are solved in the Gauss–Seidel fashion (note that the newly obtained x^{k+1} is used in the solution of y^{k+1} in (2.4)).

REFERENCES

[1] B. Borchers and J. G. Yong, *Implementation of a primal-dual method for SDP on a shared memory parallel architecture*, Comput. Optim. Appl., 37 (2007), pp. 355–369.

[2] S. Boyd and L. Xiao, *Least-squares covariance matrix adjustment*, SIAM J. Matrix Anal. Appl., 27 (2005), pp. 532–546.

[3] S. Burer and D. Vandenbussche, *Solving lift-and-project relaxations of binary integer programs*, SIAM J. Optim., 16 (2006), pp. 726–750.

[4] G. Chen and M. Teboulle, *A proximal-based decomposition method for convex minimization problems*, Math. Program., 64 (1994), pp. 81–101.

[5] I. S. Dhillon and J. A. Tropp, *Matrix nearness problems with Bregman divergences*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 1120–1146.

[6] J. Eckstein and M. Fukushima, *Some reformulations and applications of the alternating directions method of multipliers*, in: Large Scale Optimization: State of the Art, W. W. Hager, D. W. Hearn, and P. M. Pardalos. eds., Kluwer Academic Publishers, Norwell, MA, 1994, pp. 115–134.

[7] E. Esser, *Applications of Lagrangian-based Alternating Direction Methods and Connections to Split Bregman*, manuscript, ftp://ftp.math.ucla.edu/pub/camreport/cam09-31.pdf (2009).

[8] R. Fletcher, *Practical Methods of Optimization*, 2nd ed., John Wiley & Sons, New York, 1987.

[9] M. Fukushima, *Application of the alternating direction method of multipliers to separable convex programming problems*, Comput. Optim. Appl., 1 (1992), pp. 93–111.

[10] T. Fushiki, *Estimation of positive semidefinite correlation matrices by using convex quadratic semidefinite programming*, Neural Comput., 21 (2009), pp. 2028–2048.

[11] D. Gabay, *Application of the method of multipliers to variational inequalities*, in Augmented Lagrangian Methods: Application to the Numerical Solution of Boundary-value Problems, M. Fortin and R. Glowinski, eds., North-Holland, Amsterdam, The Netherlands, 1983, pp. 299–331.

[12] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximations*, Comput. Math. Appl., 2 (1976), pp. 17–40.

[13] Y. Gao and D.F. Sun, *Calibrating least squares semidefinite programming with equality and inequality constraints*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1432–1457.

[14] R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*, Springer, New York, 1984.

[15] W. Glunt, *An alternating projections method for certain linear problems in a Hilbert space*, IMA. J. Numer. Anal., 15 (1995), pp. 291–305.

[16] G. H. Golub and C. F. van Loan, *Matrix Computation*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.

[17] B. S. He, *Solving a class of linear projection equations*, Numer. Math, 68 (1994), pp. 71–80.

[18] B. S. He and H. Yang, *Some convergence properties of a method of multipliers for linearly constrained monotone variational inequalities*, Oper. Res. Lett., 23 (1998), pp. 151–161.

[19] B. S. He, L. Z. Liao, D. Han, and H. Yang, *A new inexact alternating directions method for monotone variational inequalities*, Math. Program., 92 (2002), pp. 103–118.

[20] M. R. Hestenes, *Multiplier and gradient methods*, J. Optim. Theory Appli., 4 (1969), pp. 303–320.

[21] N. J. Higham, *Computing a nearest symmetric positive semidefinite matrix*, Linear Algebra Appl., 103 (1988), pp. 103–118.

[22] N. J. Higham, *Matrix nearness problems and applications*, in Applications of Matrix Theory, M. Gover and S. Barnett, eds., Oxford University Press, Oxford, 1989, pp. 1–27.

[23] N. J. Higham, *Computing the nearest correlation matrix—a problem from finance*, IMA J. Numer. Anal., 22 (2002), pp. 329–343.

[24] J. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms*, Springer, New York, 1993.

[25] S. Kontogiorgis and R. R. Meyer, *A variable-penalty alternating directions method for convex optimization*, Math. Program., 83 (1998), pp. 29–53.

[26] F. Lin and M. R. Jovanović, *Least-squares approximation of structured covariances*, IEEE Transactions on Automat. Conrol., 54 (2009), pp. 1643–1648.

[27] J. Malick, *A dual approach to semidefinite least-squares problems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 272–284.

[28] J. Malick, J. Povh, F. Rendl, and A. Wiegele, *Regularization methods for semidefinite programming*, SIAM J. Optim., 20 (2009), pp. 336–356.

[29] M. K. Ng, P. A. Weiss, and X. M. Yuan, *Solving constrained total-variation image restoration and reconstruction problems via alternating direction methods*, SIAM J. Sci. Comput., 32 (2010), pp. 2710–2736.

[30] M. J. D. Powell, *A method for nonlinear constraints in minimization problems*, in Optimization: Symposium on the Institute of Mathematics and Its Applications, University of Keele, England, 1968, R. Fletcher, ed., Academic Press, New York, 1969, pp. 283–298.

[31] H. Qi and D. Sun, *A quadratically convergent Newton method for computing the nearest correlation matrix*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 360–385.

[32] J. F. Sturm, *Using SeDuMi* 1.02*, a Matlab toolbox for optimization over symmetric cones*, Optim. Methods Softw., 11–12 (1999), pp. 625–653.

[33] D. F. Sun, J. Sun, and L. W. Zhang *The rate of convergence of the augmented Lagrangian method for nonlinear semidefinite programming*, Math. Program., 114 (2008), pp. 349–391.

[34] J. Sun and S. Zhang, *A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs*, European J. Oper. Res., accepted.

[35] K. C. Toh, *An inexact primal-dual path following algorithm for convex quadratic SDP*, Math. Program., 112 (2008), pp. 221–254.

[36] K. C. Toh, R. H. Tütüncü, and M. J. Todd, *Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems*, Pacific J. Optim., 3 (2007), pp. 135–164.

[37] P. Tseng, *Alternating projection-proximal methods for convex programming and variational inequalities*, SIAM J. Optim., 7 (1997), pp. 951–965.

[38] R. H. Tütüncü, K. C. Toh, and M. J. Todd, *Solving semidefinite-quadratic-linear programs using SDPT*3, Math. Program., 95 (2003), pp. 189–217.

[39] Z. W. Wen, D. Goldfarb, and W. T. Yin, *Alternating direction augmented Lagrangian methods for semidefinite programming*, Math. Program. Comput., accepted.

[40] C. H. Ye and X. M. Yuan, *A descent method for structured monotone variational inequalities*, Optim. Methods Softw., 22 (2007), pp. 329–338.

[41] X. Y. Zhao, D. F. Sun, and K. C. Toh, *A Newton-CG augmented Lagrangian method for semidefinite programming*, SIAM J. Optim., 20 (2010), pp. 1737–1765.