A Newton-CG Augmented Lagrangian Method for Semidefinite Programming

Citation	Zhao, Xin-Yuan, Defeng Sun, and Kim-Chuan Toh. "A Newton-CG Augmented Lagrangian Method for Semidefinite Programming." SIAM J. Optim. Volume 20, Issue 4, pp. 1737-1765 (2010) ©2010 Society for Industrial and Applied Mathematics.
As Published	http://dx.doi.org/10.1137/080718206
Publisher	Society for Industrial and Applied Mathematics
Version	Final published version
Citable link	http://hdl.handle.net/1721.1/58308
Terms of Use	Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.

SIAM J. OPTIM.
Vol. 20, No. 4, pp. 1737–1765

A NEWTON-CG AUGMENTED LAGRANGIAN METHOD FOR SEMIDEFINITE PROGRAMMING*

XIN-YUAN ZHAO†, DEFENG SUN‡, AND KIM-CHUAN TOH§

Abstract. We consider a Newton-CG augmented Lagrangian method for solving semidefinite programming (SDP) problems from the perspective of approximate semismooth Newton methods. In order to analyze the rate of convergence of our proposed method, we characterize the Lipschitz continuity of the corresponding solution mapping at the origin. For the inner problems, we show that the positive definiteness of the generalized Hessian of the objective function in these inner problems, a key property for ensuring the efficiency of using an inexact semismooth Newton-CG method to solve the inner problems, is equivalent to the constraint nondegeneracy of the corresponding dual problems. Numerical experiments on a variety of large-scale SDP problems with the matrix dimension n up to $4,110$ and the number of equality constraints m up to $2,156,544$ show that the proposed method is very efficient. We are also able to solve the SDP problem `fap36` (with $n = 4,110$ and $m = 1,154,467$) in the Seventh DIMACS Implementation Challenge much more accurately than in previous attempts.

Key words. semidefinite programming, augmented Lagrangian, semismoothness, Newton method, iterative solver

AMS subject classifications. 90C06, 90C22, 90C25, 65F10

DOI. 10.1137/080718206

1. Introduction.

Let $\mathcal{S}^n$ be the linear space of all $n \times n$ symmetric matrices and $\mathcal{S}^n_+$ be the cone of all $n \times n$ symmetric positive semidefinite matrices. The notation $X \succeq \mathbf{0}$ means that X is a symmetric positive semidefinite matrix. This paper is devoted to studying an augmented Lagrangian method for solving the following semidefinite programming (SDP) problem:

$$(D) \quad \min \Big\{ b^{\mathrm{T}} y \ \big| \ \mathcal{A}^* y - C \succeq \mathbf{0} \Big\},$$

where $C \in \mathcal{S}^n$, $b \in \Re^m$, $\mathcal{A}$ is a linear operator from $\mathcal{S}^n$ to $\Re^m$, and $\mathcal{A}^* : \Re^m \to \mathcal{S}^n$ is the adjoint of $\mathcal{A}$. The dual of (D) takes the form

$$(P) \quad \max \Big\{ \langle C, X \rangle \ \big| \ \mathcal{A}(X) = b, \quad X \succeq \mathbf{0} \Big\}.$$

Given a penalty parameter $\sigma > 0$, the *augmented Lagrangian* function for problem (D) is defined as

$$(1) \quad L_\sigma(y, X) = b^{\mathrm{T}} y + \frac{1}{2\sigma} (\|\Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^* y - C))\|^2 - \|X\|^2), \ (y, X) \in \Re^m \times \mathcal{S}^n,$$

*Received by the editors March 12, 2008; accepted for publication (in revised form) October 30, 2009; published electronically January 27, 2010. A preliminary version of this paper was presented at the Second International Conference on Continuous Optimization (ICCOPT II), Hamilton, Canada, 2007.

http://www.siam.org/journals/siopt/20-4/71820.html

†Department of Applied Mathematics, Beijing University of Technology, 100 Pingleyuan, Chaoyang District, Beijing 100022, People's Republic of China (xy_zhao@hotmail.com).

‡Department of Mathematics and Risk Management Institute, National University of Singapore, 2 Science Drive 2, Singapore 117543 (matsundf@nus.edu.sg). This author's research was supported in part by Academic Research Fund under grant R-146-000-104-112.

§Department of Mathematics, National University of Singapore, 2 Science Drive 2, Singapore 117543 (mattohkc@nus.edu.sg), and Singapore-MIT Alliance, 4 Engineering Drive 3, Singapore 117576.

where for any closed convex set $\mathcal{D}$ in a finite dimensional real vector space $\mathcal{X}$ equipped with a scalar inner product $\langle \cdot, \cdot \rangle$ and its induced norm $\| \cdot \|$, $\Pi_{\mathcal{D}}(\cdot)$ is the metric projection operator over $\mathcal{D}$; i.e., for any $Y \in \mathcal{X}$, $\Pi_{\mathcal{D}}(Y)$ is the unique optimal solution to the convex optimization problem

$$\min \left\{ \frac{1}{2} \|Z - Y\|^2 \ \mid \ Z \in \mathcal{D} \right\}.$$

Note that, since $\|\Pi_{\mathcal{D}}(\cdot)\|^2$ is continuously differentiable [44], the augmented Lagrangian function defined in (1) is continuously differentiable. In particular, for any given $X \in \mathcal{S}^n$, we have

$$\nabla_y L_\sigma(y, X) = b - \mathcal{A}\Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^* y - C)). \tag{2}$$

For given $X^0 \in \mathcal{S}^n$, $\sigma_0 > 0$, and $\rho > 1$, the augmented Lagrangian method for solving problem (D) and its dual (P) generates sequences $\{y^k\} \subset \Re^m$ and $\{X^k\} \subset \mathcal{S}^n$ as follows:

$$\left\{ \begin{array}{l} y^{k+1} \approx \arg\min\limits_{y \in \Re^m} L_{\sigma_k}(y, X^k), \\ X^{k+1} = \Pi_{\mathcal{S}^n_+}(X^k - \sigma_k(\mathcal{A}^* y^{k+1} - C)), \qquad k = 0, 1, 2, \ldots, \\ \sigma_{k+1} = \rho\sigma_k \text{ or } \sigma_{k+1} = \sigma_k. \end{array} \right. \tag{3}$$

For a general discussion on the augmented Lagrangian method for solving convex optimization problems and beyond, see [32, 33].

For small and medium sized SDP problems, it is widely accepted that interior-point methods (IPMs) with direct solvers are generally very efficient and robust. For large-scale SDP problems with m large and n moderate (say, less than $5,000$), the limitations of IPMs with direct solvers become very severe due to the need for computing, storing, and factorizing the $m \times m$ Schur complement matrix. In order to alleviate these difficulties, Toh and Kojima [41] and Toh [39, 40] proposed inexact IPMs using an iterative solver to compute the search direction at each iteration. The approach in [39] was demonstrated to be able to solve large sparse SDP problems with m up to $125,000$ in a few hours. Kočvara and Stingl [17] used a modified barrier method (a variant of the Lagrangian method) combined with iterative solvers for linear SDP problems having only inequality constraints, and they reported computational results in the code PENNON [16] with m up to $125,000$. More recently, Malick et al. [19] applied the Moreau–Yosida regularization approaches to solve SDP problems, and Jarre and Rendl [13] proposed an augmented primal-dual method for solving linear conic programs including SDP problems.

In this paper, we study an augmented Lagrangian dual approach to solving large-scale SDP problems with m large (say, up to a few million) but n moderate (say, up to $5,000$). Our approach is similar in spirit to those in [17] and [19], where the idea of augmented Lagrangian methods (or methods of multipliers in general) was heavily exploited. However, our point of view of employing the augmented Lagrangian methods is fundamentally different from [17] and [19] in solving both the outer and inner problems. It has long been known that the augmented Lagrangian method for convex problems is a gradient ascent method applied to the corresponding dual problems [30]. This inevitably leads to the impression that the augmented Lagrangian method for solving SDP problems may converge slowly for the outer iteration sequence $\{X^k\}$. In spite of that, under mild conditions, a linear rate of convergence is available (superlinear convergence is also possible when σ_k goes to infinity, which should be avoided in numerical implementations) [33]. However, recent studies conducted by Sun, Sun, and Zhang [37] and Chan and Sun [7] revealed that under the constraint nondegenerate conditions for (D) and (P) (i.e., the dual and primal nondegeneracies in the IPM literature, e.g., [1]), respectively, the augmented Lagrangian method can be locally regarded as an approximate generalized Newton method applied to a semismooth equation. It is this connection that inspired us to investigate the augmented Lagrangian method for SDP problems. The approach of Jarre and Rendl [13] is to reformulate the problem as the minimization of a convex differentiable function in the primal-dual space. It is demonstrated in [13] that, numerically, the performance of using a nonlinear CG method to minimize this smooth convex function is quite comparable to that of using the Moreau–Yosida regularization approach presented in [19].

The objective functions $L_{\sigma_k}(\cdot, X^k)$ in the inner problems of the augmented Lagrangian method (3) are convex and continuously differentiable but not twice continuously differentiable (cf. (2)) due to the fact that $\Pi_{\mathcal{S}^n_+}(\cdot)$ is not continuously differentiable. It seems that Newton's method cannot be applied to solve the inner problems. However, since $\Pi_{\mathcal{S}^n_+}(\cdot)$ is strongly semismooth [36], the superlinear (quadratic) convergence analysis of the generalized Newton method established by Kummer [18] and Qi and Sun [26] for solving semismooth equations may be used to get fast convergence for solving the inner problems. In fact, the quadratic convergence and superb numerical results of the generalized Newton method combined with the CG method reported in [25] for solving a related problem strongly motivated us to study the semismooth Newton-CG method (see section 3) to solve the inner problems.

In [32, 33], Rockafellar established a general theory on the global convergence and local linear rate of convergence of the sequence generated by the augmented Lagrangian method for solving convex optimization problems including (D) and (P). In order to apply the general results in [32, 33], we characterize the Lipschitz continuity of the solution mapping for (P) defined in [33] at the origin in terms of the second order sufficient condition and the extended strict primal-dual constraint qualification for (P). In particular, under the uniqueness of Lagrange multipliers, we establish the equivalence among the Lipschitz continuity of the solution mapping at the origin, the second order sufficient condition, and the strict primal-dual constraint qualification. As for the inner problems in (3), we show that the constraint nondegeneracy for the corresponding dual problems is equivalent to the positive definiteness of the generalized Hessian of the objective functions in the inner problems. This is important for the success of applying an iterative solver to the generalized Newton equations in solving these inner problems. The differential structure of the nonsmooth metric projection operator $\Pi_{\mathcal{S}^n_+}(\cdot)$ in the augmented Lagrangian function L_σ plays a key role in achieving this result.

Besides the theoretical results we establish for the Newton-CG augmented Lagrangian (in short, SDPNAL) method proposed in this paper, we also demonstrate convincingly that with efficient implementations, the SDPNAL method can solve some very large SDP problems, with moderate accuracy, much more efficiently than the best alternative methods such as the inexact IPMs in [39], the modified barrier method in [17], the boundary-point method in [19], as well as the dedicated augmented Lagrangian method [5] for solving SDP problems arising from the lift-and-project procedure of Lovász and Schrijver.

The remaining parts of this paper are as follows. In section 2, we give some preliminaries including a brief introduction about concepts related to the method of multipliers and the characterizations of the Lipschitz continuity of the solution mapping for problem (P) at the origin. In section 3, we introduce a semismooth Newton-CG method for solving the inner optimization problems and analyze its global and local superlinear (quadratic) convergence for solving these inner problems. Section 4 presents the SDPNAL dual approach and its linear rate of convergence. Section 5 is on numerical issues of the SDPNAL method. We report numerical results in sections 6 and 7 for a variety of large-scale linear SDP problems and make final conclusions in section 8.

2. Preliminaries. From [32, 33], we know that the augmented Lagrangian method can be expressed in terms of the method of multipliers for (D). For the sake of subsequent discussions, we introduce related concepts.

Let $l(y, X) : \Re^m \times \mathcal{S}^n \to \Re$ be the ordinary Lagrangian function for (D) in extended form:

$$l(y,X) = \begin{cases} b^{\mathrm{T}} y - \langle X, \mathcal{A}^* y - C \rangle & \text{if } y \in \Re^m \text{ and } X \in \mathcal{S}^n_+, \\ -\infty & \text{if } y \in \Re^m \text{ and } X \notin \mathcal{S}^n_+. \end{cases} \tag{4}$$

The essential objective function in (D) is

$$f(y) = \sup_{X \in \mathcal{S}^n} l(y,X) = \begin{cases} b^{\mathrm{T}} y & \text{if } y \in \mathcal{F}_D, \\ +\infty & \text{otherwise}, \end{cases} \tag{5}$$

where $\mathcal{F}_D := \{y \in \Re^m \mid \mathcal{A}^* y - C \succeq \mathbf{0}\}$ is the feasible set of (D), while the essential objective function in (P) is

$$g(X) = \inf_{y \in \Re^m} l(y,X) = \begin{cases} \langle C, X \rangle & \text{if } X \in \mathcal{F}_P, \\ -\infty & \text{otherwise}, \end{cases} \tag{6}$$

where $\mathcal{F}_P := \{X \in \mathcal{S}^n \mid \mathcal{A}(X) = b, X \succeq \mathbf{0}\}$ is the feasible set of (P).

Assume that $\mathcal{F}_D \neq \emptyset$ and $\mathcal{F}_P \neq \emptyset$. As in Rockafellar [33], we define the following maximal monotone operator:

$$T_l(y,X) = \{(v,U) \in \Re^m \times \mathcal{S}^n \mid (v, -U) \in \partial l(y,X)\}, \quad (y,X) \in \Re^m \times \mathcal{S}^n.$$

Throughout this paper, the following condition for (P) is assumed to hold.

Assumption 1. *Problem (P) satisfies the condition*

$$\begin{cases} \mathcal{A} : \mathcal{S}^n \to \Re^m \text{ is onto}, \\ \exists\, X_0 \in \mathcal{S}^n_+ \text{ such that } \mathcal{A}(X_0) = b, \ X_0 \succ \mathbf{0}, \end{cases} \tag{7}$$

where $X_0 \succ \mathbf{0}$ means that X_0 is a symmetric positive definite matrix.

For each $v \in \Re^m$ and $U \in \mathcal{S}^n$, we consider the following parameterized problem:

$$(P(v,U)) \quad \max\{\langle C, X\rangle + \langle U, X \rangle \mid \mathcal{A}(X) + v = b, \quad X \succeq \mathbf{0}\}.$$

By using the fact that f is convex and g is concave, we know from Rockafellar [29, Theorem 23.5] that for each $v \in \Re^m$,

$$(\partial f)^{-1}(v) = \text{set of all optimal solutions to } (D(v, \mathbf{0})), \tag{8}$$

and that for each $U \in \mathcal{S}^n$,

$$-(\partial g)^{-1}(U) = \text{set of all optimal solutions to } (P(0,U)), \tag{9}$$

where for $(v,U) \in \Re^m \times \mathcal{S}^n$, $(D(v,U))$ is the (ordinary) dual of $(P(v,U))$ in the sense that

$$(D(v,U)) \quad \min\{b^{\mathrm{T}}y - v^{\mathrm{T}}y \,:\, \mathcal{A}^*y - U \succeq C\}.$$

Furthermore for any $(v,U) \in \Re^m \times \mathcal{S}^n$, under Assumption 1, we have that

$$\begin{aligned} T_l^{-1}(v,U) &= \arg\operatorname{minimax}\{\, l(y,X) - v^{\mathrm{T}}y + \langle U,\, X\rangle \mid y \in \Re^m, X \in \mathcal{S}^n\} \\ &= \text{set of all}\,(y,X)\,\text{satisfying the KKT conditions for } (P(v,U)) \text{ (cf. (12))}. \end{aligned} \tag{10}$$

Definition 1 (see [32]). *For a maximal monotone operator T from a finite dimensional linear vector space $\mathcal{X}$ to itself, we say that its inverse T^{-1} is Lipschitz continuous at the origin (with modulus $a \geq 0$) if there is a unique solution $\bar{z}$ to $z = T^{-1}(0)$, and for some $\tau > 0$ we have*

$$(\text{see})\|z - \bar{z}\| \leq a\|w\| \quad \textit{whenever} \quad z \in T^{-1}(w) \quad \textit{and} \quad \|w\| \leq \tau. \tag{11}$$

The first order optimality conditions, namely, the KKT conditions, of (D) and (P) are as follows:

$$\mathcal{A}(X) = b, \qquad \mathcal{S}^n_+ \ni (\mathcal{A}^*y - C) \perp X \in \mathcal{S}^n_+, \tag{12}$$

where "$(\mathcal{A}^*y - C) \perp X$" means that $(\mathcal{A}^*y - C)$ and X are orthogonal to each other; i.e., $\langle \mathcal{A}^*y - C, X\rangle = 0$. For any $X \in \mathcal{F}_P$, define the set

$$\mathcal{M}(X) := \{y \in \Re^m \mid (y,X) \text{ satisfies the KKT conditions (12)}\}. \tag{13}$$

Let $\overline{X}$ be an optimal solution to (P). Since (P) satisfies condition (7), $\mathcal{M}(\overline{X})$ is nonempty and bounded [31, Theorems 17 and 18]. Let $y \in \mathcal{M}(\overline{X})$ be arbitrarily chosen. Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ be the eigenvalues of $\overline{X}$ being arranged in nonincreasing order and let $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$ be the eigenvalues of $(\mathcal{A}^*y - C)$ being arranged in nondecreasing order. Denote $\alpha := \{i \,|\, \lambda_i > 0,\, i = 1,\ldots,n\}$ and $\gamma := \{i \,|\, \mu_i > 0,\, i = 1,\ldots,n\}$. Since $\overline{X}(\mathcal{A}^*y - C) = (\mathcal{A}^*y - C)\overline{X} = 0$, there exists an orthogonal matrix $P \in \Re^{n\times n}$ such that

$$\overline{X} = P\begin{bmatrix} \Lambda_\alpha & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} P^{\mathrm{T}} \quad \text{and} \quad (\mathcal{A}^*y - C) = P\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Lambda_\gamma \end{bmatrix} P^{\mathrm{T}}, \tag{14}$$

where Λ_α is the diagonal matrix whose diagonal entries are λ_i for $i \in \alpha$ and Λ_γ is the diagonal matrix whose diagonal entries are μ_i for $i \in \gamma$.

Let $A := \overline{X} - (\mathcal{A}^*y - C) \in \mathcal{S}_n$. Then, A has the following spectral decomposition:

$$A = P\Lambda P^{\mathrm{T}}, \quad \text{where} \quad \Lambda = \begin{bmatrix} \Lambda_\alpha & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -\Lambda_\gamma \end{bmatrix}. \tag{15}$$

Denote $\beta := \{1,\ldots,n\}\backslash(\alpha\cup\gamma)$. Write $P = [P_\alpha\ P_\beta\ P_\gamma]$ with $P_\alpha \in \Re^{n\times|\alpha|}$, $P_\beta \in \Re^{n\times|\beta|}$, and $P_\gamma \in \Re^{n\times|\gamma|}$. From [2], we know that the tangent cone of $\mathcal{S}^n_+$ at $\overline{X} \in \mathcal{S}^n_+$ can be characterized as

$$\mathcal{T}_{\mathcal{S}^n_+}(\overline{X}) = \{B \in \mathcal{S}^n \mid [P_\beta\ P_\gamma]^{\mathrm{T}}\, B\, [P_\beta\ P_\gamma] \succeq 0\}. \tag{16}$$

Similarly, the tangent cone of $\mathcal{S}^n_+$ at $(\mathcal{A}^*y - C)$ takes the form

$$\mathcal{T}_{\mathcal{S}^n_+}(\mathcal{A}^*y - C) = \{B \in \mathcal{S}^n \mid [P_\alpha\ P_\beta]^{\mathrm{T}}\, B\, [P_\alpha\ P_\beta] \succeq 0\}. \tag{17}$$

Recall that the *critical cone* of problem (P) at $\overline{X}$ is defined by (cf. [4, p. 151])

$$\mathcal{C}(\overline{X}) = \{B \in \mathcal{S}^n \mid \mathcal{A}(B) = 0,\ B \in \mathcal{T}_{\mathcal{S}^n_+}(\overline{X}),\ \langle C, B\rangle = 0\}. \tag{18}$$

Choose an arbitrary element $B \in \mathcal{C}(\overline{X})$. Denote $\widetilde{B} := P^{\mathrm{T}}BP$. Since $\overline{X}$ and $(\mathcal{A}^*y - C)$ have the spectral decompositions as in (14), we obtain that

$$0 = \langle C, B\rangle = \langle \mathcal{A}^*y - C, B\rangle = \left\langle \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Lambda_\gamma \end{bmatrix}, \begin{bmatrix} \widetilde{B}_{\alpha\alpha} & \widetilde{B}_{\alpha\beta} & \widetilde{B}_{\alpha\gamma} \\ \widetilde{B}^{\mathrm{T}}_{\alpha\beta} & \widetilde{B}_{\beta\beta} & \widetilde{B}_{\beta\gamma} \\ \widetilde{B}^{\mathrm{T}}_{\alpha\gamma} & \widetilde{B}^{\mathrm{T}}_{\beta\gamma} & \widetilde{B}_{\gamma\gamma} \end{bmatrix} \right\rangle,$$

which, together with (16) and (18), implies that $\widetilde{B}_{\gamma\gamma} = 0$. Thus

$$\widetilde{B}_{\beta\gamma} = 0 \quad \text{and} \quad \widetilde{B}_{\gamma\gamma} = 0.$$

Hence, $\mathcal{C}(\overline{X})$ can be rewritten as

$$\mathcal{C}(\overline{X}) = \{B \in \mathcal{S}^n \mid \mathcal{A}(B) = 0,\ P^{\mathrm{T}}_\beta BP_\beta \succeq 0,\ P^{\mathrm{T}}_\beta BP_\gamma = 0,\ P^{\mathrm{T}}_\gamma BP_\gamma = 0\}. \tag{19}$$

By using arguments similar to those above, we can also obtain that

$$\mathcal{T}_{\mathcal{S}^n_+}(\mathcal{A}^*y - C) \cap \overline{X}^\perp = \{B \in \mathcal{S}^n \mid P^{\mathrm{T}}_\alpha BP_\alpha = 0,\ P^{\mathrm{T}}_\alpha BP_\beta = 0,\ P^{\mathrm{T}}_\beta BP_\beta \succeq 0\}, \tag{20}$$

where $\overline{X}^\perp := \{B \in \mathcal{S}^n \mid \langle B, \overline{X}\rangle = 0\}$.

In order to analyze the rate of convergence of the SDPNAL method presented in section 4, we need the following result which characterizes the Lipschitz continuity of $-(\partial g)^{-1}$ at the origin. The result we establish here is stronger than that established in Proposition 15 of [7].

PROPOSITION 2.1. *Suppose that (P) satisfies condition (7). Let $\overline{X} \in \mathcal{S}^n_+$ be an optimal solution to (P). Then the following conditions are equivalent:*

(i) $-(\partial g)^{-1}$ *is Lipschitz continuous at the origin.*

(ii) *The second order sufficient condition*

$$\sup_{y\in\mathcal{M}(\overline{X})} \Upsilon_{\overline{X}}(\mathcal{A}^*y - C, H) > 0 \quad \forall\, H \in \mathcal{C}(\overline{X}) \setminus \{0\} \tag{21}$$

holds at $\overline{X}$, where for any $B \in \mathcal{S}^n$, the linear-quadratic function $\Upsilon_B : \mathcal{S}^n \times \mathcal{S}^n \to \Re$ is defined by

$$\Upsilon_B(M, H) := 2\,\langle M, HB^\dagger H\rangle, \quad (M, H) \in \mathcal{S}^n \times \mathcal{S}^n \tag{22}$$

and $B^\dagger$ is the Moore–Penrose pseudoinverse of B.

(iii) $\overline{X}$ *satisfies the extended strict primal-dual constraint qualification*

$$\mathcal{A}^*\Re^m + \operatorname{conv}\left(\bigcup_{y\in\mathcal{M}(\overline{X})}\left(\mathcal{T}_{\mathcal{S}^n_+}(\mathcal{A}^*y - C)\cap \overline{X}^{\perp}\right)\right) = \mathcal{S}^n, \tag{23}$$

where for any set $\mathcal{W}\subset\mathcal{S}^n$, $\operatorname{conv}(\mathcal{W})$ *denotes the convex hull of* $\mathcal{W}$.

Proof. "(i) $\Leftrightarrow$ (ii)." From [4, Theorem 3.137], we know that (ii) holds if and only if the quadratic growth condition

$$\langle C, \overline{X}\rangle \geq \langle C, X\rangle + c\|X-\overline{X}\|^2 \qquad \forall\, X\in\mathcal{N} \text{ such that } X\in\mathcal{F}_P \tag{24}$$

holds at $\overline{X}$ for some positive constant c and an open neighborhood $\mathcal{N}$ of $\overline{X}$ in $\mathcal{S}^n$. On the other hand, from [33, Proposition 3], we know that $-(\partial g)^{-1}$ is Lipschitz continuous at the origin if and only if the quadratic growth condition (24) holds at $\overline{X}$. Hence, (i) $\Leftrightarrow$ (ii).

Next we shall prove that (ii) $\Leftrightarrow$ (iii). For notational convenience, let

$$\Gamma := \operatorname{conv}\left(\bigcup_{y\in\mathcal{M}(\overline{X})}\left(\mathcal{T}_{\mathcal{S}^n_+}(\mathcal{A}^*y - C)\cap \overline{X}^{\perp}\right)\right). \tag{25}$$

"(ii) $\Rightarrow$ (iii)." Denote $\mathcal{D} := \mathcal{A}^*\Re^m + \Gamma$. For the purpose of contradiction, we assume that (iii) does not hold, i.e., $\mathcal{D}\neq\mathcal{S}^n$. Let $\operatorname{cl}(\mathcal{D})$ and $\operatorname{ri}(\mathcal{D})$ denote the closure of $\mathcal{D}$ and the relative interior of $\mathcal{D}$, respectively. By [29, Theorem 6.3], since $\operatorname{ri}(\mathcal{D}) = \operatorname{ri}(\operatorname{cl}(\mathcal{D}))$, the relative interior of $\operatorname{cl}(\mathcal{D})$, we know that $\operatorname{cl}(\mathcal{D})\neq\mathcal{S}^n$. Thus, there exists $B\in\mathcal{S}^n$ such that $B\notin\operatorname{cl}(\mathcal{D})$. Let $\overline{B}$ be the metric projection of B onto $\operatorname{cl}(\mathcal{D})$, i.e., $\overline{B} = \Pi_{\operatorname{cl}(\mathcal{D})}(B)$. Let $H = \overline{B} - B \neq 0$. Since $\operatorname{cl}(\mathcal{D})$ is a nonempty closed convex cone, from Zarantonello [44], we know that

$$\langle H, Z\rangle = \langle \overline{B} - B, Z\rangle \geq 0 \quad \forall\, Z\in\operatorname{cl}(\mathcal{D}).$$

In particular, we have $\langle H, \mathcal{A}^*z + Q\rangle \geq 0$ for all $z\in\Re^m$ and $Q\in\Gamma$, which implies that (by taking $Q = \mathbf{0}$) $\langle \mathcal{A}(H), z\rangle = \langle H, \mathcal{A}^*z\rangle \geq 0$ for all $z\in\Re^m$. Thus

$$\mathcal{A}(H) = 0 \quad \text{and} \quad \langle H, Q\rangle \geq 0 \quad \text{for any } Q\in\Gamma. \tag{26}$$

Since $0\neq H\in\mathcal{C}(\overline{X})$ and (ii) is assumed to hold, there exists $y\in\mathcal{M}(\overline{X})$ such that

$$\Upsilon_{\overline{X}}(\mathcal{A}^*y - C, H) > 0. \tag{27}$$

By using the fact that $(y, \overline{X})$ satisfies (12), we can assume that $\overline{X}$ and $(\mathcal{A}^*y - C)$ have spectral decompositions as in (14). Then, we know from (20) that for any $Q\in\mathcal{T}_{\mathcal{S}^n_+}(\mathcal{A}^*y - C)\cap\overline{X}^{\perp}$,

$$\begin{aligned} 0 \leq \langle H,\, Q\rangle &= \langle P\widetilde{H}P^{\mathrm{T}}, P\widetilde{Q}P^{\mathrm{T}}\rangle \\ &= \left\langle \begin{bmatrix} \widetilde{H}_{\alpha\alpha} & \widetilde{H}_{\alpha\beta} & \widetilde{H}_{\alpha\gamma} \\ \widetilde{H}^{\mathrm{T}}_{\alpha\beta} & \widetilde{H}_{\beta\beta} & \widetilde{H}_{\beta\gamma} \\ \widetilde{H}^{\mathrm{T}}_{\alpha\gamma} & \widetilde{H}^{\mathrm{T}}_{\beta\gamma} & \widetilde{H}_{\gamma\gamma} \end{bmatrix}, \begin{bmatrix} 0 & 0 & \widetilde{Q}_{\alpha\gamma} \\ 0 & \widetilde{Q}_{\beta\beta} & \widetilde{Q}_{\beta\gamma} \\ \widetilde{Q}_{\alpha\gamma} & \widetilde{Q}_{\beta\gamma} & \widetilde{Q}_{\gamma\gamma} \end{bmatrix} \right\rangle, \end{aligned} \tag{28}$$

where $\widetilde{H} = P^{\mathrm{T}}HP$ and $\widetilde{Q} = P^{\mathrm{T}}QP$. From (20) and (28), we have

$$\widetilde{H}_{\alpha\gamma} = 0, \quad \widetilde{H}_{\beta\gamma} = 0, \quad \widetilde{H}_{\gamma\gamma} = 0, \quad \text{and} \quad \widetilde{H}_{\beta\beta} \succeq 0. \tag{29}$$

By using (19), (26), and (29), we obtain that $H \in \mathcal{C}(\overline{X})$ and $P_\alpha^{\mathrm{T}}HP_\gamma = 0$. Note that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ and $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$ are the eigenvalues of $\overline{X}$ and $(\mathcal{A}^*y - C)$, respectively, and $\alpha = \{i \,|\, \lambda_i > 0,\, i = 1, \ldots, n\}$ and $\gamma = \{j \,|\, \mu_j > 0,\, j = 1, \ldots, n\}$. Therefore, from (22) and (14), we obtain that

$$\Upsilon_{\overline{X}}(\mathcal{A}^*y - C, H) = 2 \sum_{i \in \alpha, j \in \gamma} \frac{\mu_j}{\lambda_i} (P_i^{\mathrm{T}}HP_j)^2 = 0,$$

which contradicts (27). This contradiction shows (ii) $\Rightarrow$ (iii).

"(iii) $\Rightarrow$ (ii)." Assume that (ii) does not hold at $\overline{X}$. Then there exists $0 \neq H \in \mathcal{C}(\overline{X})$ such that

$$\sup_{y \in \mathcal{M}(\overline{X})} \Upsilon_{\overline{X}}(\mathcal{A}^*y - C, H) = 0. \tag{30}$$

Let y be an arbitrary element in $\mathcal{M}(\overline{X})$. Since $(y, \overline{X})$ satisfies (12), we can assume that there exists an orthogonal matrix $P \in \Re^{n \times n}$ such that $\overline{X}$ and $(\mathcal{A}^*y - C)$ have the spectral decompositions as in (14). From (14), (22), and (30), we have

$$0 \leq 2 \sum_{i \in \alpha, j \in \gamma} \frac{\mu_j}{\lambda_i} (P_i^{\mathrm{T}}HP_j)^2 = \Upsilon_{\overline{X}}(\mathcal{A}^*y - C, H) \leq \sup_{z \in \mathcal{M}(\overline{X})} \Upsilon_{\overline{X}}(\mathcal{A}^*z - C, H) = 0,$$

which implies

$$P_\alpha^{\mathrm{T}}HP_\gamma = 0. \tag{31}$$

Then, by using (19), (20), and (31), we have that for all $Q^y \in \mathcal{T}_{\mathcal{S}_+^n}(\mathcal{A}^*y - C) \cap \overline{X}^\perp$,

$$\langle Q^y, H \rangle = \langle P^{\mathrm{T}}Q^yP, P^{\mathrm{T}}HP \rangle = \langle P_\beta^{\mathrm{T}}Q^yP_\beta, P_\beta^{\mathrm{T}}HP_\beta \rangle \geq 0. \tag{32}$$

Since (iii) is assumed to hold, there exist $z \in \Re^m$ and $Q \in \Gamma$ such that

$$-H = \mathcal{A}^*z + Q. \tag{33}$$

By Carathéodory's theorem, there exist an integer $k \leq \frac{n(n+1)}{2} + 1$ and scalars $\alpha_i \geq 0$, $i = 1, 2, \ldots, k$, with $\sum_{i=1}^k \alpha_i = 1$, and

$$Q_i \in \bigcup_{y \in \mathcal{M}(\overline{X})} \left(\mathcal{T}_{\mathcal{S}_+^n}(\mathcal{A}^*y - C) \cap \overline{X}^\perp \right), \quad i = 1, 2, \ldots, k,$$

such that Q can be represented as

$$Q = \sum_{i=1}^k \alpha_i Q_i.$$

For each Q_i, there exists a $y^i \in \mathcal{M}(\overline{X})$ such that $Q_i \in \mathcal{T}_{\mathcal{S}_+^n}(\mathcal{A}^*y^i - C) \cap \overline{X}^\perp$. Then by using the fact that $H \in \mathcal{C}(\overline{X})$ and (32), we obtain that

$$\langle H, H \rangle = \langle -\mathcal{A}^*z - Q, H \rangle = -\langle z, \mathcal{A}H \rangle - \langle Q, H \rangle = 0 - \sum_{i=1}^k \alpha_i \langle Q_i, H \rangle \leq 0,$$

which contradicts the fact that $H \neq 0$. This contradiction shows that (ii) holds. $\square$

Proposition 2.1 characterizes the Lipschitz continuity of $-(\partial g)^{-1}$ at the origin by either the second sufficient condition (21) or the extended strict primal-dual constraint qualification (23). In particular, if $\mathcal{M}(\overline{X})$ is a singleton, we have the following simple equivalent conditions.

COROLLARY 2.2. *Suppose that* (P) *satisfies condition* (7)*. Let* $\overline{X}$ *be an optimal solution to* (P)*. If* $\mathcal{M}(\overline{X}) = \{\bar{y}\}$*, then the following are equivalent:*

(i) $-(\partial g)^{-1}$ *is Lipschitz continuous at the origin.*

(ii) *The following second order sufficient condition holds at* $\overline{X}$*:*

$$\Upsilon_{\overline{X}}(\mathcal{A}^*\bar{y} - C, H) > 0 \quad \forall\, H \in \mathcal{C}(\overline{X}) \setminus \{0\}. \tag{34}$$

(iii) $\overline{X}$ *satisfies the strict primal-dual constraint qualification*

$$\mathcal{A}^*\Re^m + \mathcal{T}_{\mathcal{S}^n_+}(\mathcal{A}^*\bar{y} - C) \cap \overline{X}^{\perp} = \mathcal{S}^n. \tag{35}$$

Remark 1. Note that in [7, Proposition 15], Chan and Sun proved that if $\mathcal{M}(\overline{X})$ is a singleton, then the *strong* second order sufficient condition (with the set $\mathcal{C}(\overline{X})$ in (34) being replaced by the superset $\{B \in \mathcal{S}^n \mid \mathcal{A}(B) = 0, P_\beta^T B P_\gamma = 0, P_\gamma^T B P_\gamma = 0\}$) is equivalent to the constraint nondegenerate condition, in the sense of Robinson [27, 28], at $\bar{y}$ for (D); i.e,

$$\mathcal{A}^*\Re^m + \mathrm{lin}(\mathcal{T}_{\mathcal{S}^n_+}(\mathcal{A}^*\bar{y} - C)) = \mathcal{S}^n. \tag{36}$$

Corollary 2.2 further establishes the equivalence between the second order sufficient condition (34) and the strict constraint qualification (35) under the condition that $\mathcal{M}(\overline{X})$ is a singleton.

One may observe that the strict primal-dual constraint qualification condition (35) is weaker than the constraint nondegenerate condition (36). However, if strict complementarity holds, i.e., $\overline{X} + (\mathcal{A}^*\bar{y} - C) \succ 0$ and hence β is the empty set, then (35) and (36) coincide.

The constraint nondegenerate condition (36) is equivalent to the dual nondegeneracy stated in [1, Theorem 9]. Note that under such a condition, the optimal solution $\overline{X}$ to (P) is unique.

Remark 2. In a similar way, we can establish parallel results for $(\partial f)^{-1}$ as for $-(\partial g)^{-1}$ in Proposition 2.1 and Corollary 2.2. For brevity, we omit the details.

3. A semismooth Newton-CG method for inner problems. In this section we introduce a semismooth Newton-CG method for solving the inner problems involved in the augmented Lagrangian method (3). For this purpose, we need the practical CG method described in [11, Algorithm 10.2.1] for solving the symmetric positive definite linear system. Since our convergence analysis of the semismooth Newton-CG method depends heavily on this practical CG method and its convergence property (Lemma 3.1), we shall give a brief description here.

3.1. A practical CG method. In this subsection, we consider a practical CG method to solve the linear equation

$$Ax = b\,, \tag{37}$$

where $b \in \Re^m$ and $A \in \Re^{m\times m}$ is assumed to be a symmetric positive definite matrix. The practical conjugate gradient algorithm [11, Algorithm 10.2.1] depends on two parameters: a maximum number of CG iterations $i_{\max} > 0$ and a tolerance $\eta \in (0, \|b\|)$.

ALGORITHM 1. A practical CG algorithm ($CG(\eta, i_{\max})$).

Given $x^0 = 0$ and $r^0 = b$.

While ($\|r^i\| > \eta$) or ($i < i_{\max}$)

Step 1.1. $i = i + 1$

Step 1.2. If $i = 1$; $p^1 = r^0$; else; $\beta_i = \|r^{i-1}\|^2/\|r^{i-2}\|^2$, $p^i = r^{i-1} + \beta_i\, p^{\,i-1}$; end

Step 1.3. $\alpha_i = \|r^{i-1}\|^2/\langle p^i, Ap^i\rangle$

Step 1.4. $x^i = x^{i-1} + \alpha_i p^i$

Step 1.5. $r^i = r^{i-1} - \alpha_i Ap^i$

LEMMA 3.1. *Let $0 < \bar{i} \le i_{\max}$ be the number of iterations when the practical CG algorithm, Algorithm 1, terminates. For all $i = 1, 2, \ldots, \bar{i}$, the iterates $\{x^i\}$ generated by Algorithm 1 satisfy*

$$\frac{1}{\lambda_{\max}(A)} \le \frac{\langle x^i, b\rangle}{\|b\|^2} \le \frac{1}{\lambda_{\min}(A)}, \tag{38}$$

where $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ are the smallest and largest eigenvalue of A, respectively.

Proof. Let x^* be the exact solution to (37), and let $e^i = x^* - x^i$ be the error in the ith iteration for $i \ge 0$. From [38, Theorem 38.1], we know that

$$\langle r^i, r^j\rangle = 0 \quad \text{for } j = 1, 2, \ldots, i-1, \tag{39}$$

where $r^i = b - Ax^i$. By using (39), the fact that in Algorithm 1, $r^0 = b$, and the definition of β_i, we have that

$$\begin{aligned} &\langle p^1, b\rangle = \|r^0\|^2, \\ &\langle p^i, b\rangle = \langle r^{i-1}, b\rangle + \beta_i \langle p^{i-1}, b\rangle = 0 + \prod_{j=2}^{i} \beta_j \langle p^1, b\rangle = \|r^{i-1}\|^2 \quad \forall\, i > 1. \end{aligned} \tag{40}$$

From [38, Theorem 38.2], we know that for $i \ge 1$,

$$\|e^{i-1}\|_A^2 = \|e^i\|_A^2 + \langle \alpha_i p^i, A(\alpha_i p^i)\rangle, \tag{41}$$

which, together with $\alpha_i \|r^{i-1}\|^2 = \langle \alpha_i p^i, A(\alpha_i p^i)\rangle$ (see Step 1.3), implies that

$$\alpha_i \|r^{i-1}\|^2 = \|e^{i-1}\|_A^2 - \|e^i\|_A^2. \tag{42}$$

Here for any $x \in \Re^m$, $\|x\|_A := \sqrt{\langle x, Ax\rangle}$. For any $i \ge 1$, by using (40), (42), and the fact that $x^0 = 0$, we have that

$$\begin{aligned} \langle x^i, b\rangle &= \langle x^{i-1}, b\rangle + \alpha_i \langle p^i, b\rangle = \langle x^0, b\rangle + \sum_{j=1}^{i} \alpha_j \langle p^j, b\rangle = \sum_{j=1}^{i} \alpha_j \|r^{j-1}\|^2 \\ &= \sum_{j=1}^{i} \left[\|e^{j-1}\|_A^2 - \|e^j\|_A^2\right] = \|e^0\|_A^2 - \|e^i\|_A^2, \end{aligned} \tag{43}$$

which, together with (41), implies that $\langle x^i, b\rangle \ge \langle x^{i-1}, b\rangle$, $i = 1, 2, \ldots, \bar{i}$. Thus

$$\frac{1}{\lambda_{\max}(A)} \le \alpha_1 = \frac{\langle x^1, b\rangle}{\|b\|^2} \le \frac{\langle x^i, b\rangle}{\|b\|^2}. \tag{44}$$

Since $e^0 = x^* - x^0 = A^{-1}b$, by (43), we obtain that for $1 \le i \le \bar{i}$,

$$\frac{\langle x^i, b\rangle}{\|b\|^2} \le \frac{\|e^0\|_A^2}{\|b\|^2} = \frac{\|A^{-1}b\|_A^2}{\|b\|^2} \le \frac{1}{\lambda_{\min}(A)}. \tag{45}$$

By combining (44) and (45), we complete the proof. $\square$

3.2. A semismooth Newton-CG method. For the augmented Lagrangian method (3), for some fixed $X \in \mathcal{S}^n$ and $\sigma > 0$, we need to consider the following form of inner problems:

$$\min\{\varphi(y) := L_\sigma(y, X) \mid y \in \Re^m\}. \tag{46}$$

As explained in the introduction, $\varphi(\cdot)$ is a continuously differentiable convex function, but fails to be twice continuously differentiable because the metric projector $\Pi_{\mathcal{S}^n_+}(\cdot)$ is not continuously differentiable. Fortunately, because $\Pi_{\mathcal{S}^n_+}(\cdot)$ is strongly semismooth [36], we can develop locally a semismooth Newton-CG method to solve the nonlinear equation

$$\nabla\varphi(y) = b - \mathcal{A}\Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^*y - C)) = 0 \tag{47}$$

and expect a superlinear (quadratic) convergence for solving (47).

Since $\Pi_{\mathcal{S}^n_+}(\cdot)$ is Lipschitz continuous with modulus 1, the mapping $\nabla\varphi$ is Lipschitz continuous on $\Re^m$. According to Rademacher's theorem, $\nabla\varphi$ is almost everywhere Fréchet-differentiable in $\Re^m$. Let $y \in \Re^m$. The generalized Hessian of φ at y is defined as

$$\partial^2\varphi(y) := \partial(\nabla\varphi)(y), \tag{48}$$

where $\partial(\nabla\varphi)(y)$ is the Clarke's generalized Jacobian of $\nabla\varphi$ at y [8]. Since it is difficult to express $\partial^2\varphi(y)$ exactly, we define the following alternative for $\partial^2\varphi(y)$:

$$\hat{\partial}^2\varphi(y) := \sigma\mathcal{A}\,\partial\Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^*y - C))\mathcal{A}^*. \tag{49}$$

From [8, p. 75], for $d \in \Re^m$,

$$\partial^2\varphi(y)d \subseteq \hat{\partial}^2\varphi(y)d, \tag{50}$$

which means that if every element in $\hat{\partial}^2\varphi(y)$ is positive definite, so is every element in $\partial^2\varphi(y)$.

For the semismooth Newton-CG method to be presented later, we need to compute an element $V \in \hat{\partial}^2\varphi(y)$. Since $X - \sigma(\mathcal{A}^*y - C)$ is a symmetric matrix in $\Re^{n\times n}$, there exists an orthogonal matrix $Q \in \Re^{n\times n}$ such that

$$X - \sigma(\mathcal{A}^*y - C) = Q\,\Gamma_y\,Q^{\mathrm{T}}, \tag{51}$$

where Γ_y is the diagonal matrix with diagonal entries consisting of the eigenvalues $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_n$ of $X - \sigma(\mathcal{A}^*y - C)$ being arranged in nonincreasing order. Define three index sets

$$\alpha := \{i \mid \lambda_i > 0\}, \quad \beta := \{i \mid \lambda_i = 0\}, \quad \text{and} \quad \gamma := \{i \mid \lambda_i < 0\}.$$

Define the operator $W_y^0 : \mathcal{S}^n \to \mathcal{S}^n$ by

$$W_y^0(H) := Q(\Omega \circ (Q^{\mathrm{T}} H Q))Q^{\mathrm{T}}, \quad H \in \mathcal{S}^n, \tag{52}$$

where " $\circ$ " denotes the Hadamard product of two matrices and

$$\Omega = \begin{bmatrix} E_{\alpha\alpha} & \nu_{\alpha\bar{\alpha}} \\ \nu_{\alpha\bar{\alpha}}^{\mathrm{T}} & 0 \end{bmatrix}, \quad \nu_{ij} := \frac{\lambda_i}{\lambda_i - \lambda_j}, \ i \in \alpha, j \in \bar{\alpha}, \tag{53}$$

$\bar{\alpha} = \{1, \ldots, n\} \backslash \alpha$, and $E_{\alpha\alpha} \in \mathcal{S}^{|\alpha|}$ is the matrix of ones. Define $V_y^0 : \Re^m \to \mathcal{S}^n$ by

$$V_y^0 d := \sigma \mathcal{A}[\, Q(\Omega \circ (\, Q^{\mathrm{T}}(\mathcal{A}^* d)\, Q))\, Q^{\mathrm{T}}], \quad d \in \Re^m. \tag{54}$$

Since, by Pang, Sun, and Sun [22, Lemma 11],

$$W_y^0 \in \partial \Pi_{\mathcal{S}_+^n}(X - \sigma(\mathcal{A}^* y - C)),$$

we know that $V_y^0 = \sigma \mathcal{A} W_y^0 \mathcal{A}^* \in \hat{\partial}^2 \varphi(y)$.

Next we shall characterize the positive definiteness of any $V_y \in \hat{\partial}^2 \varphi(y)$. From [33, p. 107] and the definitions of $l(y, X)$ in (4), we know that for any $(y, X, \sigma) \in \Re^m \times \mathcal{S}^n \times (0, +\infty)$,

$$L_\sigma(y, X) = \max_{Z \in \mathcal{S}^n} \left\{ l(y, Z) - \frac{1}{2\sigma} \|Z - X\|^2 \right\}.$$

Since condition (7) is assumed to hold, by the definition of $g(\cdot)$ in (6), we can deduce from [31, Theorems 17 and 18] that

$$\min_{y \in \Re^m} \varphi(y) = \min_{y \in \Re^m} \max_{Z \in \mathcal{S}^n} \left\{ l(y, Z) - \frac{1}{2\sigma} \|Z - X\|^2 \right\} \ = \ \max_{Z \in \mathcal{S}^n} \left\{ g(Z) - \frac{1}{2\sigma} \|Z - X\|^2 \right\}$$

$$= \max_{\mathcal{A}(Z) = b,\, Z \succeq \mathbf{0}} \left\{ \langle C, Z \rangle - \frac{1}{2\sigma} \|Z - X\|^2 \right\}. \tag{55}$$

Hence, (46) is the dual of

$$\max \left\{ \langle\, C, Z \rangle - \frac{1}{2\sigma} \|Z - X\|^2 \ \mid \ \mathcal{A}(Z) = b, \quad Z \succeq \mathbf{0} \right\}. \tag{56}$$

The KKT conditions of (56) are as follows:

$$\mathcal{A}(Z) = b, \qquad \mathcal{S}_+^n \ni Z \perp [Z - (X - \sigma(\mathcal{A}^* y - C))] \in \mathcal{S}_+^n. \tag{57}$$

PROPOSITION 3.2. *Suppose that the problem* (56) *satisfies condition* (7). *Let* $(\hat{y}, \widehat{Z}) \in \Re^m \times \mathcal{S}^n$ *be a pair that satisfies the KKT conditions* (57), *and let* P *be an orthogonal matrix such that* $\widehat{Z}$ *and* $\widehat{Z} - (X - \sigma(\mathcal{A}^* \hat{y} - C))$ *have the spectral decomposition as in* (14). *Then the following conditions are equivalent:*

(i) *The constraint nondegenerate condition*

$$\mathcal{A} \operatorname{lin}(\mathcal{T}_{\mathcal{S}_+^n}(\widehat{Z})) = \Re^m \tag{58}$$

holds at $\widehat{Z}$, *where* $\operatorname{lin}(\mathcal{T}_{\mathcal{S}_+^n}(\widehat{Z}))$ *denotes the lineality space of* $\mathcal{T}_{\mathcal{S}_+^n}(\widehat{Z})$, *i.e.,*

$$\operatorname{lin}(\mathcal{T}_{\mathcal{S}_+^n}(\widehat{Z})) = \{ B \in \mathcal{S}^n \mid [P_\beta \ P_\gamma]^T B \, [P_\beta \ P_\gamma] = 0 \}. \tag{59}$$

(ii) *Every* $V_{\hat{y}} \in \hat{\partial}^2\varphi(\hat{y})$ *is symmetric and positive definite.*

(iii) $V_{\hat{y}}^0 \in \hat{\partial}^2\varphi(\hat{y})$ *is symmetric and positive definite.*

Proof. "(i) $\Rightarrow$ (ii)." This part is implied in [3, Proposition 2.8] by the Jacobian amicability of the metric projector $\Pi_{\mathcal{S}^n_+}(\cdot)$.

"(ii) $\Rightarrow$ (iii)." This is obviously true since $V_{\hat{y}}^0 \in \hat{\partial}^2\varphi(\hat{y})$.

"(iii) $\Rightarrow$ (i)." Assume on the contrary that the constraint nondegenerate condition (58) does not hold at $\widehat{Z}$. Then we have

$$[\mathcal{A}\mathrm{lin}(\mathcal{T}_{\mathcal{S}^n_+}(\widehat{Z}))]^\perp \neq \{0\}.$$

Let $0 \neq d \in [\mathcal{A}\mathrm{lin}(\mathcal{T}_{\mathcal{S}^n_+}(\widehat{Z}))]^\perp$. Then

$$\langle\, d, \mathcal{A}(Q)\,\rangle = 0 \quad \forall\ Q \in \mathrm{lin}(\mathcal{T}_{\mathcal{S}^n_+}(\widehat{Z})),$$

which can be written as

$$0 = \langle\, \mathcal{A}^* d, Q\,\rangle = \langle\, P^{\mathrm{T}} H P, P^{\mathrm{T}} Q P\,\rangle \quad \forall\ Q \in \mathrm{lin}(\mathcal{T}_{\mathcal{S}^n_+}(\widehat{Z})), \tag{60}$$

where $H := \mathcal{A}^* d$. By using (59) and (60), we obtain that

$$P_\alpha^{\mathrm{T}} H P_\alpha = 0, \quad P_\alpha^{\mathrm{T}} H P_\beta = 0, \quad \text{and} \quad P_\alpha^{\mathrm{T}} H P_\gamma = 0.$$

By the definition of $W_{\hat{y}}^0$ in (52), it follows that $W_{\hat{y}}^0(H) = 0$. Therefore, for the corresponding $V_{\hat{y}}^0$ defined in (54), we have

$$\langle\, d, V_{\hat{y}}^0\, d\,\rangle = \langle\, d, \sigma\mathcal{A}W_{\hat{y}}^0(\mathcal{A}^* d)\,\rangle = \sigma\langle\, H, W_{\hat{y}}^0(H)\,\rangle = 0,$$

which contradicts (iii) since $d \neq 0$. This contradiction shows that (i) holds. ∎

Remark 3. The constraint nondegenerate condition (58) is equivalent to the primal nondegeneracy stated in [1, Theorem 6]. Under this condition, the solution $\hat{y}$ for (57) is unique.

3.3. Convergence analysis. In this subsection, we shall introduce the promised semismooth Newton-CG algorithm to solve (46). Choose $y^0 \in \Re^m$. Then the algorithm can be stated as follows.

ALGORITHM 2. A semismooth Newton-CG algorithm ($NCG(y^0, X, \sigma)$).

Step 0. Given $\mu \in (0, 1/2)$, $\bar{\eta} \in (0, 1)$, $\tau \in (0, 1]$, $\tau_1, \tau_2 \in (0, 1)$, and $\delta \in (0, 1)$.

Step 1. For $j = 0, 1, 2, \ldots$

Step 1.1. Given a maximum number of CG iterations $n_j > 0$, compute

$$\eta_j := \min(\bar{\eta}, \|\nabla\varphi(y^j)\|^{1+\tau}).$$

Apply the practical CG algorithm, Algorithm 1 ($CG(\eta_j, n_j)$), to find an approximation solution d^j to

$$(V_j + \varepsilon_j I)\, d = -\nabla\varphi(y^j), \tag{61}$$

where $V_j \in \hat{\partial}^2\varphi(y^j)$ is defined in (54) and $\varepsilon_j := \tau_1 \min\{\tau_2, \|\nabla\varphi(y^j)\|\}$.

Step 1.2. Set $\alpha_j = \delta^{m_j}$, where m_j is the first nonnegative integer m for which

$$\varphi(y^j + \delta^m d^j) \leq \varphi(y^j) + \mu\delta^m \langle \nabla\varphi(y^j), d^j\rangle. \tag{62}$$

Step 1.3. Set $y^{j+1} = y^j + \alpha_j\, d^j$.

Remark 4. In Algorithm 2, since V_j is always positive semidefinite, the matrix $V_j + \varepsilon_j I$ is positive definite as long as $\nabla\varphi(y^j) \neq 0$. Thus we can always apply Algorithm 1 to (61). As pointed out by one of the referees, for globalizing the semismooth Newton-CG algorithm, one may also use Steihaug's trust region truncated CG approach [34] instead of the line search truncated CG approach. Here we chose the line search approach because it is more convenient for our subsequent convergence analysis, and, more importantly, there is no evidence to suggest that the trust region approach is more preferable for convex optimization problems, in particular for the SDP problems studied in this paper.

Now we can analyze the global convergence of Algorithm 2 with the assumption that $\nabla\varphi(y^j) \neq 0$ for any $j \geq 0$. From Lemma 3.1, we know that the search direction d^j generated by Algorithm 2 is always a descent direction. This is stated in the following proposition.

PROPOSITION 3.3. *For every $j \geq 0$, the search direction d^j generated in Step* 1.2 *of Algorithm* 2 *satisfies*

$$\frac{1}{\lambda_{\max}(\widetilde{V_j})} \leq \frac{\langle -\nabla\varphi(y^j), d^j\rangle}{\|\nabla\varphi(y^j)\|^2} \leq \frac{1}{\lambda_{\min}(\widetilde{V_j})}\,, \tag{63}$$

where $\widetilde{V}_j := V_j + \varepsilon_j I$ and $\lambda_{\max}(\widetilde{V}_j)$ and $\lambda_{\min}(\widetilde{V}_j)$ are the largest and smallest eigenvalues of $\widetilde{V}_j$, respectively.

THEOREM 3.4. *Suppose that problem* (56) *satisfies condition* (7). *Then Algorithm* 2 *is well defined and any accumulation point $\hat{y}$ of $\{y^j\}$ generated by Algorithm* 2 *is an optimal solution to the inner problem* (46).

Proof. By Step 1.1 in Algorithm 2, for any $j \geq 0$, since, by (63), d^j is a descent direction, Algorithm 2 is well defined. Since problem (56) satisfies condition (7), from [31, Theorems 17′ and 18′], we know that the level set $\mathcal{L} := \{y \in \Re^m \mid \varphi(y) \leq \varphi(y^0)\}$ is a closed and bounded convex set. Therefore, the sequence $\{y^j\}$ is bounded. Let $\hat{y}$ be any accumulation point of $\{y^j\}$. Then, by making use of Proposition 3.3 and the Lipschitz continuity of $\Pi_{\mathcal{S}^n_+}(\cdot)$, we can easily derive that $\nabla\varphi(\hat{y}) = 0$. By the convexity of $\varphi(\cdot)$, $\hat{y}$ is an optimal solution of (46). ∎

Next we shall discuss the rate of convergence of Algorithm 2.

THEOREM 3.5. *Assume that problem* (56) *satisfies condition* (7). *Let $\hat{y}$ be an accumulation point of the infinite sequence $\{y^j\}$ generated by Algorithm* 2 *for solving the inner problem* (46). *Suppose that at each step $j \geq 0$, when the practical CG algorithm, Algorithm* 1, *terminates, the tolerance η_j is achieved (e.g., when $n_j = m+1$); i.e.,*

$$\|\nabla\varphi(y^j) + (V_j + \varepsilon_j I)\, d^j\| \leq \eta_j. \tag{64}$$

Assume that the constraint nondegenerate condition (58) *holds at $\widehat{Z} := \Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^*\hat{y} - C))$. Then the whole sequence $\{y^j\}$ converges to $\hat{y}$ and*

$$\|y^{j+1} - \hat{y}\| = O(\|y^j - \hat{y}\|^{1+\tau}). \tag{65}$$

Proof. By Theorem 3.4, we know that the infinite sequence $\{y^j\}$ is bounded and $\hat{y}$ is an optimal solution to (46) with

$$\nabla\varphi(\hat{y}) = 0.$$

Since the constraint nondegenerate condition (58) is assumed to hold at $\widehat{Z}$, $\hat{y}$ is the unique optimal solution to (46). It then follows from Theorem 3.4 that $\{y^j\}$ converges to $\hat{y}$. From Proposition 3.2, we know that for any $V_{\hat{y}} \in \hat{\partial}^2\varphi(\hat{y})$ defined in (49), there exists a $W_{\hat{y}} \in \partial\Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^*\hat{y} - C))$ such that

$$V_{\hat{y}} = \sigma \mathcal{A} W_{\hat{y}} \mathcal{A}^* \succ \mathbf{0}.$$

Then, for all j sufficiently large, $\{\|(V_j + \varepsilon_j I)^{-1}\|\}$ is uniformly bounded.

For any V_j, $j \geq 0$, there exists a $W_j \in \partial\Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^*y^j - C))$ such that

$$V_j = \sigma \mathcal{A} W_j \mathcal{A}^*. \tag{66}$$

Since $\Pi_{\mathcal{S}^n_+}(\cdot)$ is strongly semismooth [36], it holds that for all j sufficiently large,

$$\begin{aligned}
&\|y^j + d^j - \hat{y}\| \;=\; \|y^j + (V_j + \varepsilon_j I)^{-1}((\nabla\varphi(y^j) + (V_j + \varepsilon_j I)\, d^j) - \nabla\varphi(y^j)) - \hat{y}\| \\
&\leq\; \|y^j - \hat{y} - (V_j + \varepsilon_j I)^{-1}\nabla\varphi(y^j)\| + \|(V_j + \varepsilon_j I)^{-1}\|\, \|\nabla\varphi(y^j) + (V_j + \varepsilon_j I)\, d^j\| \\
&\leq\; \|(V_j + \varepsilon_j I)^{-1}\|(\|\nabla\varphi(y^j) - \nabla\varphi(\hat{y}) - V_j(y^j - \hat{y})\| + \varepsilon_j\|y^j - \hat{y}\| + \eta_j) \\
&\leq\; O(\|\mathcal{A}\|\|\Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^*y^j - C)) - \Pi_{\mathcal{S}^n_+}(X - \sigma(\mathcal{A}^*\hat{y} - C)) - W_j(\sigma\mathcal{A}^*(y^j - \hat{y}))\|) \\
&\quad + O(\tau_1\|\nabla\varphi(y^j)\|\|y^j - \hat{y}\| + \|\nabla\varphi(y^j)\|^{1+\tau}) \\
&\leq\; O(\|\sigma\mathcal{A}^*(y^j - \hat{y})\|^2) + O(\tau_1\, \|\nabla\varphi(y^j) - \nabla\varphi(\hat{y})\|\|y^j - \hat{y}\| + \|\nabla\varphi(y^j) - \nabla\varphi(\hat{y})\|^{1+\tau}) \\
&\leq\; O(\|y^j - \hat{y}\|^2) + O(\tau_1\sigma\|\mathcal{A}\|\|\mathcal{A}^*\|\|y^j - \hat{y}\|^2 + (\sigma\|\mathcal{A}\|\|\mathcal{A}^*\|\|y^j - \hat{y}\|)^{1+\tau}) \\
&=\; O(\|y^j - \hat{y}\|^{1+\tau}),
\end{aligned} \tag{67}$$

which implies that, for all j sufficiently large,

$$y^j - \hat{y} = -d^j + O(\|d^j\|^{1+\tau}) \quad \text{and} \quad \|d^j\| \to 0. \tag{68}$$

For each $j \geq 0$, let $R^j := \nabla\varphi(y^j) + (V_j + \varepsilon_j I)\, d^j$. Then, for all j sufficiently large,

$$\begin{aligned}
&\langle\, \nabla\varphi(y^j), d^j\, \rangle + \langle\, d^j, (V_j + \varepsilon_j I)\, d^j\, \rangle \;=\; \langle\, R^j, d^j\, \rangle \\
&\quad \leq \eta_j\|d^j\| \;\leq\; \|\nabla\varphi(y^j)\|^{1+\tau}\, \|d^j\| \;=\; \|\nabla\varphi(y^j) - \nabla\varphi(\hat{y})\|^{1+\tau}\|d^j\| \\
&\quad \leq \sigma\|d^j\|\|\mathcal{A}\|\|\mathcal{A}^*\|\|y^j - \hat{y}\|^{1+\tau} \\
&\quad \leq O(\|d^j\|^{2+\tau}),
\end{aligned}$$

which, together with (68) and the fact that $\|(V_j + \varepsilon_j I)^{-1}\|$ is uniformly bounded, implies that there exists a constant $\hat{\delta} > 0$ such that

$$-\langle\, \nabla\varphi(y^j), d^j\, \rangle \geq \hat{\delta}\|d^j\|^2 \qquad \forall\; j \text{ sufficiently large.}$$

Since $\nabla\varphi(\cdot)$ is (strongly) semismooth at $\hat{y}$ (because $\Pi_{\mathcal{S}^n_+}(\cdot)$ is strongly semismooth everywhere), from [10, Theorem 3.3 and Remark 3.4] or [21], we know that for $\mu \in (0, 1/2)$, there exists an integer j_0 such that for any $j \geq j_0$,

$$\varphi(y^j + d^j) \leq \varphi(y^j) + \mu\langle\nabla\varphi(y^j), d^j\rangle,$$

which means that, for all $j \geq j_0$,

$$y^{j+1} = y^j + d^j.$$

This, together with (67), completes the proof. $\square$

Theorem 3.5 shows that the rate of convergence for Algorithm 2 is of order $(1+\tau)$. If $\tau = 1$, this corresponds to quadratic convergence. However, this will need more CG iterations in Algorithm 1. To save computational time, in practice we choose $\tau = 0.1 \sim 0.2$, which still ensures that Algorithm 2 achieves superlinear convergence.

4. A Newton-CG augmented Lagrangian method.

In this section, we shall introduce a SPDNAL algorithm for solving problems (D) and (P). For any $k \geq 0$, denote $\varphi_k(\cdot) \equiv L_{\sigma_k}(\cdot, X^k)$. Since the inner problems cannot be solved exactly, we will use the following stopping criteria considered by Rockafellar [32, 33] for terminating Algorithm 2:

(A) $\varphi_k(y^{k+1}) - \inf \varphi_k \leq \epsilon_k^2/2\sigma_k$, $\epsilon_k \geq 0$, $\sum_{k=0}^{\infty} \epsilon_k < \infty$.

(B) $\varphi_k(y^{k+1}) - \inf \varphi_k \leq (\delta_k^2/2\sigma_k)\|X^{k+1} - X^k\|^2$, $\delta_k \geq 0$, $\sum_{k=0}^{\infty} \delta_k < \infty$.

(B′) $\|\nabla\varphi_k(y^{k+1})\| \leq (\delta_k^{'}/\sigma_k)\|X^{k+1} - X^k\|$, $0 \leq \delta_k^{'} \to 0$.

Algorithm 3. A Newton-CG augmented Lagrangian (SDPNAL) algorithm.

Step 0. Given $(y^0, X^0) \in \Re^m \times \mathcal{S}_+^n$, $\sigma_0 > 0$, a threshold $\bar{\sigma} \geq \sigma_0 > 0$, and $\rho > 1$.

Step 1. For $k = 0, 1, 2, \ldots$

Step 1.1. Starting with y^k as the initial point, apply Algorithm 2 to $\varphi_k(\cdot)$ to find $y^{k+1} = \mathrm{NCG}(y^k, X^k, \sigma_k)$ and $X^{k+1} = \Pi_{\mathcal{S}_+^n}(X^k - \sigma_k(\mathcal{A}^* y^{k+1} - C))$ satisfying (A), (B), or (B′).

Step 1.2. If $\sigma_k \leq \bar{\sigma}$, $\sigma_{k+1} = \rho\,\sigma_k$ or $\sigma_{k+1} = \sigma_k$.

The global convergence of Algorithm 3 follows from Rockafellar [32, Theorem 1] and [33, Theorem 4] without much difficulty.

Theorem 4.1. *Let Algorithm* 3 *be executed with stopping criterion* (A). *If* (D) *satisfies condition* (7), *i.e., if there exists* $z^0 \in \Re^m$ *such that*

$$\mathcal{A}^* z^0 - C \succ \mathbf{0}, \tag{69}$$

then the sequence $\{X^k\} \subset \mathcal{S}_+^n$ *generated by Algorithm* 3 *is bounded and* $\{X^k\}$ *converges to* $\overline{X}$, *where* $\overline{X}$ *is some optimal solution to* (P), *and* $\{y^k\}$ *is asymptotically minimizing for* (D) *with* $\max(P) = \inf(D)$.

If $\{X^k\}$ *is bounded and* (P) *satisfies condition* (7), *then the sequence* $\{y^k\}$ *is also bounded, and all of its accumulation points of the sequence* $\{y^k\}$ *are optimal solutions to* (D).

Next we state the local linear convergence of the SPDNAL algorithm.

Theorem 4.2. *Let Algorithm* 3 *be executed with stopping criteria* (A) *and* (B). *Assume that* (D) *satisfies condition* (69) *and* (P) *satisfies condition* (7). *If the extended strict primal-dual constraint qualification* (23) *holds at* $\overline{X}$, *where* $\overline{X}$ *is an optimal solution to* (P), *then the generated sequence* $\{X^k\} \subset \mathcal{S}_+^n$ *is bounded and* $\{X^k\}$ *converges to the unique solution* $\overline{X}$ *with* $\max(P) = \min(D)$, *and*

$$\|X^{k+1} - \overline{X}\| \leq \theta_k \|X^k - \overline{X}\| \quad \forall\ k \text{ sufficiently large},$$

where

$$\theta_k = \left[a_g(a_g^2 + \sigma_k^2)^{-1/2} + \delta_k\right](1 - \delta_k)^{-1} \to \theta_\infty = a_g(a_g^2 + \sigma_\infty^2)^{-1/2} < 1 \quad \text{as } \sigma_k \to \sigma_\infty,$$

and a_g is a Lipschitz constant of $-(\partial g)^{-1}$ at the origin (cf. Proposition 2.1*). The conclusions of Theorem* 4.1 *about $\{y^k\}$ are valid.*

Moreover, if the stopping criterion (B′) *is also used and the constraint nondegenerate conditions* (36) *and* (58) *hold at $\bar{y}$ and $\overline{X}$, respectively, then, in addition to the above conclusions, the sequence $\{y^k\} \to \bar{y}$, where $\bar{y}$ is the unique optimal solution to* (*D*)*, and one has*

$$\|y^{k+1} - \bar{y}\| \le \theta'_k \|X^{k+1} - X^k\| \quad \forall\ k \text{ sufficiently large},$$

where $\theta'_k = a_l(1 + \delta'_k)/\sigma_k \to \delta_\infty = a_l/\sigma_\infty$ and a_l is a Lipschitz constant of T_l^{-1} at the origin.

Proof. Conclusions of the first part of Theorem 4.2 follow from the results in [32, Theorem 2] and [33, Theorem 5] combined with Proposition 2.1. By using the fact that T_l^{-1} is Lipschitz continuous near the origin under the assumption that the constraint nondegenerate conditions (36) and (58) hold, respectively, at $\bar{y}$ and $\overline{X}$ [7, Theorem 18], we can directly obtain conclusions of the second part of this theorem from [32, Theorem 2] and [33, Theorem 5]. □

Remark 5. Note that in (3) we can also add the term $\frac{1}{2\sigma_k}\|y - y^k\|^2$ to $L_{\sigma_k}(y, X^k)$ such that $L_{\sigma_k}(y, X^k) + \frac{1}{2\sigma_k}\|y - y^k\|^2$ is a strongly convex function. This actually corresponds to the proximal method of multipliers considered in [33, section 5] for which the kth iteration is given by

$$\left\{ \begin{array}{l} y^{k+1} \approx \arg\min\limits_{y \in \Re^m} \left\{ L_{\sigma_k}(y, X^k) + \frac{1}{2\sigma_k}\|y - y^k\|^2 \right\}, \\ X^{k+1} = \Pi_{\mathcal{S}^n_+}(X^k - \sigma_k(\mathcal{A}^* y^{k+1} - C)), \\ \sigma_{k+1} = \rho\sigma_k \quad \text{or} \quad \sigma_{k+1} = \sigma_k. \end{array} \right. \tag{70}$$

Convergence analysis for (70) can be conducted in a way parallel to that in (3).

5. Numerical issues in the semismooth Newton-CG algorithm. In applying Algorithm 2 to solve the inner subproblem (46), the most expensive step is in computing the direction d at a given y from the linear system (61). Thus (61) must be solved as efficiently as possible. Let

$$M := \sigma\, AQ \otimes Q \mathrm{diag}(\mathrm{vec}(\Omega)) Q^{\mathrm{T}} \otimes Q^{\mathrm{T}} A^{\mathrm{T}},$$

where Q and Ω are given as in (51) and (53), respectively. Here A denotes the matrix representation of $\mathcal{A}$ with respect to the standard bases of $\Re^{n\times n}$ and $\Re^m$. The direction d is computed from the following linear system:

$$(M + \varepsilon I)\, d \;=\; -\nabla\varphi(y). \tag{71}$$

To achieve a faster convergence rate when applying the CG method to solve (71), one may apply a preconditioner to the system. By observing that the matrix Ω has elements all in the interval $[0, 1]$ and that the elements in the (α, α) block are all ones, one may simply approximate Ω by the matrix of ones, and hence a natural preconditioner for the coefficient matrix in (71) is simply the matrix $\widehat{M} := \sigma AA^{\mathrm{T}} + \varepsilon I$. However, using $\widehat{M}$ as the preconditioner may be costly since it requires the Cholesky factorization of AA^{T} and each preconditioning step requires the solution of two triangular linear systems. The last statement holds in particular when the Cholesky factor has a large number of fill-ins. Thus, in our implementation, we simply use $\mathrm{diag}(\widehat{M})$ as the preconditioner rather than $\widehat{M}$.

Next we discuss how to efficiently compute the matrix-vector multiplication Md for a given $d \in \Re^m$ by exploiting the structure of Ω. Observe that $Md = \sigma\mathcal{A}(Y)$, where $Y = Q(\Omega \circ (Q^{\mathrm{T}} D Q))Q^{\mathrm{T}}$ with $D = \mathcal{A}^* d$. Thus the efficient computation of Md relies on our ability to efficiently compute the matrix Y given D. By noting that

$$Y = [Q_\alpha \; Q_{\bar\alpha}] \begin{bmatrix} Q_\alpha^{\mathrm{T}} D Q_\alpha & \nu_{\alpha\bar\alpha} \circ (Q_\alpha^{\mathrm{T}} D Q_{\bar\alpha}) \\ \nu_{\alpha\bar\alpha}^{\mathrm{T}} \circ (Q_{\bar\alpha}^{\mathrm{T}} D Q_\alpha) & 0 \end{bmatrix} \begin{bmatrix} Q_\alpha^{\mathrm{T}} \\ Q_{\bar\alpha}^{\mathrm{T}} \end{bmatrix} = H + H^{\mathrm{T}}, \tag{72}$$

where $H = Q_\alpha[\frac{1}{2}(UQ_\alpha)Q_\alpha^{\mathrm{T}} + (\nu_{\alpha\bar\alpha} \circ (UQ_{\bar\alpha}))Q_{\bar\alpha}^{\mathrm{T}}]$ with $U = Q_\alpha^{\mathrm{T}} D$, it is easy to see that Y can be computed in at most $8|\alpha|n^2$ flops. By considering $Y = D - Q((E - \Omega) \circ (Q^T D Q))Q^T$, where E is the matrix of all ones, one can also compute Y in at most $8|\bar\alpha|n^2$ flops. Thus Y can be computed in at most $8\min\{|\alpha|, |\bar\alpha|\}n^2$ flops. The above computational complexity shows that the SDPNAL algorithm is able to take advantage of any low-rank or high-rank property of the optimal solution $\overline{X}$ to reduce computational cost. In contrast, for inexact IPMs such as those proposed in [39], the matrix-vector multiplication in each CG iteration would require $\Theta(n^3)$ flops.

Finally, we should mention that the computational cost of the full eigenvalue decomposition in (51) can sometimes dominate the cost of solving (71), especially when n is large. In our implementation, we use the LAPACK routine `dsyevd.f` (based on a divide-and-conquer strategy) to compute the full eigenvalue decomposition of a symmetric matrix. We have found it to be 7 to 10 times faster than the MATLAB `eig` routine when n is larger than 500.

6. Numerical experiments. We implemented the SDPNAL algorithm in MATLAB to solve a variety of large SDP problems with m up to $2,156,544$ and n up to $4,110$ on a PC (Intel Xeon 3.2 GHz with 4G of RAM). We measure the infeasibilities and optimality for the primal and dual problems as follows:

$$R_D = \frac{\|C + S - \mathcal{A}^* y\|}{1 + \|C\|}, \quad R_P = \frac{\|b - \mathcal{A}(X)\|}{1 + \|b\|}, \quad \text{gap} = \frac{b^T y - \langle C, X\rangle}{1 + |b^T y| + |\langle C, X\rangle|}, \tag{73}$$

where $S = (\Pi_{\mathcal{S}_+^n}(W) - W)/\sigma$ with $W = X - \sigma(\mathcal{A}^* y - C)$. The above measures are the same as those adopted in the Seventh DIMACS Implementation Challenge [15], except that we used the Euclidean norms $\|b\|$ and $\|C\|$ in the denominators instead of ∞-norms. We do not check the infeasibilities of the conditions $X \succeq 0$, $Z \succeq 0$, $XZ = 0$, since they are satisfied up to machine precision throughout the SDPNAL algorithm.

In our numerical experiments, we stop the SDPNAL algorithm when

$$\max\{R_D, R_P\} \;\le\; 10^{-6}. \tag{74}$$

We choose the initial iterate $y^0 = 0$, $X^0 = 0$, and $\sigma_0 = 10$.

In solving the subproblem (46), we cap the number of Newton iterations at 40, while in computing the inexact Newton direction from (61), we stop the CG solver when the maximum number of CG steps exceeds 500, or when the convergence is too slow in that the reduction in the residual norm is exceedingly small.

In this paper, we will mainly compare the performance of the SDPNAL algorithm with the boundary-point method, introduced in [24], that is coded in the MATLAB program `mprw.m` downloaded from Rendl's web page (http://www.math.uni-klu.ac.at/or/Software/). It basically implements the following algorithm: given $\sigma_0 > 0$, $X^0 \in \mathcal{S}^n$, $y^0 \in \Re^m$, accuracy level ε, perform the following loop:

$W = X^j - \sigma_j(\mathcal{A}^* y^j - C)$, $X^{j+1} = \Pi_{\mathcal{S}^n_+}(W)$, $S = (X^{j+1} - W)/\sigma_j$,

$y^{j+1} = y^j - (\sigma_j \mathcal{A}\mathcal{A}^*)^{-1}(b - \mathcal{A}(X^{j+1}))$,

$R_P = \|b - \mathcal{A}(X^{j+1})\|/(1 + \|b\|)$, $R_D = \|C + S - \mathcal{A}^* y^{j+1}\|/(1 + \|C\|)$,

If $\max\{R_P, R_D\} \leq \varepsilon$, stop; else, update σ_j, end.

Note that in the second step of the above algorithm, it is actually applying one iteration of a modified gradient method to solve the subproblem (46). But as the iterate y^{j+1} in the above algorithm is not necessarily a good approximate minimizer for (46), there is no convergence guarantee for the algorithm implemented. Next, we remark on the computational aspects of the above algorithm. Suppose that the Cholesky factorization of $\mathcal{A}\mathcal{A}^*$ is precomputed. Then each iteration of the above algorithm requires the solution of two triangular linear systems and one full eigenvalue decomposition of an $n \times n$ symmetric matrix. Thus each iteration of the algorithm may become rather expensive when the Cholesky factor of $\mathcal{A}\mathcal{A}^*$ is fairly dense or when $n \geq 500$, and the whole algorithm may be very expensive if a large number of iterations are needed to reach the desired accuracy. In our experiments, we set the maximum number of iterations allowed in the boundary-point method to 2000 instead of the default of 300. This is because the latter is sometimes too small for `mprw.m` to deliver a solution with decent accuracy.

In the program `mprw.m` [24], the authors suggested choosing σ_0 in the interval $[0.1, 10]$ if the SDP data is normalized. But we should mention that the performance of the boundary-point method is quite sensitive to the choice of σ_0. Another point mentioned in [24] is that when the rank of the optimal solution $\overline{X}$ is much smaller than n, the boundary-point method typically will perform poorly.

6.1. Random sparse SDP problems. We first consider the collection of 18 random sparse SDP problems tested in [19], which reported the performance of the boundary-point method introduced in [24].

In Table 1, we report the results obtained by the SDPNAL algorithm for the sparse SDP problems considered in [19]. In the interest of saving space, we report the results for only 6 of the larger instances. Interested readers may refer to [45] for the full table. The first three columns of the table give the problem name, the dimension of the variable y (m), and the size of the matrix C (n_s) and the number of linear inequality constraints (n_l) in (D), respectively. The middle five columns give the number of outer iterations, the total number of inner iterations, the average number of PCG steps taken to solve (71), and the objective values $\langle C, X \rangle$ and $b^T y$. The relative infeasibilities and gap, as well as times (in the format hours:minutes:seconds) are listed in the last four columns.

TABLE 1
Results for the SDPNAL algorithm on the random sparse SDP problems considered in [19].

Problem	$m \mid n_s; n_l$	it| itsub| pcg	$\langle C, X \rangle$	$b^T y$	R_P| R_D| gap	Time
Rn6m50p3	50000 | 600;	10| 50| 58.2	-3.86413091 2	-3.86353173 2	2.8-7| 8.5-7| -7.7-5	7:53
Rn6m60p3	60000 | 600;	9| 47| 48.3	6.41737682 2	6.41803361 2	5.0-7| 8.7-7| -5.1-5	7:00
Rn7m50p3	50000 | 700;	12| 52| 31.6	3.13203609 2	3.13240876 2	7.4-7| 5.4-7| -5.9-5	6:18
Rn7m70p3	70000 | 700;	10| 48| 41.6	-3.69557843 2	-3.69479811 2	2.4-7| 8.7-7| **-1.1-4**	8:48
Rn8m70p3	70000 | 800;	11| 51| 33.3	2.33139641 3	2.33149302 3	1.8-7| 9.9-7| -2.1-5	9:37
Rn8m100p3	100000 | 800;	10| 52| 55.8	2.25928848 3	2.25937157 3	1.3-7| 7.3-7| -1.8-5	18:49

TABLE 2

Results obtained by the boundary-point method in [19] *on the random sparse SDP problems considered therein. The parameter σ_0 is set to* 0.1*, which gives better timings than the default initial value of* 1.

Problem	$m \mid n_s; n_l$	it	$\langle C, X\rangle$	$b^T y$	$R_P \mid R_D \mid$ gap	Time
Rn6m50p3	50000 | 600;	142	-3.86413897 2	-3.86413511 2	9.9-7| 5.7-8| -5.0-7	1:21
Rn6m60p3	60000 | 600;	137	6.41736718 2	6.41736746 2	9.9-7| 3.0-8| -2.2-8	2:09
Rn7m50p3	50000 | 700;	165	3.13202583 2	3.13205602 2	9.9-7| 1.1-7| -4.8-6	2:07
Rn7m70p3	70000 | 700;	136	-3.69558765 2	-3.69558700 2	9.9-7| 4.2-8| -8.9-8	2:10
Rn8m70p3	70000 | 800;	158	2.33139551 3	2.33139759 3	9.9-7| 8.3-8| -4.5-7	2:54
Rn8m100p3	100000 | 800;	135	2.25928693 3	2.25928781 3	9.4-7| 2.9-8| -1.9-7	4:16

Table 2 lists the results obtained by the boundary-point method implemented in the program `mprw.m`.

Comparing the results in Tables 1 and 2, we observe that the boundary-point method outperformed the SDPNAL algorithm. The former is about 2 to 5 times faster than the latter on most of the problems. It is rather surprising that the boundary-point method implemented in `mprw.m`, being a gradient-based method and without convergence guarantee, can be so efficient in solving this class of sparse random SDP problems, with all the SDP problems solved within 250 iterations. For this collection of SDP problems, the ratios $\mathrm{rank}(\overline{X})/n$ for all the problems, except for `Rn6m20p4` (not listed in Table 1), are greater than 0.25.

6.2. SDP problems arising from relaxation of frequency assignment problems. Next we consider SDP problems arising from semidefinite relaxation of frequency assignment problems (FAPs) [9]. The explicit description of the SDP in the form (P) is given in [6, eq. (5)].

Observe that for the FAPs, the SDP problems contain nonnegative vector variables in addition to positive semidefinite matrix variables. However, it is easy to extend the SDPNAL algorithm and the boundary-point method in `mprw.m` to accommodate the nonnegative variables.

Tables 3 and 4 list the results obtained by the SDPNAL algorithm and the boundary-point method for the SDP relaxation of FAPs tested in [6], respectively. Note that we report the results for only 6 out of 14 problems tested to save space. Again, the reader may refer to [45] for the full tables. For this collection of SDP problems, the SDPNAL algorithm outperformed the boundary-point method. While the SDPNAL algorithm can achieve rather high accuracy in $\max\{R_P, R_D, \mathrm{gap}\}$ for all the SDP problems, the boundary-point method fails to achieve satisfactory accuracy after 2000 iterations in that the primal and dual objective values obtained have yet to converge close to the optimal values. The results in Table 4 demonstrate a phenomenon that is typical of a purely gradient-based method; i.e., it may stagnate or converge very slowly well before the required accuracy is achieved.

Notice that in Table 4, we also report (at the request of one of the referees) the results obtained by `mprw.m` using the default of 300 iterations. It is quite obvious that the primal and dual objective values obtained for most of the problems differ significantly from the optimal values. For example, for `fap25`, the values obtained are 14.70 and 16.90, whereas the optimal value is 12.88. Thus the default setting of 300 iterations in `mprw.m` is sometimes not enough to achieve a solution with decent accuracy.

TABLE 3
Results for the SDPNAL algorithm on the FAPs.

Problem	m | $n_s; n_l$	it| itsub| pcg	$\langle C, X\rangle$	$b^T y$	R_P| R_D| gap	Time
fap09	15225 | 174; 14025	22| 120| 38.4	1.07978114 1	1.07978423 1	8.9-7| 9.6-7| -1.4-6	41
fap10	14479 | 183; 13754	23| 140| 57.4	9.67044948-3	9.74974306-3	1.5-7| 9.3-7| -7.8-5	1:18
fap11	24292 | 252; 23275	25| 148| 69.0	2.97000004-2	2.98373492-2	7.7-7| 6.0-7| **-1.3-4**	3:21
fap12	26462 | 369; 24410	25| 169| 81.3	2.73251961-1	2.73410714-1	6.0-7| 7.8-7| **-1.0-4**	9:07
fap25	322924 | 2118; 311044	24| 211| 84.8	1.28761356 1	1.28789892 1	3.2-6| 5.0-7| **-1.1-4**	10:53:22
fap36	1154467 | 4110; 1112293	17| 197| 87.4	6.98561787 1	6.98596286 1	7.7-7| 6.7-7| -2.5-5	65:25:07

TABLE 4
Results obtained by the boundary-point method in [19] *on the FAPs. The parameter* σ_0 *is set to* 1 *(better than* 0.1*).*

Problem	m | $n_s; n_l$	it	$\langle C, X\rangle$	$b^T y$	R_P| R_D| gap	Time
fap09	15225 | 174; 14025	2000	1.07978251 1	1.07982902 1	9.2-7| 9.8-6| -2.1-5	59
fap10	14479 | 183; 13754	2000	1.70252739-2	2.38972400-2	1.1-5| **1.1-4**| **-6.6-3**	1:25
fap11	24292 | 252; 23275	2000	4.22711513-2	5.94650102-2	8.8-6| **1.4-4**| **-1.6-2**	2:31
fap12	26462 | 369; 24410	2000	2.93446247-1	3.26163363-1	6.0-6| **1.5-4**| **-2.0-2**	4:37
fap25	322924 | 2118; 311044	2000	1.31895665 1	1.35910952 1	4.8-6| **2.0-4**| **-1.4-2**	8:04:00
fap36	1154467 | 4110; 1112293	2000	7.03339309 1	7.09606078 1	3.9-6| **1.4-4**| **-4.4-3**	46:59:28
fap09	15225 | 174; 14025	300	1.08257732 1	1.09208378 1	**1.7-4**| **7.2-4**| **-4.2-3**	09
fap10	14479 | 183; 13754	300	5.54148690-2	9.98476591-2	8.3-5| **6.9-4**| **-3.8-2**	12
fap11	24292 | 252; 23275	300	1.33930656-1	1.82368305-1	**2.4-4**| **7.9-4**| **-3.7-2**	22
fap12	26462 | 369; 24410	300	4.11473718-1	5.69735906-1	**1.2-4**| **8.4-4**| **-8.0-2**	41
fap25	322924 | 2118; 311044	300	1.47010392 1	1.69017693 1	**1.1-4**| **1.2-3**| **-6.8-2**	1:10:36
fap36	1154467 | 4110; 1112293	300	7.28509749 1	7.67389918 1	8.6-5| **8.9-4**| **-2.6-2**	6:53:36

It is interesting to note that for this collection, the SDP problems (D) and (P) are likely to be degenerate at the optimal solution $\bar{y}$ and $\overline{X}$, respectively. It is surprising that the SDPNAL algorithm can attain the required accuracy within moderate CPU time despite the fact that the problems may not satisfy the constraint nondegeneracy conditions (36) and (58) at the optimal solution $\bar{y}$ and $\overline{X}$.

The SDP problems arising from FAPs form a particularly difficult class of problems. Previous methods such as the spectral bundle (SB) method [12], the Burer–Monteiro–Zhang (BMZ) method (a log-barrier method applied to a nonlinear programming reformulation of (D)) [6], and the inexact IPM [39] largely fail to solve these SDP problems to satisfactory accuracy within moderate computer time. For example, the SB and BMZ methods took more than 50 and 3.3 hours, respectively, to solve `fap09` on an SGI Origin2000 computer using a single 300MHz R1200 processor. The inexact IPM [39] took more than 2.5 hours to solve the same problem on a 700MHz HP c3700 workstation. Comparatively, our SDPNAL algorithm took only 41 seconds to solve `fap09` to the same or better accuracy. In [20], the largest problem `fap36` was tested on the SB and BMZ methods using a 450MHz Sun Ultra 60 workstation. The SB and BMZ methods obtained the lower bounds of 63.77 and 63.78 for the optimal objective value after running for 4250 and 2036 hours, respectively. In contrast, our SDPNAL algorithm was able to solve `fap36` to a rather good accuracy in about 65 hours, and obtained the approximate optimal objective value of 69.85.

Table 5
Results for the SDPNAL algorithm on computing $\theta(G)$ in (75) for the maximum stable set problems.

Problem	m | $n_s; n_l$	it| itsub| pcg	$\langle C, X\rangle$	$b^T y$	R_P| R_D| gap	Time
theta103	62516 | 500;	18| 27| 10.7	2.25285688 1	2.25285667 1	4.4-8| 5.8-7| 4.6-8	1:00
theta104	87245 | 500;	17| 28| 11.2	1.33361400 1	1.33361379 1	6.1-8| 6.5-7| 7.6-8	58
theta123	90020 | 600;	18| 26| 10.9	2.46686513 1	2.46686492 1	3.3-8| 5.2-7| 4.1-8	1:34
theta162	127600 | 800;	17| 26| 10.2	3.70097353 1	3.70097324 1	3.6-8| 5.4-7| 3.8-8	2:53
c-fat200-1	18367 | 200;	8| 36| 20.3	1.19999983 1	1.19999962 1	1.5-7| 8.3-7| 8.5-8	09
hamming-10	23041 | 1024;	7| 9| 5.6	1.02399780 2	1.02400070 2	7.1-8| 7.1-7| -1.4-6	1:33
hamming-8-	16129 | 256;	4| 4| 4.8	2.56000007 1	2.55999960 1	2.8-9| 2.1-7| 9.0-8	02
hamming-9-	53761 | 512;	4| 6| 6.5	8.53333333 1	8.53333311 1	1.4-11| 3.9-8| 1.3-8	10
brock400-1	20078 | 400;	21| 25| 10.6	3.97018902 1	3.97018916 1	5.4-7| 9.9-7| -1.7-8	26
p-hat300-1	33918 | 300;	20| 84| 38.7	1.00679674 1	1.00679561 1	5.5-7| 9.4-7| 5.3-7	1:45
G43	9991 | 1000;	18| 27| 11.6	2.80624585 2	2.80624562 2	3.0-8| 4.6-7| 4.2-8	1:33
G44	9991 | 1000;	18| 28| 11.1	2.80583335 2	2.80583149 2	3.6-7| 9.2-7| 3.3-7	2:59
G45	9991 | 1000;	17| 26| 11.5	2.80185131 2	2.80185100 2	3.6-8| 5.8-7| 5.6-8	2:51
G46	9991 | 1000;	18| 26| 11.4	2.79837027 2	2.79836899 2	3.2-7| 9.1-7| 2.3-7	2:53
G47	9991 | 1000;	17| 27| 11.4	2.81893976 2	2.81893904 2	7.0-8| 9.3-7| 1.3-7	2:54
2dc.512	54896 | 512;	27| 258| 61.3	1.17732077 1	1.17690636 1	2.4-5| 5.0-7| **1.7-4**	32:16
1dc.1024	24064 | 1024;	26| 130| 64.0	9.59854968 1	9.59849281 1	1.4-6| 4.9-7| 2.9-6	41:26
1et.1024	9601 | 1024;	19| 117| 76.8	1.84226899 2	1.84226245 2	2.5-6| 3.5-7| 1.8-6	1:01:14
1tc.1024	7937 | 1024;	30| 250| 79.1	2.06305257 2	2.06304344 2	1.7-6| 6.3-7| 2.2-6	1:48:04
1zc.1024	16641 | 1024;	15| 22| 12.2	1.28666659 2	1.28666651 2	2.8-8| 3.0-7| 3.3-8	4:15
2dc.1024	169163 | 1024;	28| 219| 68.0	1.86426368 1	1.86388392 1	7.8-6| 6.8-7| 9.9-5	2:57:56
1dc.2048	58368 | 2048;	27| 154| 82.5	1.74729647 2	1.74729135 2	7.7-7| 4.0-7| 1.5-6	6:11:11
1et.2048	22529 | 2048;	22| 138| 81.6	3.42029313 2	3.42028707 2	6.9-7| 6.3-7| 8.8-7	7:13:55
1tc.2048	18945 | 2048;	26| 227| 78.5	3.74650769 2	3.74644820 2	3.3-6| 3.7-7| 7.9-6	9:52:09
1zc.2048	39425 | 2048;	13| 24| 14.0	2.37400485 2	2.37399909 2	1.5-7| 7.3-7| 1.2-6	45:16
2dc.2048	504452 | 2048;	27| 184| 67.1	3.06764717 1	3.06737001 1	3.7-6| 4.5-7| 4.4-5	15:13:19

6.3. SDP problems arising from relaxation of maximum stable set problems. For a graph G with edge set $\mathcal{E}$, the stability number $\alpha(G)$ is the cardinality of a maximal stable set of G, and $\alpha(G) := \{e^T x \,:\, x_i x_j = 0, (i,j) \in \mathcal{E}, x \in \{0,1\}^n\}$. It is known that $\alpha(G) \le \theta(G) \le \theta_+(G)$, where

$$\theta(G) = \max\{\langle ee^T,\, X\rangle \,:\, \langle E_{ij},\, X\rangle = 0, (i,j) \in \mathcal{E},\, \langle I,\, X\rangle = 1,\, X \succeq 0\}, \tag{75}$$

$$\theta_+(G) = \max\{\langle ee^T,\, X\rangle \,:\, \langle E_{ij},\, X\rangle = 0, (i,j) \in \mathcal{E},\, \langle I,\, X\rangle = 1,\, X \succeq 0,\, X \ge 0\}, \tag{76}$$

where $E_{ij} = e_i e_j^{\mathrm{T}} + e_j e_i^{\mathrm{T}}$ and e_i denotes column i of the identity matrix I. Note that for (76), the problem is reformulated as a standard SDP problem by replacing the constraint $X \ge 0$ by constraints $X - Y = 0$ and $Y \ge 0$. Thus such a reformulation introduces $n(n+1)/2$ additional linear equality constraints to the SDP.

Tables 5 and 6 list the results obtained by the SDPNAL algorithm for the SDP problems (75) and (76) arising from computing $\theta(G)$ and $\theta_+(G)$ for the maximum stable set problems, respectively. The first collection of graph instances in Table 5 consists of the randomly generated instances considered in [39], whereas the second collection is from the Second DIMACS Challenge on Maximum Clique Problems [42]. The last collection consists of graphs arising from coding theory, available from Sloane's web page [35]. Again, in the interest of saving space, we report the results for only 26 instances out of a total of 62 tested.

Table 6

Results for the SDPNAL algorithm on computing $\theta_+(G)$ in (76) for the maximum stable set problems.

Problem	$m - n_l \mid n_s; n_l$	it| itsub| pcg	$\langle C, X\rangle$	$b^T y$	R_P| R_D| gap	Time
theta103	62516 | 500; 125250	12| 38| 26.5	2.23774200 1	2.23774190 1	1.0-7| 9.3-7| 2.3-8	3:28
theta104	87245 | 500; 125250	14| 35| 22.0	1.32826023 1	1.32826068 1	8.1-7| 8.4-7| -1.6-7	2:35
theta123	90020 | 600; 180300	15| 43| 29.2	2.44951438 1	2.44951497 1	7.7-7| 8.5-7| -1.2-7	6:44
theta162	127600 | 800; 320400	14| 42| 26.2	3.67113362 1	3.67113729 1	8.1-7| 4.5-7| -4.9-7	11:24
c-fat200-1	18367 | 200; 20100	7| 48| 42.1	1.20000008 1	1.19999955 1	1.3-7| 9.5-7| 2.1-7	36
hamming-10	23041 | 1024; 524800	8| 17| 10.6	8.53334723 1	8.53334002 1	6.0-8| 7.9-7| 4.2-7	4:35
hamming-8-	16129 | 256; 32896	6| 7| 7.0	2.56000002 1	2.56000002 1	2.0-9| 5.1-9| -2.7-10	05
hamming-9-	53761 | 512; 131328	11| 18| 10.6	5.86666682 1	5.86666986 1	1.1-7| 4.4-7| -2.6-7	42
brock400-1	20078 | 400; 80200	14| 42| 26.4	3.93309197 1	3.93309200 1	9.5-7| 6.5-7| -3.5-9	1:45
p-hat300-1	33918 | 300; 45150	21| 123| 73.5	1.00202172 1	1.00202006 1	8.7-7| 7.2-7| 7.9-7	6:50
G43	9991 | 1000; 500500	9| 126| 52.2	2.79735847 2	2.79735963 2	9.1-7| 8.1-7| -2.1-7	52:00
G44	9991 | 1000; 500500	8| 122| 51.4	2.79746110 2	2.79746078 2	3.3-7| 6.2-7| 5.7-8	49:32
G45	9991 | 1000; 500500	9| 124| 52.0	2.79317531 2	2.79317544 2	9.3-7| 8.6-7| -2.4-8	50:25
G46	9991 | 1000; 500500	8| 112| 52.2	2.79032493 2	2.79032511 2	3.5-7| 9.6-7| -3.3-8	44:38
G47	9991 | 1000; 500500	9| 102| 53.1	2.80891719 2	2.80891722 2	4.7-7| 6.0-7| -5.1-9	40:27
2dc.512	54896 | 512; 131328	33| 513| 106.2	1.13946331 1	1.13857125 1	**2.1-4**| 7.7-7| **3.8-4**	2:25:15
1dc.1024	24064 | 1024; 524800	24| 260| 81.4	9.55539508 1	9.55512205 1	1.4-5| 6.9-7| 1.4-5	5:03:49
1et.1024	9601 | 1024; 524800	20| 198| 155.0	1.82075477 2	1.82071562 2	4.8-6| 7.0-7| 1.1-5	6:45:50
1tc.1024	7937 | 1024; 524800	27| 414| 124.6	2.04591268 2	2.04236122 2	**1.5-4**| 7.3-7| **8.7-4**	10:37:57
1zc.1024	16641 | 1024; 524800	11| 67| 38.1	1.27999936 2	1.27999977 2	6.4-7| 5.7-7| -1.6-7	40:13
2dc.1024	169163 | 1024; 524800	28| 455| 101.8	1.77416130 1	1.77149535 1	**1.6-4**| 6.2-7| **7.3-4**	11:57:25
1dc.2048	58368 | 2048; 2098176	20| 320| 73.0	1.74292685 2	1.74258827 2	1.9-5| 7.1-7| 9.7-5	35:52:44
1et.2048	22529 | 2048; 2098176	22| 341| 171.5	3.38193695 2	3.38166811 2	6.3-6| 5.7-7| 4.0-5	80:48:17
1tc.2048	18945 | 2048; 2098176	24| 381| 150.2	3.71592017 2	3.70575527 2	**3.5-4**| 7.9-7| **1.4-3**	73:56:01
1zc.2048	39425 | 2048; 2098176	11| 38| 29.3	2.37400054 2	2.37399944 2	2.5-7| 7.9-7| 2.3-7	2:13:04
2dc.2048	504452 | 2048; 2098176	27| 459| 53.4	2.89755241 1	2.88181157 1	**1.3-4**| 7.2-7| **2.7-3**	45:21:42

Observe that the SDPNAL algorithm is not able to achieve the required accuracy level for some of the SDP problems from Sloane's collection. It is not surprising that this may happen because many of these SDP problems are degenerate at the optimal solution. For example, the problem `2dc.512` is degenerate at the optimal solution $\bar{y}$ even though it is nondegenerate at the optimal solution $\overline{X}$.

In [19], the performance of the boundary-point method was compared with that of the iterative solver–based primal-dual IPM in [39], as well as the iterative solver–based modified barrier method in [17], on a subset of the large SDP problems arising from the first collection of random graphs. The conclusion was that the boundary-point method was 5–10 times faster than the methods in [39] and [17]. Since the SDPNAL algorithm is at least as efficient as the boundary-point method on the theta problems for random graphs (not reported here in the interest of saving space), it is safe to assume that the SDPNAL algorithm would be at least 5–10 times faster than the methods in [39] and [17]. Note that the SDPNAL algorithm is more efficient than the boundary-point method on the collection of graphs from DIMACS. For example, the SDPNAL algorithm takes less than 100 seconds to solve the problem `G43` in Table 5 to an accuracy of less than 10^{-6}, while the boundary-point method (with $\sigma_0 = 0.1$) takes more than 3900 seconds to achieve an accuracy of 1.5×10^{-5}. Such a result for `G43` is not surprising because the rank of the optimal X (equal to 58) is much smaller than n, and, as already mentioned in [24], the boundary-point method typically would perform poorly under such a situation.

7. Applications to quadratic assignment and binary integer quadratic programming problems. In this section, we apply our SDPNAL algorithm to compute lower bounds for quadratic assignment problems (QAPs) and binary integer quadratic (BIQ) problems through SDP relaxations. Our purpose here is to demonstrate that the SDPNAL algorithm can potentially be very efficient in solving large SDP problems (and hence in computing bounds) arising from hard combinatorial problems.

Let Π be the set of $n \times n$ permutation matrices. Given matrices $A, B \in \Re^{n\times n}$, the quadratic assignment problem is

$$v^*_{\rm QAP} := \min\{\langle X, AXB\rangle \,:\, X \in \Pi\}. \tag{77}$$

For a matrix $X = [x_1, \ldots, x_n] \in \Re^{n\times n}$, we will identify it with the n^2-vector $x = [x_1; \ldots; x_n]$. For a matrix $Y \in R^{n^2\times n^2}$, we let Y^{ij} be the $n \times n$ block corresponding to $x_i x_j^T$ in the matrix xx^T. It is shown in [23] that $v^*_{\rm QAP}$ is bounded below by the following number:

$$\begin{array}{rl} v := \min & \langle B \otimes A,\, Y\rangle \\ \text{subject to (s.t.)} & \displaystyle\sum_{i=1}^{n} Y^{ii} = I,\ \langle I,\, Y^{ij}\rangle = \delta_{ij} \quad \forall\, 1 \le i \le j \le n, \\ & \langle E,\, Y^{ij}\rangle = 1 \quad \forall\, 1 \le i \le j \le n, \\ & Y \succeq 0,\ Y \ge 0, \end{array} \tag{78}$$

where E is the matrix of ones, and $\delta_{ij} = 1$ if $i = j$, and 0 otherwise. There are $3n(n+1)/2$ equality constraints in (78). But two of them are actually redundant, and we remove them when solving the standard SDP problem generated from (78). Note that [23] actually used the constraint $\langle E, Y\rangle = n^2$ in place of the last set of the equality constraints in (78). But we prefer to use the formulation here because the associated SDP problem has slightly better numerical behavior. Note also that the SDP problems (78) typically do not satisfy the constraint nondegenerate conditions (36) and (58) at the optimal solutions.

In our experiment, we apply the SDPNAL algorithm to the dual of (78), and hence any dual feasible solution would give a lower bound for (78). But in practice, our algorithm delivers only an approximately feasible dual solution $\tilde{y}$. We therefore apply the procedure given in [14, Theorem 2] to $\tilde{y}$ to construct a valid lower bound for (78), which we denote by $\underline{v}$.

Table 7 lists the results of the SDPNAL algorithm on the quadratic assignment instances (78). The details of the table are the same as for Table 1 except that the objective values are replaced by the best known upper bound on (77) under the column "best upper bound" and the lower bound $\underline{v}$. The entries under the column under "%gap" are calculated as follows:

$$\%\text{gap} \;=\; \frac{\text{best upper bound} - \underline{v}}{\text{best upper bound}} \times 100\%.$$

In the table, we report the results for 45 instances out of 85 tested. We compare our results with those obtained in [5], which used a dedicated augmented Lagrangian algorithm to solve the SDP problem arising from applying the lift-and-project procedure of Lovász and Schrijver to (77). As the augmented Lagrangian algorithm in

Table 7

Results for the SDPNAL algorithm on the QAPs. The entries under the column "%gap" are calculated with respect to the best solution listed, which is known to be optimal unless the symbol (†) is prefixed.

Problem	$m - n_l$	$n_s; n_l$	it	itsub	pcg	Best upper bound	Lower bound $\underline{v}$	R_P	R_D	%gap	Time
bur26a	1051	676; 228826	27	389	105.9	5.42667000 6	5.42577700 6	**2.9-3**	2.8-7	0.016	4:28:43
bur26b	1051	676; 228826	25	358	92.3	3.81785200 6	3.81663900 6	**2.3-3**	6.1-7	0.032	3:23:39
bur26c	1051	676; 228826	26	421	107.5	5.42679500 6	5.42593600 6	**3.9-3**	4.7-7	0.016	4:56:09
bur26d	1051	676; 228826	27	424	102.3	3.82122500 6	3.81982900 6	**3.8-3**	5.0-7	0.037	4:21:32
bur26e	1051	676; 228826	27	573	100.0	5.38687900 6	5.38683200 6	**7.5-3**	1.7-7	0.001	5:34:39
bur26f	1051	676; 228826	25	534	100.9	3.78204400 6	3.78184600 6	**3.1-3**	6.2-7	0.005	5:32:51
bur26g	1051	676; 228826	24	422	91.0	1.01171720 7	1.01167630 7	**3.8-3**	6.6-7	0.004	3:33:58
bur26h	1051	676; 228826	24	450	96.8	7.09865800 6	7.09856700 6	**2.0-3**	2.3-7	0.001	3:53:22
chr22a	757	484; 117370	26	467	116.7	6.15600000 3	6.15600000 3	**2.3-3**	9.3-8	0.000	1:50:37
chr22b	757	484; 117370	26	465	106.4	6.19400000 3	6.19400000 3	**1.8-3**	6.9-8	0.000	1:47:16
chr25a	973	625; 195625	26	462	84.7	3.79600000 3	3.79600000 3	**1.9-3**	1.4-7	0.000	3:20:35
esc32a	1582	1024; 524800	26	232	101.9	† 1.30000000 2	1.04000000 2	2.5-5	7.8-7	20.000	4:48:55
esc32b	1582	1024; 524800	22	201	99.4	† 1.68000000 2	1.32000000 2	**1.7-4**	7.8-7	21.429	3:52:36
esc32c	1582	1024; 524800	30	479	140.2	† 6.42000000 2	6.16000000 2	**6.5-4**	2.1-7	4.050	11:12:30
esc32d	1582	1024; 524800	25	254	132.0	† 2.00000000 2	1.91000000 2	5.3-7	5.6-7	4.500	5:43:54
esc32e	1582	1024; 524800	15	46	58.2	2.00000000 0	2.00000000 0	2.2-7	1.1-7	0.000	31:11
esc32f	1582	1024; 524800	15	46	58.2	2.00000000 0	2.00000000 0	2.2-7	1.1-7	0.000	31:13
esc32g	1582	1024; 524800	15	38	50.7	6.00000000 0	6.00000000 0	1.7-7	3.2-7	0.000	23:25
esc32h	1582	1024; 524800	30	403	113.3	† 4.38000000 2	4.23000000 2	**9.9-4**	3.0-7	3.425	8:05:32
kra30a	1393	900; 405450	27	313	68.0	8.89000000 4	8.64280000 4	**4.5-4**	6.5-7	2.781	4:08:17
kra30b	1393	900; 405450	28	289	68.9	9.14200000 4	8.74500000 4	**3.1-4**	7.4-7	4.343	3:50:35
kra32	1582	1024; 524800	31	307	78.6	8.89000000 4	8.52980000 4	**4.6-4**	6.0-7	4.052	6:43:41
lipa30a	1393	900; 405450	20	252	78.2	1.31780000 4	1.31780000 4	2.5-7	1.1-10	0.000	3:41:44
lipa30b	1393	900; 405450	18	83	80.8	1.51426000 5	1.51426000 5	6.9-7	3.3-8	0.000	1:23:34
lipa40a	2458	1600; 1280800	22	324	81.7	3.15380000 4	3.15380000 4	4.1-7	4.6-11	0.000	21:02:51
lipa40b	2458	1600; 1280800	19	121	76.6	4.76581000 5	4.76581000 5	3.9-6	1.3-8	0.000	7:24:25
nug22	757	484; 117370	28	369	86.0	3.59600000 3	3.52200000 3	**3.1-4**	5.9-7	2.058	1:21:58
nug24	898	576; 166176	29	348	63.7	3.48800000 3	3.39600000 3	**1.8-4**	3.6-7	2.638	1:33:59
nug25	973	625; 195625	27	335	60.2	3.74400000 3	3.62100000 3	**1.8-4**	3.0-7	3.285	1:41:49
nug27	1132	729; 266085	29	380	80.1	5.23400000 3	5.12400000 3	**1.3-4**	4.5-7	2.102	3:31:50
nug28	1216	784; 307720	26	329	80.5	5.16600000 3	5.02000000 3	**2.4-4**	6.3-7	2.826	3:36:38
nug30	1393	900; 405450	27	360	61.4	6.12400000 3	5.94400000 3	**1.3-4**	3.3-7	2.939	4:22:09
ste36a	1996	1296; 840456	26	318	93.8	9.52600000 3	9.23600000 3	**1.7-4**	4.1-7	3.044	15:09:10
ste36b	1996	1296; 840456	29	348	101.0	1.58520000 4	1.56030000 4	**1.8-3**	4.3-7	1.571	19:05:19
ste36c	1996	1296; 840456	28	360	105.3	8.23911000 6	8.11864500 6	**6.3-4**	4.0-7	1.462	19:56:15
tai25a	973	625; 195625	27	194	77.3	1.16725600 6	1.01301000 6	8.0-7	7.9-7	13.214	1:17:54
tai25b	973	625; 195625	29	408	70.4	3.44355646 8	3.33685462 8	**2.6-3**	6.2-7	3.099	2:33:26
tai30a	1393	900; 405450	27	207	82.4	† 1.81814600 6	1.70578200 6	8.1-5	2.0-7	6.180	3:35:03
tai30b	1393	900; 405450	30	421	71.6	6.37117113 8	5.95926267 8	**1.4-3**	4.9-7	6.465	6:26:30
tai35a	1888	1225; 750925	28	221	81.0	2.42200200 6	2.21523000 6	**1.5-4**	5.0-7	8.537	8:09:44
tai35b	1888	1225; 750925	28	401	58.3	2.83315445 8	2.68328155 8	**8.7-4**	6.4-7	5.290	10:33:10
tai40a	2458	1600; 1280800	27	203	85.1	3.13937000 6	2.84184600 6	7.5-5	5.3-7	9.477	15:25:52
tai40b	2458	1600; 1280800	30	362	74.1	6.37250948 8	6.06880822 8	**1.7-3**	4.9-7	4.766	23:32:56
tho30	1393	900; 405450	27	315	61.1	1.49936000 5	1.43267000 5	**2.4-4**	7.3-7	4.448	3:41:26
tho40	2458	1600; 1280800	27	349	60.9	† 2.40516000 5	2.26161000 5	**2.0-4**	6.5-7	5.968	17:13:24

[5] is designed specifically for the SDP problems arising from the lift-and-project procedure, the details of that algorithm are very different from those of our SDPNAL algorithm. Note that the algorithm in [5] was implemented in C (with LAPACK library), and the results reported were obtained from a 2.4 GHz Pentium 4 PC with 1

GB of RAM (which is about 50% slower than our PC). By comparing the results in Table 7 against those in [5, Tables 6 and 7], we can safely conclude that the SDPNAL algorithm applied to (78) is superior in terms of CPU time and the accuracy of the approximate optimal solution computed. Take, for example, the SDP problems corresponding to the QAPs `nug30` and `tai35b`: the SDPNAL algorithm obtains the lower bounds with %gap of 2.939 and 5.318 in $15,729$ and $37,990$ seconds, respectively, whereas the algorithm in [5] computes the bounds with %gap of 3.10 and 15.42 in $127,011$ and $430,914$ seconds, respectively.

The paper [5] also solved the lift-and-project SDP relaxations for the maximum stable set problems (denoted as N_+ and known to be at least as strong as θ_+) using a dedicated augmented Lagrangian algorithm. By comparing the results in Table 6 against those in [5, Table 4], we can again conclude that the SDPNAL algorithm applied to (76) is superior in terms of CPU time and the accuracy of the approximate optimal solution computed. Take, for example, the SDP problems corresponding to the graphs `p-hat300-1` and `c-fat200-1`: the SDPNAL algorithm obtains the upper bounds of $\theta_+ = 10.0202$ and $\theta_+ = 12.0000$ in 410 and 36 seconds, respectively, whereas the algorithm in [5] computes the bounds of $N_+ = 18.6697$ and $N_+ = 14.9735$ in $322,287$ and $126,103$ seconds, respectively.

The BIQ problem we consider is the following:

$$v^*_{\rm BIQ} := \min\{x^TQx \,:\, x \in \{0,1\}^n\}, \tag{79}$$

where Q is a symmetric matrix (nonpositive semidefinite) of order n. A natural SDP relaxation of (79) is the following:

$$\begin{array}{ll} \min & \langle Q, Y\rangle \\ \text{s.t.} & \operatorname{diag}(Y) - y = 0, \quad \alpha = 1, \\ & \begin{bmatrix} Y & y \\ y^T & \alpha \end{bmatrix} \succeq 0, \quad Y \geq 0,\ y \geq 0. \end{array} \tag{80}$$

Table 8 lists the results obtained by the SDPNAL algorithm on the SDP problems (80) arising from the BIQ instances described in [43]. Here we report only the results for 30 instances out of 165 tested. It is interesting to note that the lower bound obtained from (80) is within 10% of the optimal value $v^*_{\rm BIQ}$ for all the instances tested, and for the instances `gka1b`–`gka9b`, the lower bounds are actually equal to $v^*_{\rm BIQ}$.

8. Conclusion. In this paper, we introduced a Newton-CG augmented Lagrangian algorithm for solving SD problems (D) and (P) and analyzed its convergence and rate of convergence. Our convergence analysis is based on classical results of proximal point methods [32, 33] along with recent developments in perturbation analysis of the problems under consideration. Extensive numerical experiments conducted on a variety of large-scale SDP problems demonstrated that our algorithm is very efficient. This opens a way to attack problems in which a fast solver for large-scale SDP problems is crucial, for example, in applications within a branch-and-bound algorithm for solving hard combinatorial problems such as the QAPs.

Acknowledgment. We thank Brian Borchers for providing us with the SDP data for the maximum stable set problems collected by Neil Sloane [35].

TABLE 8
Results for the SDPNAL algorithm on the BIQ problems. The entries under the column "%gap" are calculated with respect to the best solution listed, which is known to be optimal unless the symbol (†) *is prefixed.*

Problem	$m - n_l \mid n_s; n_l$	it| itsub| pcg	Best upper bound	Lower bound $\underline{v}$	R_P| R_D| %gap	Time
be200.3.1	201 | 201; 20301	29| 615| 89.7	-2.54530000 4	-2.77160000 4	5.6-7| 5.0-7| 8.891	10:29
be200.3.3	201 | 201; 20301	29| 507| 120.8	-2.80230000 4	-2.94780000 4	5.6-5| 5.7-7| 5.192	12:09
be200.3.5	201 | 201; 20301	28| 466| 116.2	-2.63550000 4	-2.80730000 4	1.4-6| 5.5-7| 6.519	10:38
be200.3.7	201 | 201; 20301	29| 534| 93.9	-3.04830000 4	-3.16200000 4	1.1-6| 5.8-7| 3.730	9:43
be200.3.9	201 | 201; 20301	28| 482| 87.1	-2.46830000 4	-2.64370000 4	3.2-5| 3.7-7| 7.106	8:28
be200.8.1	201 | 201; 20301	28| 489| 97.5	-4.85340000 4	-5.08690000 4	3.7-5| 6.2-7| 4.811	9:41
be200.8.3	201 | 201; 20301	28| 476| 116.1	-4.32070000 4	-4.62540000 4	5.8-7| 9.2-7| 7.052	10:53
be200.8.5	201 | 201; 20301	28| 521| 93.8	-4.14820000 4	-4.42710000 4	1.7-5| 7.7-7| 6.723	9:53
be200.8.7	201 | 201; 20301	27| 248| 92.6	-4.68280000 4	-4.93530000 4	4.7-7| 6.8-7| 5.392	4:30
be200.8.9	201 | 201; 20301	29| 543| 115.6	-4.32410000 4	-4.54950000 4	5.8-6| 3.8-7| 5.213	12:16
be250.1	251 | 251; 31626	29| 532| 94.7	-2.40760000 4	-2.51190000 4	4.0-5| 4.6-7| 4.332	16:41
be250.3	251 | 251; 31626	28| 561| 95.7	-2.29230000 4	-2.40000000 4	2.9-5| 6.0-7| 4.698	17:17
be250.5	251 | 251; 31626	29| 463| 98.1	-2.10570000 4	-2.23740000 4	9.3-5| 4.4-7| 6.254	14:30
be250.7	251 | 251; 31626	28| 507| 84.7	-2.40950000 4	-2.51190000 4	5.9-5| 7.1-7| 4.250	14:00
be250.9	251 | 251; 31626	28| 589| 85.8	-2.00510000 4	-2.13970000 4	**1.1-4**| 3.6-7| 6.713	17:13
bqp250-1	251 | 251; 31626	28| 483| 117.7	-4.56070000 4	-4.76630000 4	3.9-7| 6.6-7| 4.508	17:42
bqp250-3	251 | 251; 31626	28| 296| 116.4	-4.90370000 4	-5.10770000 4	9.9-7| 7.9-7| 4.160	10:36
bqp250-5	251 | 251; 31626	28| 570| 103.7	-4.79610000 4	-5.00040000 4	4.4-5| 6.9-7| 4.260	19:03
bqp250-7	251 | 251; 31626	30| 429| 126.3	-4.67570000 4	-4.89220000 4	8.2-7| 5.9-7| 4.630	16:36
bqp250-9	251 | 251; 31626	29| 453| 117.0	-4.89160000 4	-5.14970000 4	3.7-7| 3.9-7| 5.276	16:12
bqp500-1	501 | 501; 125751	30| 357| 117.8	-1.16586000 5	-1.25965000 5	2.9-7| 5.5-7| 8.045	1:00:59
bqp500-3	501 | 501; 125751	30| 363| 118.9	-1.30812000 5	-1.38454000 5	4.4-7| 4.0-7| 5.842	1:01:47
bqp500-5	501 | 501; 125751	30| 539| 119.6	-1.25487000 5	-1.34092000 5	4.5-5| 2.5-7| 6.857	1:36:43
bqp500-7	501 | 501; 125751	31| 648| 87.7	-1.22201000 5	-1.31492000 5	8.1-5| 5.7-7| 7.603	1:25:26
bqp500-9	501 | 501; 125751	30| 612| 92.7	-1.20798000 5	-1.30289000 5	9.5-5| 7.3-7| 7.857	1:24:40
gka2e	201 | 201; 20301	29| 367| 103.4	-2.33950000 4	-2.49170000 4	4.7-7| 4.3-7| 6.506	7:23
gka4e	201 | 201; 20301	29| 512| 113.0	-3.55940000 4	-3.72250000 4	1.2-5| 4.2-7| 4.582	11:25
gka1f	501 | 501; 125751	30| 563| 102.8	†-6.11940000 4	-6.55590000 4	9.9-5| 5.2-7| 7.133	1:28:54
gka3f	501 | 501; 125751	30| 523| 120.4	†-1.38035000 5	-1.50152000 5	2.8-5| 6.7-7| 8.778	1:31:34
gka5f	501 | 501; 125751	31| 665| 90.5	†-1.90507000 5	-2.06916000 5	6.6-6| 7.1-7| 8.613	1:25:48

REFERENCES

[1] F. ALIZADEH, J. P. A. HAEBERLY, AND O. L. OVERTON, *Complementarity and nondegeneracy in semidefinite programming*, Math. Programming, 77 (1997), pp. 111–128.

[2] V. I. ARNOLD, *On matrices depending on parameters*, Russian Math. Surveys, 26 (1971), pp. 29–43.

[3] Z.-J. BAI, D. CHU, AND D. SUN, *A dual optimization approach to inverse quadratic eigenvalue problems with partial eigenstructure*, SIAM J. Sci. Comput., 29 (2007), pp. 2531–2561.

[4] J. F. BONNANS AND A. SHAPIRO, *Perturbation Analysis of Optimization Problems*, Springer-Verlag, New York, 2000.

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

[6] S. BURER, R. MONTEIRO, AND Y. ZHANG, *A computational study of a gradient-based log-barrier algorithm for a class of large scale SDPs*, Math. Program., 95 (2003), pp. 359–379.

[7] Z. X. CHAN AND D. SUN, *Constraint nondegeneracy, strong regularity, and nonsingularity in semidefinite programming*, SIAM J. Optim., 19 (2008), pp. 370–396.

[8] F. H. CLARKE, *Optimization and Nonsmooth Analysis*, John Wiley and Sons, New York, 1983.

[9] A. EISENBLÄTTER, M. GRÖTSCHEL, AND A. M. C. A. KOSTER, *Frequency planning and ramification of coloring*, Discuss. Math. Graph Theory, 22 (2002), pp. 51–88.

[10] F. FACCHINEI, *Minimization of SC^1 functions and the Maratos effect*, Oper. Res. Lett., 17 (1995), pp. 131–137.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, London, 1996.
[12] C. Helmberg, *Numerical Evaluation of SBmethod*, Math. Program., 95 (2003), pp. 381–406.
[13] F. Jarre and F. Rendl, *An augmented primal-dual method for linear conic programs*, SIAM J. Optim., 19 (2008), pp. 808–823.
[14] C. Jansson, *Termination and Verification for Ill-Posed Semidefinite Programs*, Technical report, Informatik III, TU Hamburg-Harburg, Hamburg, Germany, 2005.
[15] D. Johnson, G. Pataki, and Farid Alizadeh, *The Seventh DIMACS Implementation Challenge: Semidefinite and Related Optimization Problems*, Rutgers University, New Brunswick, NJ, 2000; available online at http://dimacs.rutgers.edu/Challenges/.
[16] M. Kočvara and M. Stingl, *PENNON: A code for convex nonlinear and semidefinite programming*, Optim. Methods Softw., 18 (2003), pp. 317–333.
[17] M. Kočvara and M. Stingl, *On the solution of large-scale SDP problems by the modified barrier method using iterative solvers*, Math. Program., 109 (2007), pp. 413–444.
[18] B. Kummer, *Newton's method for nondifferentiable functions*, in Advances in Mathematical Optimization, J. Guddat et al., eds., Akademie-Verlag, Berlin, 1988, pp. 114–125.
[19] J. Malick, J. Povh, F. Rendl, and A. Wiegele, *Regularization methods for semidefinite programming*, SIAM J. Optim., 20 (2009), pp. 336–356.
[20] H. Mittelmann, *An independent benchmarking of SDP and SOCP solvers*, Math. Program., 95 (2003), pp. 407–430.
[21] J. S. Pang and L. Qi, *A globally convergent Newton method of convex* SC^1 *minimization problems*, J. Optim. Theory Appl., 85 (1995), pp. 633–648.
[22] J. S. Pang, D. F. Sun, and J. Sun, *Semismooth homeomorphisms and strong stability of semidefinite and Lorentz complementarity problems*, Math. Oper. Res., 28 (2003), pp. 272–284.
[23] J. Povh and F. Rendl, *Copositive and semidefinite relaxations of the quadratic assignment problem*, Discrete Optim., 6 (2009), pp. 231–241.
[24] J. Povh, F. Rendl, and A. Wiegele, *A boundary point method to solve semidefinite programs*, Computing, 78 (2006), pp. 277–286.
[25] 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.
[26] L. Qi and J. Sun, *A nonsmooth version of Newton's method*, Math. Programming, 58 (1993), pp. 353–367.
[27] S. M. Robinson, *Local structure of feasible sets in nonlinear programming.* II. *Nondegeneracy*, Math. Programming Stud., No. 22 (1984), pp. 217–230.
[28] S. M. Robinson, *Local structure of feasible sets in nonlinear programming.* III. *Stability and sensitivity*, Math. Programming Stud., No. 30 (1987), pp. 45–66.
[29] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.
[30] R. T. Rockafellar, *A dual approach to solving nonlinear programming problems by unconstrained optimization*, Math. Programming, 5 (1973), pp. 354–373.
[31] R. T. Rockafellar, *Conjugate Duality and Optimization*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 16, SIAM, Philadelphia, 1974.
[32] R. T. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM J. Control Optim., 14 (1976), pp. 877–898.
[33] R. T. Rockafellar, *Augmented Lagrangians and applications of the proximal point algorithm in convex programming*, Math. Oper. Res., 1 (1976), pp. 97–116.
[34] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.
[35] N. Sloane, *Challenge Problems: Independent Sets in Graphs*, http://www.research.att.com/~njas/doc/graphs.html.
[36] D. F. Sun and J. Sun, *Semismooth matrix valued functions*, Math. Oper. Res., 27 (2002), pp. 150–169.
[37] 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.
[38] L. N. Trefethen and D. Bau, III, *Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
[39] K.-C. Toh, *Solving large scale semidefinite programs via an iterative solver on the augmented systems*, SIAM J. Optim., 14 (2003), pp. 670–698.
[40] K. C. Toh, *An inexact primal-dual path-following algorithm for convex quadratic SDP*, Math. Program., 112 (2007), pp. 221–254.
[41] K.-C. Toh and M. Kojima, *Solving some large scale semidefinite programs via the conjugate residual method*, SIAM J. Optim., 12 (2002), pp. 669–691.

[42] M. Trick, V. Chvatal, W. Cook, D. Johnson, C. McGeoch, and R. Tarjan, *The Second DIMACS Implementation Challenge: NP Hard Problems: Maximum Clique, Graph Coloring, and Satisfiability*, Rutgers University, New Brunswick, NJ, 1992; available online at http://dimacs.rutgers.edu/Challenges/.

[43] A. Wiegele, *Biq Mac Library—A Collection of Max-Cut and Quadratic* 0-1 *Programming Instances of Medium Size*, preprint, Alpen-Adria-Universität Klagenfurt, Klagenfurt, Austria, 2007.

[44] E. H. Zarantonello, *Projections on convex sets in Hilbert space and spectral theory.* I *and* II, in Contributions to Nonlinear Functional Analysis, E. H. Zarantonello, ed., Academic Press, New York, 1971, pp. 237–341 and 343–424.

[45] X. Y. Zhao, D. Sun, and K. C. Toh, *A Newton-CG Augmented Lagrangian Method for Semidefinite Programming*, preprint, National University of Singapore, 2008; available online at http://www.math.nus.edu.sg/~mattohkc/papers/SDPNAL.pdf.