Math. Prog. Comp. (2018) 10:703–743
https://doi.org/10.1007/s12532-018-0137-6

FULL LENGTH PAPER

QSDPNAL: a two-phase augmented Lagrangian method for convex quadratic semidefinite programming

Xudong Li[1] · **Defeng Sun**[2] · **Kim-Chuan Toh**[3]

Received: 19 December 2016 / Accepted: 19 March 2018 / Published online: 13 April 2018

Abstract In this paper, we present a two-phase augmented Lagrangian method, called QSDPNAL, for solving convex quadratic semidefinite programming (QSDP) problems with constraints consisting of a large number of linear equality and inequality constraints, a simple convex polyhedral set constraint, and a positive semidefinite cone constraint. A first order algorithm which relies on the inexact Schur complement based decomposition technique is developed in QSDPNAL-Phase I with the aim of solving a QSDP problem to moderate accuracy or using it to generate a reasonably good initial point for the second phase. In QSDPNAL-Phase II, we design an aug-

An earlier version of this paper was made available in arXiv as arXiv:1512.08872. The software that was reviewed as part of this submission has been issued the Digital Object Identifier https://doi.org/10.5281/zenodo.1206980.
D. Sun: On leave from Department of Mathematics, National University of Singapore. This author's research is partially supported by a start-up research grant from the Hong Kong Polytechnic University.
K.-C. Toh: This author's research is supported in part by the Ministry of Education, Singapore, Academic Research Fund under Grant R-146-000-256-114.

✉ Xudong Li
xudongl@princeton.edu

Defeng Sun
defeng.sun@polyu.edu.hk

Kim-Chuan Toh
mattohkc@nus.edu.sg

[1] Department of Operations Research and Financial Engineering, Princeton University, Sherrerd Hall, Princeton, NJ 08544, USA

[2] Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

[3] Department of Mathematics, and Institute of Operations Research and Analytics, National University of Singapore, 10 Lower Kent Ridge Road, Singapore 119076, Singapore

mented Lagrangian method (ALM) wherein the inner subproblem in each iteration is solved via inexact semismooth Newton based algorithms. Simple and implementable stopping criteria are designed for the ALM. Moreover, under mild conditions, we are able to establish the rate of convergence of the proposed algorithm and prove the R-(super)linear convergence of the KKT residual. In the implementation of QSDPNAL, we also develop efficient techniques for solving large scale linear systems of equations under certain subspace constraints. More specifically, simpler and yet better conditioned linear systems are carefully designed to replace the original linear systems and novel shadow sequences are constructed to alleviate the numerical difficulties brought about by the crucial subspace constraints. Extensive numerical results for various large scale QSDPs show that our two-phase algorithm is highly efficient and robust in obtaining accurate solutions. The software reviewed as part of this submission was given the DOI (Digital Object Identifier) https://doi.org/10.5281/zenodo.1206980.

Keywords Quadratic semidefinite programming · Schur complement · Augmented Lagrangian · Inexact semismooth Newton method

Mathematics Subject Classification 90C06 · 90C20 · 90C22 · 90C25 · 65F10

1 Introduction

Let $\mathcal{S}^n_+$ and $\mathcal{S}^n_{++}$ be the cones of positive semidefinite and positive definite matrices, respectively, in the space of $n \times n$ symmetric matrices $\mathcal{S}^n$ endowed with the standard trace inner product $\langle \cdot, \cdot \rangle$ and Frobenius norm $\| \cdot \|$. We consider the following convex quadratic semidefinite programming (QSDP) problem:

$$(\mathbf{P}) \quad \min \left\{ \frac{1}{2}\langle X, \mathcal{Q}X \rangle + \langle C, X \rangle \mid \mathcal{A}X = b, \ X \in \mathcal{S}^n_+ \cap \mathcal{K} \right\},$$

where $\mathcal{Q} : \mathcal{S}^n \to \mathcal{S}^n$ is a self-adjoint positive semidefinite linear operator, $\mathcal{A} : \mathcal{S}^n \to \Re^m$ is a linear map whose adjoint is denoted as $\mathcal{A}^*$, $C \in \mathcal{S}^n$, $b \in \Re^m$ are given data, $\mathcal{K}$ is a simple nonempty closed convex polyhedral set in $\mathcal{S}^n$, such as $\mathcal{K} = \{X \in \mathcal{S}^n \mid L \leq X \leq U\}$ with $L, U \in \mathcal{S}^n$ being given matrices. The main objective of this paper is to design and analyse efficient algorithms for solving (**P**) and its dual. We are particularly interested in the case where the dimensions n and/or m are large, and it may be impossible to explicitly store or compute the matrix representation of $\mathcal{Q}$. For example, if $\mathcal{Q} = H \otimes H$ is the Kronecker product of a dense matrix $H \in \mathcal{S}^n_+$ with itself, then it would be extremely expensive to store the matrix representation of $\mathcal{Q}$ explicitly when n is larger than, say, 500. As far as we are aware of, the best solvers currently available for solving (**P**) are based on inexact primal-dual interior-point methods [31]. However, they are highly inefficient for solving large scale problems as interior-point methods have severe inherent ill-conditioning limitations which would make the convergence of a Krylov subspace iterative solver employed to compute the search directions to be extremely slow. While sophisticated preconditioners have been constructed in [31] to alleviate the ill-conditioning, the improvement is however not

dramatic enough for the algorithm to handle large scale problems comfortably. On the other hand, an interior-point method which employs a direct solver to compute the search directions is prohibitively expensive for solving (**P**) since the cost is at least $O((m+n^2)^3)$ arithmetic operations per iteration. It is safe to say that there is currently no solver which can efficiently handle large scale QSDP problems of the form (**P**) and our paper precisely aims to provide an efficient and robust solver for (**P**).

The algorithms which we will design later are based on the augmented Lagrangian function for the dual of (**P**) (in its equivalent minimization form):

$$(\mathbf{D}) \quad \min \left\{ \delta^*_{\mathcal{K}}(-Z) + \frac{1}{2}\langle W,\ \mathcal{Q}W\rangle - \langle b,\ y\rangle \ \middle|\ \begin{array}{l} Z - \mathcal{Q}W + S + \mathcal{A}^* y = C, \\ S \in \mathcal{S}^n_+,\ \ W \in \mathcal{W},\ \ y \in \Re^m, Z \in \mathcal{S}^n \end{array} \right\},$$

where $\mathcal{W}$ is any subspace of $\mathcal{S}^n$ containing the range space of $\mathcal{Q}$ (denoted as $\mathrm{Ran}(\mathcal{Q})$), $\delta^*_{\mathcal{K}}(\cdot)$ is the Fenchel conjugate of the indicator function $\delta_{\mathcal{K}}(\cdot)$.

Due to its great potential in applications and mathematical elegance, QSDP has been studied quite actively both from the theoretical and numerical aspects [1,11,14,15,19,23,31,32]. For the recent theoretical developments, one may refer to [7,10,22,30] and the references therein. Here we focus on the numerical aspect and we will next briefly review some of the methods available for solving QSDP problems. Toh et al. [32] and Toh [31] proposed inexact primal-dual path-following interior-point methods to solve the special class of convex QSDP without the constraint in $\mathcal{K}$. In theory, these methods can be used to solve QSDP problems with inequality constraints and constraint in $\mathcal{K}$ by reformulating the problems into the required standard form. However, as already mentioned above, in practice interior-point methods are not efficient for solving QSDP problems beyond moderate scales either due to the extremely high computational cost per iteration or the inherent ill-conditioning of the linear systems governing the search directions. In [34], Zhao designed a semismooth Newton-CG augmented Lagrangian (NAL) method and analyzed its convergence for solving the primal QSDP problem (**P**). However, the NAL algorithm often encounters numerical difficulty (due to singular or nearly singular generalized Hessian) when the polyhedral set constraint $X \in \mathcal{K}$ is present. Subsequently, Jiang et al. [13] proposed an inexact accelerated proximal gradient method for least squares semidefinite programming with only equality constraints where the objective function in (**P**) is expressed explicitly in the form of $\|\mathcal{B}X - d\|^2$ for some given linear map $\mathcal{B}$.

More recently, inspired by the successes achieved in [28,33] for solving linear SDP problems with nonnegative constraints, Li et al. [18] proposed a first-order algorithm, known as the Schur complement based semi-proximal alternating direction method of multipliers (SCB-sPADMM), for solving the dual form (**D**) of QSDP. As far as we aware of, [18] is the first paper to advocate using the dual approach for solving QSDP problems even though the dual problem (**D**) looks a lot more complicated than the primal problem (**P**), especially with the presence of the subspace constraint involving $\mathcal{W}$. By leveraging on the Schur complement based decomposition technique developed in [17,18], Chen et al. [6] also employed the dual approach by proposing an efficient inexact ADMM-type first-order method (which we name as SCB-isPADMM) for solving problem (**D**). Promising numerical results have been obtained by the dual based first-order algorithms in solving various classes of QSDP problems to moder-

ate accuracy [6,18]. Naturally one may hope to also rely on the ADMM scheme to compute highly accurate solutions. However, as one will observe from the numerical experiments presented later in Sect. 6, ADMM-type methods are incapable of finding accurate solutions for difficult QSDP problems due to their slow local convergence or stagnation. On the other hand, recent studies on the convergence rate of the augmented Lagrangian method (ALM) for solving convex semidefinite programming with multiple solutions [7] show that comparing to ADMM-type methods, the ALM can enjoy a faster convergence rate (in fact asymptotically superlinear) under milder conditions. These recent advances thus strongly indicate that one should be able to design a highly efficient algorithm based on the ALM for (**D**) for solving QSDP problems to high accuracy. More specifically, we will propose a two-phase augmented Lagrangian based algorithm with Phase I to generate a reasonably good initial point to warm start the Phase II algorithm so as to compute accurate solutions efficiently. We call this new method QSDPNAL since it extends the ideas of SDPNAL [35] and SDPNAL+ [33] for linear SDP problems to QSDP problems. Although the aforementioned two-phase framework has already been demonstrated to be highly efficient for solving linear SDP problems [33,35], it remains to be seen whether we can achieve comparable or even more impressive performance on various QSDP problems.

In recent years, it has become fashionable to design first-order algorithms for solving convex optimization problems, with some even claiming their efficacy in solving various challenging classes of matrix conic optimization problems based on limited performance evaluations. However, based on our extensive numerical experience in solving large scale linear SDPs [28,33,35], we have observed that while first-order methods can be rather effective in solving easy problems which are well-posed and nondegenerate, they are typically powerless in solving difficult instances which are ill-posed or degenerate. Even for a well designed first-order algorithm with guaranteed convergence and highly optimized implementations, such as the ADMM+ algorithm in [28], a first-order method may still fail on slightly more challenging problems. For example, the ADMM+ algorithm designed in [33] can encounter varying degrees of difficulties in solving linear SDPs arising from rank-one tensor approximation problems. On the other hand, the SDPNAL algorithm in [35] (which exploits second-order information) is able to solve those problems very efficiently to high accuracy. We believe that in order to design an efficient and robust algorithm to solve the highly challenging class of matrix conic optimization problems including QSDPs, one must fully combine the advantages offered by both the first and second order algorithms, rather than just solely relying on first-order algorithms even though they may appear to be easier to implement.

Next we briefly describe our algorithm QSDPNAL. Let $\mathcal{Z} = \mathcal{S}^n \times \mathcal{W} \times \mathcal{S}^n \times \Re^m$. Consider the following Lagrange function associated with (**D**):

$$l(Z, W, S, y; X) := \delta^*_{\mathcal{K}}(-Z) + \frac{1}{2}\langle W,\ \mathcal{Q}W\rangle + \delta_{\mathcal{S}^n_+}(S) - \langle b,\ y\rangle \\ +\langle Z - \mathcal{Q}W + S + \mathcal{A}^* y - C,\ X\rangle,$$

where $(Z, W, S, y) \in \mathcal{Z}$ and $X \in \mathcal{S}^n$. For a given positive scalar σ, the augmented Lagrangian function for (**D**) is defined by

$$\mathcal{L}_\sigma(Z, W, S, y; X) := l(Z, W, S, y; X) + \frac{\sigma}{2}\|Z - \mathcal{Q}W + S + \mathcal{A}^* y - C\|^2, \quad (Z, W, S, y) \in \mathcal{Z},\ X \in \mathcal{S}^n. \tag{1}$$

The algorithm which we will adopt in QSDPNAL-Phase I is a variant of the SCB-isPADMM algorithm developed in [6]. In QSDPNAL-Phase II, we design an ALM for solving (**D**) where the inner subproblem in each iteration is solved via an inexact semismooth Newton based algorithm. Given $\sigma_0 > 0$, $(Z^0, W^0, S^0, y^0, X^0) \in \mathcal{Z} \times \mathcal{S}^n$, the $(k+1)$th iteration of the ALM consists of the following steps:

$$(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) \approx \operatorname{argmin}\left\{\mathcal{L}_{\sigma_k}(Z, W, S, y; X^k) \mid (Z, W, S, y) \in \mathcal{Z}\right\},$$
$$X^{k+1} = X^k + \sigma_k(Z^{k+1} - \mathcal{Q}W^{k+1} + S^{k+1} + \mathcal{A}^* y^{k+1} - C),$$

where $\sigma_k \in (0, +\infty)$. The first issue in the above ALM is the choice of the subspace $\mathcal{W}$. The obvious choice $\mathcal{W} = \mathcal{S}^n$ can lead to various difficulties in the implementation of the above algorithm. For example, since $\mathcal{Q} : \mathcal{S}^n \to \mathcal{S}^n$ is only assumed to be positive semidefinite, the Newton systems corresponding to the inner subproblems may be singular and the sequence $\{W^k\}$ generated by the ALM can be unbounded. As a result, it will be extremely difficult to analyze the convergence of the inner algorithm for solving the ALM subproblems. The second issue is that one needs to design easy-to-check stopping criteria for the inner subproblems, and to ensure the fast convergence of the ALM under reasonable conditions imposed on the QSDP problems. Concerning the first issue, we propose to choose $\mathcal{W} = \mathrm{Ran}(\mathcal{Q})$, although such a choice also leads to obstacles which we will overcome in Section 4. Indeed, by restricting $W \in \mathrm{Ran}(\mathcal{Q})$, the difficulties in analyzing the convergence and the superlinear (quadratic) convergence of the Newton-CG algorithm are circumvented as the possibilities of singularity and unboundedness are removed. For the second issue, under the restriction that $\mathcal{W} = \mathrm{Ran}(\mathcal{Q})$, thanks to the recent advances in [7], we are able to design checkable stopping criteria for solving the inner subproblems inexactly while establishing the global convergence of the above ALM. Moreover, we are able to establish the R-(super)linear convergence rate of the KKT residual. At the first glance, the restriction that $W \in \mathrm{Ran}(\mathcal{Q})$ appears to introduce severe numerical difficulties when we need to solve a linear system under this restriction. Fortunately, by carefully examining our algorithm and devising novel numerical techniques, we are able to overcome these difficulties as we shall see in Sect. 4. Our extensive evaluations of QSDPNAL have demonstrated that our algorithm is capable of solving large scale general QSDP problems of the form (**P**) to high accuracy very efficiently and robustly. For example, we are able to solve an elementwise weighted nearest correlation matrix estimation problem with matrix dimension $n = 10{,}000$ in less than 11 h to the relative accuracy of less than 10^{-6} in the KKT residual. Such a numerical performance has not been attained in the past.

As the readers may have already observed, even though our goal in developing algorithms for solving convex optimization problems such as (**P**) and (**D**) is to design those with desirable theoretical properties such as asymptotic superlinear convergence, it is our belief that it is equally if not even more important for the algorithms designed

to be practically implementable and able to achieve realistic numerical efficiency. It is obvious that our proposed two-phase augmented Lagrangian based algorithm for solving (**P**) and (**D**) is designed based on such a belief.

The remaining parts of this paper are organized as follows. The next section is devoted to our main algorithm QSDPNAL, which is a two-phase augmented Lagrangian based algorithm whose Phase I is used to generate a reasonably good initial point to warm-start the Phase II algorithm so as to obtain accurate solutions efficiently. In Sect. 3, we propose to solve the inner minimization subproblems of the ALM by semismooth Newton based algorithms and study their global and local superlinear (quadratic) convergence. In Sect. 4, we discuss critical numerical issues concerning the efficient implementation of QSDPNAL. In Sect. 5.1, we discuss the special case of applying QSDPNAL to solve least squares semidefinite programming problems. The extension of QSDPNAL for solving QSDP problems with unstructured inequality constraints is discussed in Sect. 5.2. In Sect. 6, we conduct numerical experiments to evaluate the performance of QSDPNAL in solving various QSDP problems and their extensions. We conclude our paper in the final section.

Below we list several notation and definitions to be used in the paper. For a given closed proper convex function $\theta : \mathcal{X} \to (-\infty, \infty]$, where $\mathcal{X}$ is a finite-dimensional real inner product space, its Fenchel conjugate function is denoted by $\theta^* : \mathcal{X} \to (-\infty, +\infty]$. For a given closed convex set $D \subseteq \mathcal{X}$ and $x \in \mathcal{X}$, we define by $\Pi_D(x)$ the metric projector of x onto D and $\mathrm{dist}(x, D) := \inf_{d\in D} \|x - d\| = \|x - \Pi_D(x)\|$. For any $X \in \mathcal{S}^n$, we use $\lambda_{\max}(X)$ and $\lambda_{\min}(X)$ to denote the largest and smallest eigenvalues of X, respectively. Similar notation is used when X is replaced by the linear operator $\mathcal{Q}$.

2 A two-phase augmented Lagrangian method

In this section, we shall present our two-phase algorithm QSDPNAL for solving the QSDP problems (**D**) and (**P**). For the convergence analysis of Algorithm QSDPNAL, we need to make the following standard assumption for (**P**). Such an assumption is analogous to the Slater's condition in the context of nonlinear programming in $\Re^m$.

Assumption 1 There exists $\widehat{X} \in \mathcal{S}^n_{++} \cap \mathrm{ri}(\mathcal{K})$ such that

$$\mathcal{A}(\mathcal{T}_{\mathcal{K}}(\widehat{X})) = \Re^m,$$

where $\mathrm{ri}(\mathcal{K})$ denotes the relative interior of $\mathcal{K}$ and $\mathcal{T}_{\mathcal{K}}(\widehat{X})$ is the tangent cone of $\mathcal{K}$ at point $\widehat{X}$.

2.1 Phase I: An SCB based inexact semi-proximal ADMM

In Phase I, we propose a new variant of the Schur complement based inexact semi-proximal ADMM (SCB-isPADMM) developed in [6] to solve (**D**). Recall the augmented Lagrangian function associated with problem (**D**) defined in (1).

The detailed steps of our Phase I algorithm for solving (**D**) are given as follows.

Algorithm QSDPNAL-Phase I: **An SCB based inexact semi-proximal ADMM for (D)**.
Select an initial point $(W^0, S^0, y^0, X^0) \in \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n_+ \times \Re^m \times \mathcal{S}^n$ and $-Z^0 \in \mathrm{dom}(\delta^*_{\mathcal{K}})$. Let $\{\varepsilon_k\}$ be a summable sequence of nonnegative numbers, and $\sigma > 0$, $\tau \in (0, \infty)$ are given parameters. For $k = 0, 1, \ldots$, perform the following steps in each iteration.

Step 1. Compute

$$\widehat{W}^k = \mathrm{argmin}\{\mathcal{L}_\sigma(Z^k, W, S^k, y^k; X^k) - \langle \hat{\delta}^k_{\mathcal{Q}}, W\rangle \mid W \in \mathrm{Ran}(\mathcal{Q})\}, \quad (2)$$

$$Z^{k+1} = \mathrm{argmin}\{\mathcal{L}_\sigma(Z, \widehat{W}^k, S^k, y^k; X^k) \mid Z \in \mathcal{S}^n\},$$

$$W^{k+1} = \mathrm{argmin}\{\mathcal{L}_\sigma(Z^{k+1}, W, S^k, y^k; X^k) - \langle \delta^k_{\mathcal{Q}}, W\rangle \mid W \in \mathrm{Ran}(\mathcal{Q})\}, \quad (3)$$

$$\begin{aligned}
\hat{y}^k &= \mathrm{argmin}\{\mathcal{L}_\sigma(Z^{k+1}, W^{k+1}, S^k, y; X^k) - \langle \hat{\delta}^k_y, y\rangle \mid y \in \Re^m\},\\
S^{k+1} &= \mathrm{argmin}\{\mathcal{L}_\sigma(Z^{k+1}, W^{k+1}, S, \hat{y}^k; X^k) \mid S \in \mathcal{S}^n\},\\
y^{k+1} &= \mathrm{argmin}\{\mathcal{L}_\sigma(Z^{k+1}, W^{k+1}, S^{k+1}, y; X^k) - \langle \delta^k_y, y\rangle \mid y \in \Re^m\},
\end{aligned}$$

where $\delta^k_y, \hat{\delta}^k_y \in \Re^m$, $\delta^k_{\mathcal{Q}}, \hat{\delta}^k_{\mathcal{Q}} \in \mathrm{Ran}(\mathcal{Q})$ are error vectors such that

$$\max\{\|\delta^k_y\|, \|\hat{\delta}^k_y\|, \|\delta^k_{\mathcal{Q}}\|, \|\hat{\delta}^k_{\mathcal{Q}}\|\} \le \varepsilon_k.$$

Step 2. Compute $X^{k+1} = X^k + \tau\sigma(Z^{k+1} - \mathcal{Q}W^{k+1} + S^{k+1} + \mathcal{A}^* y^{k+1} - C)$.

Remark 2.1 We shall explain here the role of the error vectors δ^k_y, $\hat{\delta}^k_y$, $\delta^k_{\mathcal{Q}}$ and $\hat{\delta}^k_{\mathcal{Q}}$. There is no need to choose these error vectors in advance. The presence of these error vectors simply indicates that the corresponding subproblems can be solved inexactly. For example, the updating rule of y^{k+1} in the above algorithm can be interpreted as follows: find y^{k+1} inexactly through

$$y^{k+1} \approx \mathrm{argmin}\, \mathcal{L}_\sigma(Z^{k+1}, W^{k+1}, S^{k+1}, y; X^k)$$

such that the residual

$$\|\delta^k_y\| = \|b - \mathcal{A}X^k - \sigma\mathcal{A}(Z^{k+1} - \mathcal{Q}W^{k+1} + S^{k+1} + \mathcal{A}^* y^{k+1} - C)\| \le \varepsilon_k.$$

Remark 2.2 In contrast to Aglorithm SCB-isPADMM in [6], our Algorithm QSDPNAL-Phase I requires the subspace constraint $W \in \mathrm{Ran}(\mathcal{Q})$ explicitly in the subproblems

(2) and (3). Note that due to the presence of the subspace constraint $W \in \mathrm{Ran}(\mathcal{Q})$, there is no need to add extra proximal terms in the subproblems corresponding to W to satisfy the positive definiteness requirement needed in applying the inexact Schur compliment based decomposition technique developed in [17,18]. This is certainly more elegant than the indirect reformulation strategy considered in [6,18].

The convergence of the above algorithm follows from [6, Theorem 1] without much difficulty, and its proof is omitted.

Theorem 2.1 *Suppose that the solution set of* (**P**) *is nonempty and Assumption 1 holds. Let* $\{(Z^k, W^k, S^k, y^k, X^k)\}$ *be the sequence generated by Algorithm* QSDPNAL-Phase I. *If* $\tau \in (0, (1+\sqrt{5})/2)$, *then the sequence* $\{(Z^k, W^k, S^k, y^k)\}$ *converges to an optimal solution of* (**D**) *and* $\{X^k\}$ *converges to an optimal solution of* (**P**).

Remark 2.3 Under some error bound conditions on the limit point of $\{(Z^k, W^k, S^k, y^k, X^k)\}$, one can derive the linear rate of convergence of the exact version of Algorithm QSDPNAL-Phase I. For a recent study on this topic, see [10] and the references therein. Here we will not address this issue as our Phase II algorithm enjoys a better rate of convergence under weaker conditions.

2.2 Phase II: An augmented Lagrangian algorithm

In this section, we discuss our Phase II algorithm for solving the dual problem (**D**). The purpose of this phase is to obtain high accuracy solutions efficiently after being warm-started by our Phase I algorithm. The Phase II of our algorithm has the following template.

Algorithm QSDPNAL-Phase II: **An augmented Lagrangian method of multipliers for solving (D).**
Let $\sigma_0 > 0$ be a given parameter. Choose $(W^0, S^0, y^0, X^0) \in \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n_+ \times \Re^m \times \mathcal{S}^n$ and $-Z^0 \in \mathrm{dom}(\delta^*_{\mathcal{K}})$. For $k = 0, 1, \ldots$, perform the following steps in each iteration.

Step 1. Compute

$$(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) \approx \mathrm{argmin} \left\{ \begin{array}{l} \Psi_k(Z, W, S, y) := \mathcal{L}_{\sigma_k}(Z, W, S, y; X^k) \\ \mid (Z, W, S, y) \in \mathcal{S}^n \times \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n \times \Re^m \end{array} \right\}. \tag{4}$$

Step 2. Compute

$$X^{k+1} = X^k + \sigma_k(Z^{k+1} - \mathcal{Q}W^{k+1} + S^{k+1} + \mathcal{A}^* y^{k+1} - C).$$

Update $\sigma_{k+1} \uparrow \sigma_\infty \leq \infty$.

As an important issue on the implementation of the above algorithm, the stopping criteria for approximately solving subproblem (4) shall be discussed here. Let the

feasible set for (**P**) be denoted as $\mathcal{F} := \{X \in \mathcal{S}^n \mid \mathcal{A}X = b,\ X \in \mathcal{S}^n_+ \cap \mathcal{K}\}$. Define the feasibility residual function $\gamma : \mathcal{S}^n \to \Re$ for the primal problem (**P**) by

$$\gamma(X) := \|b - \mathcal{A}X\| + \|X - \Pi_{\mathcal{S}^n_+}(X)\| + \|X - \Pi_{\mathcal{K}}(X)\|, \quad \forall\, X \in \mathcal{S}^n.$$

Note that $\gamma(X) = 0$ if and only if $X \in \mathcal{F}$. Indeed, for $X \notin \mathcal{F}$, $\gamma(X)$ provides an easy-to-compute measure on the primal infeasibility of X. Similar to [7, Proposition 4.2], we can use this feasibility residual function to derive an upper bound on the distance of a given point to the feasible set $\mathcal{F}$ in the next lemma. Its proof can be obtained without much difficulty by applying Hoffman's error bound [9, Lemma 3.2.3] to the nonempty polyhedral convex set $\{X \in \mathcal{S}^n \mid \mathcal{A}X = b,\ X \in \mathcal{K}\}$, e.g., see [2, Theorem 7].

Lemma 2.1 *Assume that $\mathcal{F} \cap \mathcal{S}^n_{++} \neq \emptyset$. Then, there exists a constant $\mu > 0$ such that*

$$\|X - \Pi_{\mathcal{F}}(X)\| \le \mu(1 + \|X\|)\gamma(X), \quad \forall\, X \in \mathcal{S}^n. \tag{5}$$

When the ALM is applied to solve (**D**), numerically it is difficult to execute the criteria (A'') and (B_1'') proposed in [26]. Fortunately, Lemma 2.1 and recent advances in the analysis of the ALM [7] allow us to design easy-to-verify stopping criteria for the subproblems in QSDPNAL-Phase II. For any $k \ge 0$, denote

$$f_k(X) := -\frac{1}{2}\langle X,\ \mathcal{Q}X\rangle - \langle C,\ X\rangle - \frac{1}{2\sigma_k}\|X - X^k\|^2, \quad \forall\, X \in \mathcal{S}^n.$$

Note that $f_k(\cdot)$ is in fact the objective function in the dual of problem (4). Let $\{\varepsilon_k\}$ and $\{\delta_k\}$ be two given positive summable sequences. Given $k \ge 0$ and $X^k \in \mathcal{S}^n$, we propose to terminate the minimization of the subproblem (4) in the $(k+1)$th iteration of Algorithm QSDPNAL-Phase II with either one of the following two easy-to-check stopping criteria:

$$\text{(A)} \quad \begin{cases} \Psi_k(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) - f_k(X^{k+1}) \le \varepsilon_k^2/2\sigma_k, \\ (1 + \|X^{k+1}\|)\gamma(X^{k+1}) \le \alpha_k\varepsilon_k/\sqrt{2\sigma_k}, \end{cases}$$

$$\text{(B)} \quad \begin{cases} \Psi_k(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) - f_k(X^{k+1}) \le \delta_k^2\|X^{k+1} - X^k\|^2/2\sigma_k, \\ (1 + \|X^{k+1}\|)\gamma(X^{k+1}) \le \beta_k\delta_k\|X^{k+1} - X^k\|/\sqrt{2\sigma_k}, \end{cases}$$

where

$$\alpha_k = \min\left\{1, \sqrt{\sigma_k}, \frac{\varepsilon_k}{\sqrt{2\sigma_k}\|\nabla f_k(X^{k+1})\|}\right\} \quad \text{and}$$

$$\beta_k = \min\left\{1, \sqrt{\sigma_k}, \frac{\delta_k\|X^{k+1} - X^k\|}{\sqrt{2\sigma_k}\|\nabla f_k(X^{k+1})\|}\right\}.$$

Lemma 2.2 *Assume that Assumption 1 holds. Let μ be the constant given in (5). Suppose that for some $k \geq 0$, X^k is not an optimal solution to problem (**P**). Then one can always find $(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1})$ and $X^{k+1} = X^k + \sigma_k(Z^{k+1} - \mathcal{Q}W^{k+1} + S^{k+1} + \mathcal{A}^* y^{k+1} - C)$ satisfying both (A) and (B). Moreover, (A) implies that*

$$\Psi_k(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) - \inf \Psi_k \leq \nu \varepsilon_k^2 / 2\sigma_k$$

and (B) implies that

$$\Psi_k(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) - \inf \Psi_k \leq (\nu \delta_k^2 / 2\sigma_k) \|X^{k+1} - X^k\|^2,$$

respectively, where

$$\nu = 1 + \mu + \frac{1}{2}\lambda_{\max}(\mathcal{Q}) + \frac{1}{2}\mu^2. \tag{6}$$

Proof With the help of Lemma 2.1, one can establish the assertion in the same fashion as in [7, Proposition 4.2, Proposition 4.3]. □

For the subsequent analysis, we need to define the essential objective function of (**P**), which is given by

$$\begin{aligned} \phi(X) :=& -\inf\{l(Z, W, S, y; X) \mid (Z, W, S, y) \in \mathcal{S}^n \times \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n \times \Re^m\} \\ =& \begin{cases} \frac{1}{2}\langle X, \mathcal{Q}X\rangle + \langle X, C\rangle + \delta_{\mathcal{S}^n_+}(X) + \delta_{\mathcal{K}}(X) & \text{if } \mathcal{A}X = b, \\ +\infty & \text{otherwise.} \end{cases} \end{aligned}$$

For convenience, we also let $\Omega = \partial\phi^{-1}(0)$ denote the solution set of (**P**).

We say that for (**P**), the second order growth condition holds at an optimal solution $\overline{X} \in \Omega$ with respect to the set Ω if there exist $\kappa > 0$ and a neighborhood U of $\overline{X}$ such that

$$\phi(X) \geq \phi(\overline{X}) + \kappa^{-1}\mathrm{dist}^2(X, \Omega), \quad \forall\, X \in U. \tag{7}$$

Let the objective function $g : \mathcal{S}^n \times \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n \times \Re^m \to (-\infty, +\infty]$ associated with (**D**) be given as follows:

$$\begin{aligned} g(Z, W, S, y) := \delta^*_{\mathcal{K}}(-Z) + \frac{1}{2}\langle W, \mathcal{Q}W\rangle + \delta_{\mathcal{S}^n_+}(S) - \langle b, y\rangle, \\ \forall\, (Z, W, S, y) \in \mathcal{S}^n \times \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n \times \Re^m. \end{aligned}$$

Now, with Lemma 2.2, we can prove the global and local (super)linear convergence of Algorithm QSDPNAL-Phase II by adapting the proofs in [26, Theorem 4] and [7, Theorem 4.2]. It shows that, for most QSDP problems, one can always expect the KKT residual of the sequence generated by QSDPNAL-Phase II to converge at least R-(super)linearly.

Theorem 2.2 *Suppose that Ω, the solution set of* (**P**)*, is nonempty and Assumption 1 holds. Then the sequence $\{(Z^k, W^k, S^k, y^k, X^k)\}$ generated by Algorithm* QSDPNAL-Phase II *under the stopping criterion* (A) *for all $k \geq 0$ is bounded, and $\{X^k\}$ converges to an optimal solution X^∞ of* (**P**)*, and $\{(Z^k, W^k, S^k, y^k)\}$ converges to an optimal solution of* (**D**)*. Moreover, for all $k \geq 0$, it holds that*

$$\begin{aligned} &g(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) - \inf(\mathbf{D}) \\ &\quad \leq \Psi_k(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) - \inf \Psi_k + (1/2\sigma_k)(\|X^k\|^2 - \|X^{k+1}\|^2). \end{aligned}$$

Assume that for (**P**)*, the second order growth condition (7) holds at X^∞ with respect to the set Ω, i.e., there exists a constant $\kappa > 0$ and a neighborhood U of X^∞ such that*

$$\phi(X) \geq \phi(X^\infty) + \kappa^{-1}\mathrm{dist}^2(X, \Omega), \quad \forall\, X \in U.$$

Suppose that the algorithm is executed under criteria (A) *and* (B) *for all $k \geq 0$ and ν is the constant given in (6). Then, for all k sufficiently large, it holds that*

$$\mathrm{dist}(X^{k+1}, \Omega) \leq \theta_k \mathrm{dist}(X^k, \Omega), \tag{8}$$

$$\|Z^{k+1} - \mathcal{Q}W^{k+1} + S^{k+1} + \mathcal{A}^* y - C\| \leq \tau_k \mathrm{dist}(X^k, \Omega), \tag{9}$$

$$g(Z^{k+1}, W^{k+1}, S^{k+1}, y^{k+1}) - \inf(\mathbf{D}) \leq \tau'_k \mathrm{dist}(X^k, \Omega), \tag{10}$$

where

$$\begin{aligned} &1 > \theta_k = \big(\kappa/\sqrt{\kappa^2 + \sigma_k^2} + 2\nu\delta_k\big)(1 - \nu\delta_k)^{-1} \to \theta_\infty = \kappa/\sqrt{\kappa^2 + \sigma_\infty^2} \\ &\quad (\theta_\infty = 0 \text{ if } \sigma_\infty = \infty), \\ &\tau_k = \sigma_k^{-1}(1 - \nu\delta_k)^{-1} \to \tau_\infty = 1/\sigma_\infty \quad (\tau_\infty = 0 \text{ if } \sigma_\infty = \infty), \\ &\tau'_k = \tau_k(\nu^2\delta_k^2\|X^{k+1} - X^k\| + \|X^{k+1}\| + \|X^k\|)/2 \to \tau'_\infty = \|X^\infty\|/\sigma_\infty \\ &\quad (\tau'_\infty = 0 \text{ if } \sigma_\infty = \infty). \end{aligned}$$

Next we give a few comments on the convergence rates and assumptions made in Theorem 2.2.

Remark 2.4 Under the assumptions of Theorem 2.2, we have proven that the KKT residual, corresponding to (**P**) and (**D**), along the sequence $\{(Z^k, W^k, S^k, y^k, X^k)\}$ converges at least R-(super)linearly. Indeed, under stopping criteria (A), (B) and from (8),(9) and (10), we know that the primal feasibility, the dual feasibility and the duality gap all converge at least R-(super)linearly.

Remark 2.5 The assumption that the second order growth condition (7) holds for (**P**) is quite mild. Indeed, it holds when any optimal solution $\overline{X}$ of (**P**), together with any of its multiplier $\overline{S} \in \mathcal{S}_+^n$ corresponding only to the semidefinite constraint, satisfies the strict complementarity condition [7, Corollary 3.1]. It is also valid when the "no-gap"

second order sufficient condition holds at the optimal solution[1] to (**P**) [4, Theorem 3.137].

3 Inexact semismooth Newton based algorithms for solving the inner subproblems (4) in ALM

In this section, we will design efficient inexact semismooth Newton based algorithms to solve the inner subproblems (4) in the augmented Lagrangian method, where each subproblem takes the form of:

$$\begin{aligned}\min\{&\Psi(Z,W,S,y)\\ &:=\mathcal{L}_\sigma(Z,W,S,y;\widehat{X})\mid (Z,W,S,y)\in\mathcal{S}^n\times\mathrm{Ran}(\mathcal{Q})\times\mathcal{S}^n\times\Re^m\}\end{aligned}\tag{11}$$

for a given $\widehat{X}\in\mathcal{S}^n$. Note that the dual problem of (11) is given as follows:

$$\max\left\{-\frac{1}{2}\langle X,\ \mathcal{Q}X\rangle-\langle C,\ X\rangle-\frac{1}{2\sigma}\|X-\widehat{X}\|^2\mid \mathcal{A}X=b,\ X\in\mathcal{S}^n_+,\ X\in\mathcal{K}\right\}.$$

Under Assumption 1, from [24, Theorems 17 & 18], we know that the optimal solution set of problem (11) is nonempty and for any $\alpha\in\Re$, the level set $\mathcal{L}_\alpha:=\{(Z,W,S,y)\in\mathcal{S}^n\times\mathrm{Ran}(\mathcal{Q})\times\mathcal{S}^n\times\Re^m\mid\Psi(Z,W,S,y)\le\alpha\}$ is a closed and bounded convex set.

3.1 A semismooth Newton-CG algorithm for (11) with $\mathcal{K}=\mathcal{S}^n$

Note that in quite a number of applications, the polyhedral convex set $\mathcal{K}$ is actually the whole space $\mathcal{S}^n$. Therefore, we shall first study how the inner problems (11) in Algorithm ALM can be solved efficiently when $\mathcal{K}=\mathcal{S}^n$. Under this setting, Z is vacuous, i.e., $Z=0$.

Let $\sigma>0$ be given. Denote

$$\Upsilon(W,y):=\mathcal{A}^*y-\mathcal{Q}W-\widehat{C},\quad\forall\,(W,y)\in\mathrm{Ran}(\mathcal{Q})\times\Re^m.$$

where $\widehat{C}=C-\sigma^{-1}\widehat{X}$. Observe that if

$$(W^*,S^*,y^*)=\mathrm{argmin}\{\Psi(0,W,S,y)\mid(W,S,y)\in\mathrm{Ran}(\mathcal{Q})\times\mathcal{S}^n\times\Re^m\},$$

then (W^*,S^*,y^*) can be computed in the following manner

$$\begin{aligned}&(W^*,y^*)=\mathrm{argmin}\left\{\varphi(W,y)\mid(W,y)\in\mathrm{Ran}(\mathcal{Q})\times\Re^m\right\},\\ &S^*=\Pi_{\mathcal{S}^n_+}(-\Upsilon(W^*,y^*)),\end{aligned}\tag{12}$$

[1] In this case, the optimal solution set to (**P**) is necessarily a singleton though (**D**) may have multiple solutions.

where

$$\varphi(W, y) := \frac{1}{2}\langle W,\ \mathcal{Q}W\rangle - \langle b,\ y\rangle + \frac{\sigma}{2}\|\Pi_{\mathcal{S}^n_+}(\Upsilon(W, y))\|^2, \forall\, (W, y) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m.$$

Note that $\varphi(\cdot, \cdot)$ is a continuously differentiable function on $\mathrm{Ran}(\mathcal{Q}) \times \Re^m$ with

$$\nabla\varphi(W, y) = \begin{pmatrix} \mathcal{Q}W - \sigma\,\mathcal{Q}\Pi_{\mathcal{S}^n_+}(\Upsilon(W, y)) \\ -b + \sigma\mathcal{A}\Pi_{\mathcal{S}^n_+}(\Upsilon(W, y)) \end{pmatrix}.$$

Then, solving (12) is equivalent to solving the following nonsmooth equation:

$$\nabla\varphi(W, y) = 0, \quad (W, y) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m.$$

Since $\Pi_{\mathcal{S}^n_+}$ is strongly semismooth [27], we can design a semismooth Newton-CG (SNCG) method to solve (12) and could expect to get a fast superlinear or even quadratic convergence. For any $(W, y) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m$, define

$$\hat{\partial}^2\varphi(W, y) := \begin{bmatrix} \mathcal{Q} & \\ & 0 \end{bmatrix} + \sigma \begin{bmatrix} \mathcal{Q} \\ -\mathcal{A} \end{bmatrix} \partial\Pi_{\mathcal{S}^n_+}(\Upsilon(W, y))[\mathcal{Q}\ -\mathcal{A}^*],$$

where $\partial\Pi_{\mathcal{S}^n_+}(\Upsilon(W, y))$ is the Clarke subdifferential [8] of $\Pi_{\mathcal{S}^n_+}(\cdot)$ at $\Upsilon(W, y)$. Note that from [12], we know that

$$\hat{\partial}^2\varphi(W, y)\,(d_W, d_y) = \partial^2\varphi(W, y)\,(d_W, d_y), \quad \forall\,(d_W, d_y) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m,$$

where $\partial^2\varphi(W, y)$ denotes the generalized Hessian of φ at (W, y), i.e., the Clarke subdifferential of $\nabla\varphi$ at (W, y).

Given $(\widetilde{W}, \tilde{y}) \in \mathrm{Ran}(Q) \times \Re^m$, consider the following eigenvalue decomposition:

$$\Upsilon(\widetilde{W}, \tilde{y}) = \mathcal{A}^*\tilde{y} - \mathcal{Q}\widetilde{W} - \widehat{C} = P\,\Gamma\,P^{\mathrm{T}},$$

where $P \in \Re^{n\times n}$ is an orthogonal matrix whose columns are eigenvectors, and Γ is the corresponding diagonal matrix of eigenvalues, arranged in a nonincreasing order: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Define the following index sets

$$\alpha := \{i \mid \lambda_i > 0\}, \quad \bar{\alpha} := \{i \mid \lambda_i \leq 0\}.$$

We define the operator $U^0 : \mathcal{S}^n \to \mathcal{S}^n$ by

$$U^0(H) := P(\Sigma \circ (P^{\mathrm{T}}HP))P^{\mathrm{T}}, \quad H \in \mathcal{S}^n, \tag{13}$$

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

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

and $E_{\alpha\alpha} \in \mathcal{S}^{|\alpha|}$ is the matrix of ones. In [20, Lemma 11], it is proved that

$$U^0 \in \partial \Pi_{\mathcal{S}^n_+}(\Upsilon(\widetilde{W}, \tilde{y})).$$

Define

$$V^0 := \begin{bmatrix} \mathcal{Q} & \\ & 0 \end{bmatrix} + \sigma \begin{bmatrix} \mathcal{Q} \\ -\mathcal{A} \end{bmatrix} U^0[\mathcal{Q} - \mathcal{A}^*]. \tag{14}$$

Then, we have $V^0 \in \hat{\partial}^2 \varphi(\widetilde{W}, \tilde{y})$.

After all the above preparations, we can design the following semismooth Newton-CG method as in [35] to solve (12).

Algorithm SNCG: **A semismooth Newton-CG algorithm.**
Given $\mu \in (0, 1/2)$, $\bar{\eta} \in (0, 1)$, $\tau \in (0, 1]$, $\tau_1, \tau_2 \in (0, 1)$ and $\delta \in (0, 1)$. Choose $(W^0, y^0) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m$. Set $j = 0$. Iterate the following steps.

Step 1. Choose $U^0 \in \partial \Pi_{\mathcal{S}^n_+}(\Upsilon(W^j, y^j))$ defined as in (13). Let $V_j := V^0$ be given as in (14) and $\epsilon_j = \tau_1 \min\{\tau_2, \|\nabla \varphi(W^j, y^j)\|\}$. Apply the CG algorithm to find an approximate solution $(d_W^j, d_y^j) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m$ to

$$V_j(d_W, d_y) + \epsilon_j(0, d_y) = -\nabla \varphi(W^j, y^j) \tag{15}$$

such that

$$\|V_j(d_W^j, d_y^j) + \epsilon_j(0, d_y^j) + \nabla \varphi(W^j, y^j)\| \le \eta_j := \min(\bar{\eta}, \|\nabla \varphi(W^j, y^j)\|^{1+\tau}).$$

Step 2. Set $\alpha_j = \delta^{m_j}$, where m_j is the first nonnegative integer m for which

$$\varphi(W^j + \delta^m d_W^j, y^j + \delta^m d_y^j) \le \varphi(W^j, y^j) + \mu \delta^m \langle \nabla \varphi(W^j, y^j), (d_W^j, d_y^j) \rangle.$$

Step 3. Set $W^{j+1} = W^j + \alpha_j d_W^j$ and $y^{j+1} = y^j + \alpha_j d_y^j$.

The convergence results for the above SNCG algorithm are stated in the next theorem.

Theorem 3.1 *Suppose that Assumption 1 holds. Then Algorithm SNCG generates a bounded sequence* $\{(W^j, y^j)\}$ *and any accumulation point* $(\overline{W}, \bar{y}) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m$ *is an optimal solution to problem* (12).

The following proposition is the key ingredient in our subsequent convergence analysis.

Proposition 3.1 *Let* $\mathcal{U} : \mathcal{S}^n \to \mathcal{S}^n$ *be a self-adjoint positive semidefinite linear operator and* $\sigma > 0$. *Then, it holds that* $\mathcal{A}\mathcal{U}\mathcal{A}^*$ *is positive definite if and only if*

$$\left\langle \begin{bmatrix} W \\ y \end{bmatrix}, \left(\begin{bmatrix} \mathcal{Q} & \\ & 0 \end{bmatrix} + \sigma \begin{bmatrix} \mathcal{Q} \\ -\mathcal{A} \end{bmatrix} \mathcal{U}[\mathcal{Q} - \mathcal{A}^*] \right) \begin{bmatrix} W \\ y \end{bmatrix} \right\rangle > 0 \tag{16}$$

for all $(W, y) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m \backslash \{(0,0)\}$.

Proof Since the "if" statement obviously holds true, we only need to prove the "only if" statement. Note that

$$\langle W,\ \mathcal{Q}W \rangle > 0, \quad \forall\, W \in \mathrm{Ran}(\mathcal{Q}) \backslash \{0\}.$$

Now suppose that $\mathcal{A}\mathcal{U}\mathcal{A}^*$ is positive definite, and hence nonsingular. By the Schur complement condition for ensuring the positive definiteness of a linear operator, we know that (16) holds if and only if

$$\langle W,\ (\mathcal{Q} + \sigma\, \mathcal{Q}\mathcal{U}\mathcal{Q} - \sigma\, \mathcal{Q}\mathcal{U}\mathcal{A}^*(\mathcal{A}\mathcal{U}\mathcal{A}^*)^{-1}\mathcal{A}\mathcal{U}\mathcal{Q})W \rangle > 0, \quad \forall\, W \in \mathrm{Ran}(\mathcal{Q}) \backslash \{0\}. \tag{17}$$

But for any $W \in \mathrm{Ran}(\mathcal{Q})\backslash\{0\}$, we have that $\langle W,\ \mathcal{Q}W \rangle > 0$, and

$$\begin{aligned} &\langle W,\ (\mathcal{Q}\mathcal{U}\mathcal{Q} - \mathcal{Q}\mathcal{U}\mathcal{A}^*(\mathcal{A}\mathcal{U}\mathcal{A}^*)^{-1}\mathcal{A}\mathcal{U}\mathcal{Q})W \rangle \\ &= \langle W,\ \mathcal{Q}\mathcal{U}^{\frac{1}{2}}(\mathcal{I} - \mathcal{U}^{\frac{1}{2}}\mathcal{A}^*(\mathcal{A}\mathcal{U}\mathcal{A}^*)^{-1}\mathcal{A}\mathcal{U}^{\frac{1}{2}})\mathcal{U}^{\frac{1}{2}}\mathcal{Q}W \rangle \\ &= \langle \mathcal{U}^{\frac{1}{2}}\mathcal{Q}W,\ (\mathcal{I} - \mathcal{U}^{\frac{1}{2}}\mathcal{A}^*(\mathcal{A}\mathcal{U}\mathcal{A}^*)^{-1}\mathcal{A}\mathcal{U}^{\frac{1}{2}})\mathcal{U}^{\frac{1}{2}}\mathcal{Q}W \rangle \ \geq\ 0. \end{aligned}$$

Hence, (17) holds automatically. This completes the proof of the proposition. □

Base on the above proposition, under the constraint nondegeneracy condition for (**P**), we shall show in the next theorem that one can still ensure the positive definiteness of the coefficient matrix in the semismooth Newton system at the solution point.

Theorem 3.2 *Let* $(\overline{W}, \bar{y})$ *be the optimal solution for problem* (12). *Let* $\overline{Y} := \Pi_{\mathcal{S}^n_+}(\mathcal{A}^*\bar{y} - \mathcal{Q}\overline{W} - \widehat{C})$. *The following conditions are equivalent:*

(i) *The constraint nondegeneracy condition,*

$$\mathcal{A}\,\mathrm{lin}(\mathcal{T}_{\mathcal{S}^n_+}(\overline{Y})) = \Re^m, \tag{18}$$

holds at $\overline{Y}$, *where* $\mathrm{lin}(\mathcal{T}_{\mathcal{S}^n_+}(\overline{Y}))$ *denotes the lineality space of the tangent cone of* $\mathcal{S}^n_+$ *at* $\overline{Y}$.

(ii) *Every element in*

$$\begin{bmatrix} \mathcal{Q} & \\ & 0 \end{bmatrix} + \sigma \begin{bmatrix} \mathcal{Q} \\ -\mathcal{A} \end{bmatrix} \partial \Pi_{\mathcal{S}^n_+}(\mathcal{A}^*\bar{y} - \mathcal{Q}\overline{W} - \widehat{C})[\mathcal{Q} - \mathcal{A}^*]$$

is self-adjoint and positive definite on $\mathrm{Ran}(\mathcal{Q}) \times \Re^m$.

Proof In the same fashion as in [35, Proposition 3.2], we can prove that $\mathcal{A}\mathcal{U}\mathcal{A}^*$ is positive definite for all $\mathcal{U} \in \partial\Pi_{\mathcal{S}^n_+}(\mathcal{A}^*\bar{y} - \mathcal{Q}\overline{W} - \widehat{C})$ if only if (i) holds. Then, by Proposition 3.1, we readily obtain the desired results. □

Theorem 3.3 *Assume that Assumption 1 holds. Let* $(\overline{W}, \bar{y})$ *be an accumulation point of the infinite sequence* $\{(W^j, y^j)\}$ *generated by Algorithm SNCG for solving problem* (12). *Assume that the constraint nondegeneracy condition* (18) *holds at* $\overline{Y} := \Pi_{\mathcal{S}^n_+}(\mathcal{A}^*\bar{y} - \mathcal{Q}\overline{W} - \widehat{C})$. *Then, the whole sequence* $\{(W^j, y^j)\}$ *converges to* $(\overline{W}, \bar{y})$ *and*

$$\|(W^{j+1}, y^{j+1}) - (\overline{W}, \bar{y})\| = O(\|(W^j, y^j) - (\overline{W}, \bar{y})\|^{1+\tau}).$$

Proof From Theorem 3.2, we know that under the constraint nondegeneracy condition (18), every $V \in \hat{\partial}^2\varphi(\overline{W}, \bar{y})$ is self-adjoint and positive definite on $\mathrm{Ran}(\mathcal{Q}) \times \Re^n$. Hence one can obtain the desired results from [35, Theorem 3.5] by further noting the strong semismoothness of $\Pi_{\mathcal{S}^n_+}(\cdot)$. □

We note that the convergence results obtained in this subsection depend critically on the restriction that $W \in \mathcal{W} = \mathrm{Ran}(\mathcal{Q})$. Without this restriction, the possible singularity of the Newton systems (15) and the unboundedness of $\{W^j\}$ will make the convergence analysis highly challenging, if possible at all.

3.2 Semismooth Newton based inexact ABCD algorithms for (11) when $\mathcal{K} \neq \mathcal{S}^n$

When $\mathcal{K} \neq \mathcal{S}^n$, we will adapt the recently developed inexact accelerated block coordinate descent (ABCD) algorithm [29] to solve the inner subproblems (11) in the augmented Lagrangian method.

The detailed steps of the ABCD algorithm to be used for solving (11) will be presented below. In this algorithm, (Z, W, S, y) is decomposed into two groups, namely Z and (W, S, y). In this case, (W, S, y) is regarded as a single block and the corresponding subproblem in the ABCD algorithm can only be solved by an iterative method inexactly. Here, we propose to develop a semismooth Newton-CG method to solve the corresponding subproblem.

Algorithm ABCD$(Z^0, W^0, S^0, y^0, \widehat{X}, \sigma)$: **An inexact ABCD algorithm for (11).** Given $(W^0, S^0, y^0) \in \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n_+ \times \Re^m$, $-Z^0 \in \mathrm{dom}(\delta^*_{\mathcal{K}})$ and $\eta > 0$, set $(\widetilde{Z}^1, \widetilde{W}^1, \widetilde{S}^1, \tilde{y}^1) = (Z^0, W^0, S^0, y^0)$ and $t_1 = 1$. Let $\{\varepsilon_l\}$ be a nonnegative summable sequence. For $l = 1, \ldots,$ perform the following steps in each iteration.

Step 1. Let $\widetilde{R}^l = \sigma(\widetilde{S}^k + \mathcal{A}^*\tilde{y}^l - \mathcal{Q}\widetilde{W}^l - C + \sigma^{-1}\widehat{X})$. Compute

$$Z^l = \mathrm{argmin}\left\{\Psi(Z, \widetilde{W}^l, \widetilde{S}^l, \tilde{y}^l) \mid Z \in \mathcal{S}^n\right\} = \frac{1}{\sigma}\big(\Pi_{\mathcal{K}}(\widetilde{R}^l) - \widetilde{R}^l\big),$$

$$(W^l, S^l, y^l) = \mathrm{argmin}\left\{\begin{array}{l}\Psi(Z^l, W, S, y) + \dfrac{\eta}{2}\|y - \tilde{y}^l\|^2 - \langle\delta^l_y,\ y\rangle - \langle\delta^l_{\mathcal{Q}},\ W\rangle \\ \mid (W, S, y) \in \mathrm{Ran}(\mathcal{Q}) \times \mathcal{S}^n \times \Re^m\end{array}\right\}, \tag{19}$$

where $\delta^l_y \in \Re^m$, $\delta^l_{\mathcal{Q}} \in \mathrm{Ran}(\mathcal{Q})$ are error vectors such that

$$\max\{\|\delta^l_y\|, \|\delta^l_{\mathcal{Q}}\|\} \le \varepsilon_l / t_l.$$

Step 2. Set $t_{l+1} = \frac{1+\sqrt{1+4t_l^2}}{2}$, $\beta_l = \frac{t_l - 1}{t_{l+1}}$. Compute

$$\widetilde{W}^{l+1} = W^l + \beta_l(W^l - W^{l-1}),\ \widetilde{S}^{l+1} = S^l + \beta_l(S^l - S^{l-1}),\ \tilde{y}^{l+1} = y^l + \beta_l(y^l - y^{l-1}).$$

Note that in order to meet the convergence requirement of the inexact ABCD algorithm, a proximal term involving the positive parameter η is added in (19) to ensure the strong convexity of the objective function in the subproblem. For computational efficiency, one can always take η to be a small number, say 10^{-6}. For the subproblem (19), it can be solved by a semismooth Newton-CG algorithm similar to the one developed in Sect. 3.1. Since $\eta > 0$, the superlinear convergence of such a semismooth Newton-CG algorithm can also be proven based on the strong semismoothness of $\Pi_{\mathcal{S}^n_+}(\cdot)$ and the symmetric positive definiteness of the corresponding generalized Hessian.

The convergence results for the above Algorithm ABCD are stated in the next theorem, whose proof essentially follows from that in [29, Theorem 3.1]. Here, we omit the proof for brevity.

Theorem 3.4 *Suppose that Assumption 1 holds and $\eta > 0$. Let $\{(Z^l, W^l, S^l, y^l)\}$ be the sequence generated by Algorithm ABCD. Then,*

$$\inf_Z \Psi(Z, W^l, S^l, y^l) - \Psi(Z^*, W^*, S^*, y^*) = O(1/l^2)$$

where (Z^*, W^*, S^*, y^*) is an optimal solution of problem (11). Moreover, the sequence $\{(Z^l, W^l, S^l, y^l)\}$ is bounded and all of its cluster points are optimal solutions to problem (11).

4 Numerical issues in QSDPNAL

In Algorithm QSDPNAL-Phase I, in order to obtain $\widehat{W}^k$ and W^{k+1} at the kth iteration, we need to solve the following linear system of equations

$$(\mathcal{Q} + \sigma \mathcal{Q}^2)W \approx \mathcal{Q}R, \quad W \in \mathrm{Ran}(\mathcal{Q}) \tag{20}$$

with the residual

$$\|\mathcal{Q}R - (\mathcal{Q} + \sigma \mathcal{Q}^2)W\| \le \varepsilon, \tag{21}$$

where $R \in \mathcal{S}^n$ and $\varepsilon > 0$ are given. Note that the exact solution to (20) is unique since $\mathcal{Q} + \sigma \mathcal{Q}^2$ is positive definite on $\mathrm{Ran}(\mathcal{Q})$. But the linear system is typically very large even for a moderate n, say $n = 500$. Under the high dimensional setting which we are particularly interested in, the matrix representation of $\mathcal{Q}$ is generally not available or too expensive to be stored explicitly. Thus (20) can only be solved inexactly by an iterative method. However when $\mathcal{Q}$ is singular (and hence $\mathrm{Ran}(\mathcal{Q}) \neq \mathcal{S}^n$), due to the presence of the subspace constraint $W \in \mathrm{Ran}(\mathcal{Q})$, it is extremely difficult to apply preconditioning to (20) while ensuring that the approximate solution is contained in $\mathrm{Ran}(\mathcal{Q})$. Fortunately, as shown in the next proposition, instead of solving (20) directly, we can solve a simpler and yet better conditioned linear system to overcome this difficulty.

Proposition 4.1 *Let $\widehat{W}$ be an approximate solution to the following linear system:*

$$(\mathcal{I} + \sigma \mathcal{Q})W \approx R \tag{22}$$

with the residual satisfying

$$\|R - (\mathcal{I} + \sigma \mathcal{Q})\widehat{W}\| \le \frac{\varepsilon}{\lambda_{\max}(\mathcal{Q})}.$$

Then, $\widehat{W}_{\mathcal{Q}} := \Pi_{\mathrm{Ran}(\mathcal{Q})}(\widehat{W}) \in \mathrm{Ran}(\mathcal{Q})$ solves (20) with the residual satisfying (21). Moreover, $\mathcal{Q}\widehat{W}_{\mathcal{Q}} = \mathcal{Q}\widehat{W}$ and $\langle \widehat{W}_{\mathcal{Q}}, \mathcal{Q}\widehat{W}_{\mathcal{Q}} \rangle = \langle \widehat{W}, \mathcal{Q}\widehat{W} \rangle$.

Proof First we note that the results $\mathcal{Q}\widehat{W}_{\mathcal{Q}} = \mathcal{Q}\widehat{W}$ and $\langle \widehat{W}_{\mathcal{Q}}, \mathcal{Q}\widehat{W}_{\mathcal{Q}} \rangle = \langle \widehat{W}, \mathcal{Q}\widehat{W} \rangle$ follow from the decomposition $\widehat{W} = \Pi_{\mathrm{Ran}(\mathcal{Q})}(\widehat{W}) + \Pi_{\mathrm{Ran}(\mathcal{Q})^\perp}(\widehat{W})$. Next, by observing that

$$\|\mathcal{Q}R - (\mathcal{Q} + \sigma \mathcal{Q}^2)\widehat{W}_{\mathcal{Q}}\| = \|\mathcal{Q}R - (\mathcal{Q} + \sigma \mathcal{Q}^2)\widehat{W}\| \le \lambda_{\max}(\mathcal{Q})\, \|R - (\mathcal{I} + \sigma \mathcal{Q})\widehat{W}\| \le \varepsilon,$$

one can easily obtain the desired results. □

By Proposition 4.1, in order to obtain $\widehat{W}_{\mathcal{Q}}$, we can first apply an iterative method such as the preconditioned conjugate gradient (PCG) method to solve (22) to obtain $\widehat{W}$ and then perform the projection step. However, by carefully analysing the steps in QSDPNAL-Phase I, we are surprised to observe that instead of explicitly computing

$\widehat{W}_{\mathcal{Q}}$, we can update the iterates in the algorithm by using only $\mathcal{Q}\widehat{W}_{\mathcal{Q}} = \mathcal{Q}\widehat{W}$. Thus, we only need to compute $\mathcal{Q}\widehat{W}$ and the potentially expensive projection step to compute $\widehat{W}_{\mathcal{Q}}$ can be avoided completely.

It is important for us to emphasize the computational advantage of solving the linear system (22) over (20). First, the former only requires one evaluation of $\mathcal{Q}(\cdot)$ whereas the latter requires two such evaluations in each PCG iteration. Second, the coefficient matrix in the former system is typically much more well-conditioned than the coefficient matrix in the latter system. More precisely, when $\mathcal{Q}$ is positive definite, then $\mathcal{I} + \sigma\mathcal{Q}$ is clearly better conditioned than $\mathcal{Q} + \sigma\mathcal{Q}^2$ by a factor of $\lambda_{\max}(\mathcal{Q})/\lambda_{\min}(\mathcal{Q})$. When $\mathcal{Q}$ is singular, with its smallest positive eigenvalue denoted as $\lambda_+(\mathcal{Q})$, then $\mathcal{I} + \sigma\mathcal{Q}$ is better conditioned when $\lambda_{\max}(\mathcal{Q}) \geq \lambda_+(\mathcal{Q})(1 + \sigma\lambda_+(\mathcal{Q}))$. The previous inequality would obviously hold when $\lambda_+ \leq (\sqrt{4\sigma\lambda_{\max}(\mathcal{Q}) + 1} - 1)/(2\sigma)$.

In Algorithm QSDPAL-Phase II, the subspace constraint $W \in \mathrm{Ran}(\mathcal{Q})$ also appears when we solve the semismooth Newton linear system (15) in Algorithm SNCG. Specifically, we need to find (dW, dy) to solve the following linear system

$$V\,(dW, dy) + \varrho(0, dy) \approx (\mathcal{Q}(R_1), R_2), \quad (dW, dy) \in \mathrm{Ran}(\mathcal{Q}) \times \Re^m \tag{23}$$

with the residual satisfying the following condition

$$\|V\,(dW, dy) + \varrho(0, dy) - (\mathcal{Q}(R_1), R_2)\| \leq \varepsilon, \tag{24}$$

where

$$V := \begin{bmatrix} \mathcal{Q} & \\ & 0 \end{bmatrix} + \sigma \begin{bmatrix} \mathcal{Q} \\ -\mathcal{A} \end{bmatrix} \mathcal{U}[\mathcal{Q} \;\; -\mathcal{A}^*],$$

$\mathcal{U}$ is a given self-adjoint positive semidefinite linear operator on $\mathcal{S}^n$ and $\varepsilon > 0, \sigma > 0$ and $\varrho > 0$ are given. Again, instead of solving (23) directly, we can solve a simpler linear system to compute $\mathcal{Q}(dW)$ approximately, as shown in the next proposition. The price to pay is that we now need to solve nonsymmetric linear system instead of a symmetric one.

Proposition 4.2 *Let*

$$\widehat{V} := \begin{bmatrix} \mathcal{I} & \\ & 0 \end{bmatrix} + \sigma \begin{bmatrix} \mathcal{I} \\ -\mathcal{A} \end{bmatrix} \mathcal{U}[\mathcal{Q} \;\; -\mathcal{A}^*].$$

Suppose $(\widehat{dW}, \widehat{dy})$ *is an approximate solution to the following system:*

$$\widehat{V}\,(dW, dy) + \varrho(0, dy) \approx (R_1, R_2) \tag{25}$$

with the residual satisfying

$$\|\widehat{V}\,(\widehat{dW}, \widehat{dy}) + \varrho(0, \widehat{dy}) - (R_1, R_2)\| \leq \frac{\varepsilon}{\max\{\lambda_{\max}(\mathcal{Q}), 1\}}.$$

Let $\widehat{dW}_{\mathcal{Q}} = \Pi_{\mathrm{Ran}(\mathcal{Q})}(\widehat{dW}) \in \mathrm{Ran}(\mathcal{Q})$ *Then* $(\widehat{dW}_{\mathcal{Q}}, \widehat{dy})$ *solves* (23) *with the residual satisfying* (24). *Moreover,* $\mathcal{Q}\widehat{dW}_{\mathcal{Q}} = \mathcal{Q}\widehat{dW}$ *and* $\langle \widehat{dW}_{\mathcal{Q}},\ \mathcal{Q}\widehat{dW}_{\mathcal{Q}}\rangle = \langle \widehat{dW},\ \mathcal{Q}\widehat{dW}\rangle$.

Proof The proof that $\mathcal{Q}\widehat{dW}_{\mathcal{Q}} = \mathcal{Q}\widehat{dW}$ and $\langle \widehat{dW}_{\mathcal{Q}},\ \mathcal{Q}\widehat{dW}_{\mathcal{Q}}\rangle = \langle \widehat{dW},\ \mathcal{Q}\widehat{dW}\rangle$ is the same as in the previous proposition. Observe that $V = \mathrm{Diag}(\mathcal{Q}, \mathcal{I})\widehat{V}$. Then, by using the fact that

$$\begin{aligned}
&\|V\,(\widehat{dW}_{\mathcal{Q}}, \widehat{dy}) + \varrho(0, \widehat{dy})\\
&\quad -(\mathcal{Q}(R_1),\, R_2)\| = \|V\,(\widehat{dW}, \widehat{dy}) + \varrho(0, \widehat{dy}) - (\mathcal{Q}(R_1),\, R_2)\|\\
&\le \|\mathrm{Diag}(\mathcal{Q}, \mathcal{I})\|_2\, \|\widehat{V}\,(\widehat{dW}, \widehat{dy}) + \varrho(0, \widehat{dy}) - (R_1,\, R_2)\|\\
&\le \max\{\lambda_{\max}(\mathcal{Q}),\, 1\}\frac{\varepsilon}{\max\{\lambda_{\max}(\mathcal{Q}),\, 1\}} = \varepsilon,
\end{aligned}$$

we obtain the desired results readily. □

5 Adaption of QSDPNAL for least squares SDP and inequality constrained QSDP

Here we discuss how our algorithm QSDPNAL can be modified and adapted for solving least squares semidefinite programming as well as general QSDP problems with additional unstructured inequality constraints which are not captured by the polyhedral set $\mathcal{K}$.

5.1 The case for least squares semidefinite programming

In this subsection, we show that for least squares semidefinite programming problems, QSDPNAL can be used in a more efficient way to avoid the difficulty of handling the subspace constraint $W \in \mathrm{Ran}(\mathcal{Q})$.

Consider the following least squares semidefinite programming problem

$$\min\ \Big\{\frac{1}{2}\|\mathcal{B}X - d\|^2 + \langle C,\ X\rangle\ \mid\ \mathcal{A}X = b,\ X \in \mathcal{S}^n_+ \cap \mathcal{K}\Big\}, \tag{26}$$

where $\mathcal{A} : \mathcal{S}^n \to \Re^m$ and $\mathcal{B} : \mathcal{S}^n \to \Re^s$ are two linear maps, $C \in \mathcal{S}^n$, $b \in \Re^m$ and $d \in \Re^s$ are given data, $\mathcal{K}$ is a simple nonempty closed convex polyhedral set in $\mathcal{S}^n$.

It is easy to see that (26) can be rewritten as follows

$$\min\ \Big\{\frac{1}{2}\|u\|^2 + \langle C,\ X\rangle\ \mid\ \mathcal{B}X - d = u,\ \mathcal{A}X = b,\ X \in \mathcal{S}^n_+ \cap \mathcal{K}\Big\}. \tag{27}$$

The dual of (27) takes the following form

$$\begin{aligned}
\max\Big\{ &-\delta^*_{\mathcal{K}}(-Z) - \frac{1}{2}\|\xi\|^2 + \langle d,\ \xi\rangle + \langle b,\ y\rangle \mid Z + \mathcal{B}^*\xi + S + \mathcal{A}^*y\\
&= C,\quad S \in \mathcal{S}^n_+\Big\}.
\end{aligned} \tag{28}$$

When QSDPNAL-Phase I is applied to solve (28), instead of solving (20), the linear system corresponding to the quadratic term is given by

$$(\mathcal{I} + \sigma \mathcal{B}\mathcal{B}^*)\xi \approx R, \tag{29}$$

where $R \in \Re^s$ and $\sigma > 0$ are given data. Correspondingly, in applying QSDPNAL-Phase II to (28), the linear system in the SNCG method is given by

$$\left(\begin{bmatrix} \mathcal{I} & \\ & 0 \end{bmatrix} + \sigma \begin{bmatrix} \mathcal{B} \\ \mathcal{A} \end{bmatrix} \mathcal{U} \begin{bmatrix} \mathcal{B}^* & \mathcal{A}^* \end{bmatrix} \right) \begin{bmatrix} d\xi \\ dy \end{bmatrix} \approx \begin{bmatrix} R_1 \\ R_2 \end{bmatrix}, \tag{30}$$

where $R_1 \in \Re^s$ and $R_2 \in \Re^m$ are given data, $\mathcal{U}$ is a given self-adjoint positive semidefinite linear operator on $\mathcal{S}^n$. It is clear that just like (22), one can solve (29) efficiently via the PCG method. For (30), one can also solve it by the PCG method, which is more appealing compared to using a nonsymmetric iterative solver such as the preconditioned BiCGSTAB to solve the nonsymmetric linear system (25).

Remark 5.1 When the polyhedral constraint $X \in \mathcal{K}$ in (26) is absent, i.e., the polyhedral convex set $\mathcal{K} = \mathcal{S}^n$, Jiang, Sun and Toh in [14] have proposed a partial proximal point algorithm for solving the least squares semidefinite programming problem (26). Here our Algorithm QSDPNAL is built to solve the much more general class of convex composite QSDP problems.

5.2 Extension to QSDP problems with inequality constraints

Consider the following general QSDP problem:

$$\min \left\{ \frac{1}{2}\langle X,\ \mathcal{Q}X\rangle + \langle C,\ X\rangle \mid \mathcal{A}_E X = b_E,\ \mathcal{A}_I X \le b_I,\ X \in \mathcal{S}^n_+ \cap \mathcal{K} \right\}, \tag{31}$$

where $\mathcal{A}_E : \mathcal{S}^n \to \Re^{m_E}$ and $\mathcal{A}_I : \mathcal{S}^n \to \Re^{m_I}$ are two linear maps. By adding a slack variable x, we can equivalently rewrite (31) into the following standard form:

$$\begin{array}{l} \min \ \dfrac{1}{2}\langle X,\ \mathcal{Q}X\rangle + \langle C,\ X\rangle \\ \text{s.t.}\ \ \mathcal{A}_E X = b_E, \quad \mathcal{A}_I X + \mathcal{D}x = b_I, \quad X \in \mathcal{S}^n_+ \cap \mathcal{K}, \quad \mathcal{D}x \ge 0, \end{array} \tag{32}$$

where $\mathcal{D} : \Re^{m_I} \to \Re^{m_I}$ is a positive definite diagonal matrix introduced for the purpose of scaling the variable x. The dual of (32) is given by

$$\begin{array}{l} \max \ -\delta^*_{\mathcal{K}}(-Z) - \dfrac{1}{2}\langle W,\ \mathcal{Q}W\rangle + \langle b_E,\ y_E\rangle + \langle b_I,\ y_I\rangle \\ \text{s.t.}\ Z - \mathcal{Q}W + S + \mathcal{A}_E^* y_E + \mathcal{A}_I^* y_I = C, \\ \quad \mathcal{D}^*(s + y_I) = 0, \quad S \in \mathcal{S}^n_+, \quad s \ge 0, \quad W \in \mathrm{Ran}(\mathcal{Q}). \end{array} \tag{33}$$

We can express (33) in a form which is similar to (**D**) as follows:

$$\begin{array}{l}\max\ -\delta^*_{\mathcal{K}}(-Z) - \dfrac{1}{2}\langle W,\ \mathcal{Q}W\rangle + \langle b_E,\ y_E\rangle + \langle b_I,\ y_I\rangle \\ \text{s.t.}\ \begin{pmatrix}\mathcal{I}\\ 0\end{pmatrix} Z - \begin{pmatrix}\mathcal{Q}\\ 0\end{pmatrix} W + \begin{pmatrix}\mathcal{I} & 0\\ 0 & \mathcal{D}^*\end{pmatrix}\begin{pmatrix}S\\ s\end{pmatrix} + \begin{pmatrix}\mathcal{A}_E^* & \mathcal{A}_I^*\\ 0 & \mathcal{D}^*\end{pmatrix}\begin{pmatrix}y_E\\ y_I\end{pmatrix} = \begin{pmatrix}C\\ 0\end{pmatrix}, \\ \quad (S, s) \in \mathcal{S}^n_+ \times \Re^{m_I}_+,\quad W \in \mathrm{Ran}(\mathcal{Q}).\end{array} \tag{34}$$

We can readily extend QSDPNAL to solve the above more general form of (34), and our implementation of QSDPNAL indeed can be used to solve (34).

6 Computational experiments

In this section, we evaluate the performance of our algorithm QSDPNAL for solving large-scale QSDP problems (31). Since QSDPNAL contains two phases, we also report the numerical results obtained by running QSDPNAL-Phase I (a first-order algorithm) alone for the purpose of demonstrating the power and importance of our two-phase framework for solving difficult QSDP problems. In the numerical experiments, we measure the accuracy of an approximate optimal solution (X, Z, W, S, y_E, y_I) for QSDP (31) and its dual by using the following relative KKT residual:

$$\eta_{\mathrm{qsdp}} = \max\{\eta_P, \eta_D, \eta_Z, \eta_{S_1}, \eta_{S_2}, \eta_{I_1}, \eta_{I_2}, \eta_{I_3}, \eta_W\}, \tag{35}$$

where

$$\eta_P = \frac{\|b_E - \mathcal{A}_E X\|}{1 + \|b_E\|},\quad \eta_D = \frac{\|Z - \mathcal{Q}W + S + \mathcal{A}_E^* y_E + \mathcal{A}_I^* y_I - C\|}{1 + \|C\|},\quad \eta_Z = \frac{\|X - \Pi_{\mathcal{K}}(X - Z)\|}{1 + \|X\| + \|Z\|},$$

$$\eta_{S_1} = \frac{|\langle S,\ X\rangle|}{1 + \|S\| + \|X\|},\quad \eta_{S_2} = \frac{\|X - \Pi_{\mathcal{S}^n_+}(X)\|}{1 + \|X\|},\quad \eta_{I_1} = \frac{\|\min(b_I - \mathcal{A}_I X, 0)\|}{1 + \|b_I\|},$$

$$\eta_{I_2} = \frac{\|\max(y_I, 0)\|}{1 + \|y_I\|}, \eta_{I_3} = \frac{|\langle b_I - \mathcal{A}_I X,\ y_I\rangle|}{1 + \|y_I\| + \|b_I - \mathcal{A}_I X\|},\quad \eta_W = \frac{\|\mathcal{Q}W - \mathcal{Q}X\|}{1 + \|\mathcal{Q}\|}.$$

Additionally, we also compute the relative duality gap defined by

$$\eta_{\mathrm{gap}} = \frac{\mathrm{obj}_P - \mathrm{obj}_D}{1 + |\mathrm{obj}_P| + |\mathrm{obj}_D|},$$

where $\mathrm{obj}_P := \frac{1}{2}\langle X,\ \mathcal{Q}X\rangle + \langle C,\ X\rangle$ and $\mathrm{obj}_D := -\delta^*_{\mathcal{K}}(-Z) - \frac{1}{2}\langle W,\ \mathcal{Q}W\rangle + \langle b_E,\ y_E\rangle + \langle b_I,\ y_I\rangle$. We terminate both QSDPNAL and QSDPNAL-Phase I when $\eta_{\mathrm{qsdp}} < 10^{-6}$ with the maximum number of iterations set at 50,000.

In our implementation of QSDPNAL, we always run QSDPNAL-Phase I first to generate a reasonably good starting point to warm start our Phase II algorithm. We terminate the Phase I algorithm and switch to the Phase II algorithm if a solution with a moderate accuracy (say a solution with $\eta_{\mathrm{qsdp}} < 10^{-4}$) is obtained or if the Phase I algorithm reaches the maximum number of iterations (say 1000 iterations). If the underlying problems contain inequality or polyhedral constraints, we further employ a restarting strategy similar to the one in [33], i.e., when the progress of QSDPNAL-Phase II is not

satisfactory, we will restart the whole QSDPNAL algorithm by using the most recently computed (Z, W, S, y, X, σ) as the initial point. In addition, we also adopt a dynamic tuning strategy to adjust the penalty parameter σ appropriately based on the progress of the primal and dual feasibilities of the computed iterates.

All our computational results are obtained from a workstation running on 64-bit Windows Operating System having 16 cores with 32 Intel Xeon E5-2650 processors at 2.60 GHz and 64 GB memory. We have implemented QSDPNAL in MATLAB version 7.13.

6.1 Evaluation of QSDPNAL on the nearest correlation matrix problems

Our first test example is the problem of finding the nearest correlation matrix (NCM) to a given matrix $G \in \mathcal{S}^n$:

$$\min \left\{ \frac{1}{2}\|H \circ (X - G)\|_F^2 \mid \mathrm{diag}(X) \;=\; e,\; X \in \mathcal{S}^n_+ \cap \mathcal{K} \right\}, \tag{36}$$

where $H \in \mathcal{S}^n$ is a nonnegative weight matrix, $e \in \Re^n$ is the vector of all ones, and $\mathcal{K} = \{W \in \mathcal{S}^n \mid L \le W \le U\}$ with $L, U \in \mathcal{S}^n$ being given matrices.

In our numerical experiments, we first take a matrix $\widehat{G}$, which is a correlation matrix generated from gene expression data from [16]. For testing purpose, we then perturb $\widehat{G}$ to

$$G := (1 - \alpha)\widehat{G} + \alpha E,$$

where $\alpha \in (0, 1)$ is a given parameter and E is a randomly generated symmetric matrix with entries uniformly distributed in $[-1, 1]$ except for its diagonal elements which are all set to 1. The weight matrix H is generated from a weight matrix H_0 used by a hedge fund company. The matrix H_0 is a 93×93 symmetric matrix with all positive entries. It has about 24% of the entries equal to 10^{-5} and the rest are distributed in the interval $[2, 1.28 \times 10^3]$. The MATLAB code for generating the matrix H is given by

```
tmp = kron(ones(110,110),H0); H = tmp(1:n,1:n); H = (H'+H)/2.
```

The reason for using such a weight matrix is because the resulting problems generated are more challenging to solve as opposed to a randomly generated weight matrix. We also test four more instances, namely `PDidx2000`, `PDidx3000`, `PDidx5000` and `PDidx10000`, where the raw correlation matrix $\widehat{G}$ is generated from the probability of default (PD) data obtained from the RMI Credit Research Initiative[2] at the National University of Singapore. We consider two choices of $\mathcal{K}$, i.e., case (i): $\mathcal{K} = \mathcal{S}^n$ and case (ii): $\mathcal{K} = \{X \in \mathcal{S}^n \mid X_{ij} \ge -0.5,\ \forall\, i, j = 1, \ldots, n\}$.

In Table 1, we report the numerical results obtained by QSDPNAL and QSDPNAL-Phase I in solving various instances of the H-weighted NCM problem (36). In the table, "it (subs)" denotes the number of outer iterations with subs in the parenthesis

[2] https://www.rmicri.org.

Table 1 The performance of QSDPNAL and QSDPNAL-Phase I on H-weighted NCM problems (dual of (36)) (accuracy $= 10^{-6}$). In the table, "a" stands for QSDPNAL and "b" stands for QSDPNAL-Phase I, respectively. The computation time is in the format of "hours:minutes:seconds"

Problem	n	α	iter.a it (subs) | itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
$\mathcal{K} = \mathcal{S}^n$							
Lymph	587	0.10	12 (40) | 52	251	9.1–7 | 9.1–7	8.2–7 | −3.9–7	13 | 23
Lymph	587	0.05	11 (32) | 38	205	9.5–7 | 9.9–7	7.5–7 | −4.1–7	09 | 19
ER	692	0.10	12 (41) | 54	250	9.8–7 | 9.9–7	5.4–7 | −4.8–7	17 | 33
ER	692	0.05	12 (38) | 43	218	7.3–7 | 9.7–7	2.5–7 | −4.4–7	14 | 28
Arabidopsis	834	0.10	12 (42) | 56	285	8.5–7 | 9.9–7	2.8–7 | −5.3–7	27 | 57
Arabidopsis	834	0.05	12 (41) | 44	230	8.0–7 | 9.5–7	−6.8–8 | −4.5–7	24 | 46
Leukemia	1255	0.10	12 (41) | 62	340	8.4–7 | 9.9–7	3.1–7 | −5.4–7	1:08 | 2:48
Leukemia	1255	0.05	12 (38) | 49	248	7.6–7 | 8.7–7	−1.3–7 | −4.5–7	58 | 2:06
hereditarybc	1869	0.10	13 (47) | 76	393	6.4–7 | 9.9–7	−2.2–7 | −9.8–7	3:01 | 7:10
hereditarybc	1869	0.05	13 (45) | 60	311	8.6–7 | 9.9–7	−4.7–7 | *−1.0−6*	2:39 | 5:44
PDidx2000	2000	0.10	13 (51) | 131	590	9.5–7 | 9.9–7	2.4–7 | −8.5–7	5:04 | 11:43
PDidx2000	2000	0.05	14 (58) | 139	626	7.5–7 | 9.9–7	−5.6–8 | −9.5–7	5:52 | 12:41
PDidx3000	3000	0.10	14 (55) | 145	1201	8.1–7 | 9.9–7	−2.8–7 | *2.1−6*	14:59 | 1:15:01
PDidx3000	3000	0.05	14 (58) | 136	1263	6.8–7 | 9.7–7	−2.6–7 | *2.0−6*	14:50 | 1:19:27
PDidx5000	5000	0.10	15 (63) | 189	1031	8.0–7 | 9.9–7	−1.9–7 | *1.8−6*	1:17:47 | 4:17:10
PDidx5000	5000	0.05	14 (59) | 164	1699	9.2–7 | 9.9–7	−3.3–7 | −1.3–7	1:11:46 | 6:18:29
PDidx10000	10,000	0.10	16 (71) | 200	2572	7.1–7 | 9.9–7	1.6–7 | −1.5–7	9:57:18 | 60:07:08
PDidx10000	10,000	0.05	16 (73) | 200	2532	9.5–7 | 9.9–7	4.7–8 | 1.4–7	10:34:31 | 59:34:13
$\mathcal{K} = \{X \in \mathcal{S}^n \mid X_{ij} \geq -0.5\ \forall\, i, j = 1, \ldots, n\}$							
Lymph	587	0.10	5 (14) | 129	244	9.8–7 | 9.9–7	−1.0–7 | −4.4–7	18 | 30
Lymph	587	0.05	5 (12) | 120	257	9.9–7 | 9.9–7	−3.4–7 | −4.2–7	15 | 28

Table 1 continued

Problem	n	α	iter.a it (subs) | itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
ER	692	0.10	5 (14) | 126	266	9.9–7 | 9.9–7	− 1.5–7 | − 5.1–7	22 | 40
ER	692	0.05	5 (14) | 117	217	8.4–7 | 9.9–7	− 2.7–7 | − 4.4–7	21 | 32
Arabidopsis	834	0.10	6 (16) | 240	472	9.9–7 | 9.9–7	− 5.4–7 | − 6.0–7	1:03 | 1:56
Arabidopsis	834	0.05	6 (15) | 240	442	8.5–7 | 9.9–7	− 4.4–7 | − 5.6–7	1:02 | 1:46
Leukemia	1255	0.10	7 (22) | 188	333	9.9–7 | 9.9–7	− 4.4–7 | − 5.5–7	2:10 | 3:06
Leukemia	1255	0.05	7 (19) | 159	253	9.9–7 | 9.9–7	− 5.4–7 | − 5.3–7	1:46 | 2:18
hereditarybc	1869	0.10	8 (22) | 397	577	9.3–7 | 9.9–7	− 8.0–7 | − 8.9–7	10:28 | 12:59
hereditarybc	1869	0.05	8 (22) | 361	472	9.6–7 | 9.9–7	− 8.1–7 | − 8.6–7	9:39 | 10:04
PDidx2000	2000	0.10	20 (52) | 672	716	9.9–7 | 9.9–7	− 6.8–7 | − 7.9–7	21:32 | 17:42
PDidx2000	2000	0.05	22 (60) | 756	1333	9.6–7 | 5.8–7	− 6.3–7 | − 4.0–7	25:20 | 39:34
PDidx3000	3000	0.10	34 (101) | 659	1647	9.9–7 | 9.9–7	− 7.0–7 | − 9.4–7	1:14:15 | 1:53:13
PDidx3000	3000	0.05	41 (117) | 728	1538	9.9–7 | 9.9–7	− 6.2–7 | *− 1.2−6*	1:21:13 | 1:50:47
PDidx5000	5000	0.10	29 (79) | 829	1484	9.3–7 | 8.4–7	− 5.5–7 | 6.4–7	5:00:35 | 7:25:19
PDidx5000	5000	0.05	33 (107) | 1081	1722	9.9–7 | 9.9–7	− 6.4–7 | − 1.5–7	6:30:35 | 7:16:08
PDidx10000	10,000	0.10	42 (136) | 1289	2190	9.9–7 | 9.9–7	− 7.1–7 | 2.6–7	58:44:49 | 64:17:14
PDidx10000	10,000	0.05	40 (122) | 1519	3320	9.9–7 | 4.5–7	− 6.6–7 | − 1.7–8	65:13:19 | 94:53:10

Numbers in italic indicate that they are greater than the stopping tolerance

indicating the number of inner iterations of QSDPNAL-Phase II and "itSCB" stands for the total number of iterations used in QSDPNAL-Phase I. We can see from Table 1 that QSDPNAL is more efficient than the purely first-order algorithm QSDPNAL-Phase I. In particular, for the instance `PDidx10000` where the matrix dimension $n = 10{,}000$, we are able to solve the problem in about 11 h while the purely first-order method QSDPNAL-Phase I needs about 60 h.

6.2 Evaluation of QSDPNAL on instances generated from BIQ problems

Based on the SDP relaxation of a binary integer quadratic (BIQ) problem considered in [28], we construct our second QSDP test example as follows:

$$\begin{aligned}(\text{QSDP-BIQ}) \quad \min\ & \frac{1}{2}\langle X,\ \mathcal{Q}X\rangle + \frac{1}{2}\langle Q,\ Y\rangle + \langle c,\ x\rangle \\ \text{s.t. } & \operatorname{diag}(Y) - x = 0, \quad \alpha = 1, \quad X = \begin{pmatrix} Y & x \\ x^T & \alpha \end{pmatrix} \in \mathcal{S}^n_+, \quad X \in \mathcal{K}, \\ & -Y_{ij} + x_i \geq 0,\ -Y_{ij} + x_j \geq 0,\ Y_{ij} - x_i - x_j \geq -1, \\ & \forall\, i < j,\ j = 2, \ldots, n-1,\end{aligned}$$

where the convex set $\mathcal{K} = \{X \in \mathcal{S}^n \mid X \geq 0\}$. Here $\mathcal{Q} : \mathcal{S}^n \to \mathcal{S}^n$ is a self-adjoint positive semidefinite linear operator defined by

$$\mathcal{Q}(X) = \frac{1}{2}(AXB + BXA) \tag{37}$$

with $A, B \in \mathcal{S}^n_+$ being matrices truncated from two different large correlation matrices (generated from Russell 1000 and Russell 2000 index, respectively) fetched from Yahoo finance by MATLAB. In our numerical experiments, the test data for Q and c are taken from the Biq Mac Library maintained by Wiegele, which is available at http://biqmac.uni-klu.ac.at/biqmaclib.html.

Table 2 reports the numerical results for QSDPNAL and QSDPNAL-Phase I in solving some large scale QSDP-BIQ problems. Note that from the numerical experiments conducted in [6], one can clearly conclude that QSDPNAL-Phase I (a variant of SCB-isPADMM) is the most efficient first-order algorithm for solving QSDP-BIQ problems with a large number of inequality constraints. Even then, one can observe from Table 2 that QSDPNAL is still faster than QSDPNAL-Phase I on most of the problems tested.

6.3 Evaluation of QSDPNAL on instances generated from QAP problems

Next we test the following QSDP problem motivated from the SDP relaxation of a quadratic assignment problem (QAP) considered in [21]. The SDP relaxation we used is adopted from [33] but we add a convex quadratic term in the objective to modify it into a QSDP problem. Specifically, given the data matrices $A_1, A_2 \in \mathcal{S}^l$ of a QAP problem, the problem we test is given by:

Table 2 Same as Table 1 but for QSDP-BIQ problems

d	$m_E; m_I \mid n$	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
be200.3.1	201 ; 59,700 | 201	66 (135) | 3894	4701	7.8–7 | 9.8–7	−3.5–7 | −7.2–7	3:37 | 3:57
be200.3.2	201 ; 59,700 | 201	37 (74) | 2969	13,202	9.7–7 | 9.9–7	−2.1–7 | −6.7–8	2:42 | 12:20
be200.3.3	201 ; 59,700 | 201	51 (107) | 5220	10,375	8.1–7 | 9.9–7	−1.1–7 | −6.5–7	5:00 | 8:52
be200.3.4	201 ; 59,700 | 201	36 (72) | 3484	4966	9.8–7 | 9.9–7	−1.6–7 | −4.1–7	3:15 | 4:14
be200.3.5	201 ; 59,700 | 201	22 (44) | 2046	3976	9.8–7 | 9.9–7	−5.9–8 | −3.0–7	1:53 | 3:28
be250.1	251 ; 93,375 | 251	98 (196) | 6931	12,220	9.9–7 | 9.9–7	3.2–7 | 3.5–8	8:11 | 14:07
be250.2	251 ; 93,375 | 251	81 (169) | 6967	16,421	9.3–7 | 9.9–7	3.2–7 | −5.7–7	8:35 | 20:01
be250.3	251 ; 93,375 | 251	123 (250) | 7453	9231	9.3–7 | 9.8–7	−1.7–7 | −5.1–7	9:27 | 10:25
be250.4	251 ; 93,375 | 251	36 (72) | 3583	4542	9.9–7 | 9.9–7	5.2–8 | −2.1–7	4:31 | 5:06
be250.5	251 ; 93,375 | 251	99 (198) | 5004	12,956	8.3–7 | 9.9–7	1.8–7 | −1.8–7	6:38 | 15:52
bqp500-1	501 ; 374,250 | 501	62 (131) | 5220	11,890	9.9–7 | 9.9–7	−7.1–7 | −8.2–8	37:56 | 1:23:58
bqp500-2	501 ; 374,250 | 501	41 (84) | 3610	8159	5.5–7 | 9.9–7	−3.8–7 | −8.7–8	24:01 | 55:14
bqp500-3	501 ; 374,250 | 501	89 (200) | 5877	6402	9.9–7 | 8.6–7	5.4–7 | −1.9–7	40:29 | 41:51
bqp500-4	501 ; 374,250 | 501	95 (256) | 7480	11,393	6.3–7 | 9.9–7	−1.5–7 | −1.1–7	56:12 | 1:17:56
bqp500-5	501 ; 374,250 | 501	107 (247) | 6976	8823	5.1–7 | 9.9–7	6.2–7 | −1.0–7	52:24 | 59:11
bqp500-6	501 ; 374,250 | 501	159 (412) | 10,461	9587	8.3–7 | 9.9–7	−6.2–7 | −1.3–7	1:18:11 | 1:04:41
bqp500-7	501 ; 374,250 | 501	92 (223) | 8585	9066	8.1–7 | 9.9–7	4.7–8 | −1.1–7	1:00:52 | 1:00:35
bqp500-8	501 ; 374,250 | 501	68 (140) | 5828	7604	6.7–7 | 9.9–7	−4.7–8 | −1.1–7	40:56 | 51:58
bqp500-9	501 ; 374,250 | 501	50 (108) | 4704	11,613	9.5–7 | 9.9–7	−3.7–7 | −9.8–8	34:05 | 1:21:17
bqp500-10	501 ; 374,250 | 501	71 (163) | 6462	8474	8.7–7 | 9.9–7	−6.2–7 | −8.7–8	48:07 | 57:33
gka1e	201 ; 59,700 | 201	74 (163) | 5352	9071	9.2–7 | 9.9–7	−3.0–7 | −2.9–7	7:59 | 9:35

Table 2 continued

d	$m_E; m_I \mid n$	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
gka2e	201 ; 59,700 | 201	49 (98) | 4008	6659	9.2–7 | 9.9–7	5.0–8 | − 1.7–7	4:17 | 6:29
gka3e	201 ; 59,700 | 201	35 (71) | 2731	4103	8.3–7 | 9.7–7	2.3–7 | − 2.2–8	2:59 | 4:14
gka4e	201 ; 59,700 | 201	34 (68) | 2999	3430	9.9–7 | 9.9–7	− 1.7–7 | − 4.6–7	3:20 | 3:21
gka5e	201 ; 59,700 | 201	43 (90) | 3367	2712	9.9–7 | 9.9–7	− 4.9–8 | − 6.5–8	3:54 | 2:47

$$\begin{aligned}(\text{QSDP-QAP}) \quad \min\ & \frac{1}{2}\langle X,\ \mathcal{Q}X\rangle + \langle A_2 \otimes A_1,\ X\rangle \\ \text{s.t.}\ & \textstyle\sum_{i=1}^{l} X^{ii} = I,\ \langle I,\ X^{ij}\rangle = \delta_{ij} \quad \forall\, 1 \le i \le j \le l, \\ & \langle E,\ X^{ij}\rangle = 1 \quad \forall\, 1 \le i \le j \le l, \quad X \in \mathcal{S}^n_+,\ X \in \mathcal{K},\end{aligned}$$

where $n = l^2$, and $X^{ij} \in \Re^{l\times l}$ denotes the (i, j)-th block of X when it is partitioned uniformly into an $l \times l$ block matrix with each block having dimension $l \times l$. The convex set $\mathcal{K} = \{X \in \mathcal{S}^n \mid X \ge 0\}$, E is the matrix of ones, and $\delta_{ij} = 1$ if $i = j$, and 0 otherwise. Note that here we use the same self-adjoint positive semidefinite linear operator $\mathcal{Q} : \mathcal{S}^n \to \mathcal{S}^n$ constructed in (37). In our numerical experiments, the test instances (A_1, A_2) are taken from the QAP Library [5].

In Table 3, we present the numerical results for QSDPNAL and QSDPNAL-Phase I in solving some large scale QSDP-QAP problems. It is interesting to note that QSDPNAL can solve all the 73 difficult QSDP-QAP problems to an accuracy of 10^{-6} efficiently, while the purely first-order algorithm QSDPNAL-Phase I can only solve 2 of the problems (chr20a and tai25a) to the required accuracy. The superior numerical performance of QSDPNAL over QSDPNAL-Phase I clearly demonstrates the importance and necessity of our proposed two-phase algorithm for which second-order information is incorporated in the inexact augmented Lagrangian algorithm in Phase II. Note that for QSDP-QAP problems, the iteration counts of QSDPNAL sometimes can vary on different computers. The root cause of this phenomenon is the accumulation of the rounding errors of the PCG steps used in QSDPNAL. Indeed, for these difficult problems, a moderate number (sometimes can be up to 200) of PCG steps is needed for solving the corresponding Newton system.

6.4 Evaluation of QSDPNAL on instances generated from sensor network localization problems

We also test the QSDP problems arising from the following sensor network localization problems with m anchors and l sensors:

$$\min_{u_1,\ldots,u_l\in\Re^d} \left\{ \tfrac{1}{2} \textstyle\sum_{(i,j)\in\mathcal{N}} \left(\|u_i - u_j\|^2 - d_{ij}^2\right)^2 \ \mid\ \|u_i - a_k\|^2 = d_{ik}^2,\ (i,k) \in \mathcal{M} \right\}, \tag{38}$$

where the location of each anchor $a_k \in \Re^d$, $k = 1, \ldots, m$ is known, and the location of each sensor $u_i \in \Re^d$, $i = 1, \ldots, l$, is to be determined. The distance measures $\{d_{ij} \mid (i, j) \in \mathcal{N}\}$ and $\{d_{ik} \mid (i, k) \in \mathcal{M}\}$ are known pair-wise distances between sensor-sensor pairs and sensor-anchor pairs, respectively. Note that our model (38) is a variant of the model studied in [3]. Let $U = [u_1\ u_2\ \ldots u_l] \in \Re^{d\times l}$ be the position matrix that needs to be determined. We know that

$$\|u_i - u_j\|^2 = e_{ij}^T U^T U e_{ij}, \qquad \|u_i - a_k\|^2 = a_{ik}^T [U\ I_d]^T [U\ I_d] a_{ik},$$

where $e_{ij} = e_i - e_j$ and $a_{ik} = [e_i; -a_k]$. Here, e_i is the ith unit vector in $\Re^l$, and I_d is the $d \times d$ identity matrix. Let $g_{ik} = a_{ik}$ for $(i, k) \in \mathcal{M}$, $g_{ij} = [e_{ij}; \mathbf{0}_m]$ for $(i, j) \in \mathcal{N}$,

Table 3 Same as Table 1 but for QSDP-QAP problems

Problem	m_E | n	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
chr12a	232 ; 144	45 (239) | 1969	50,000	9.9–7 | *2.2–6*	*−6.0−6*|*−2.5−5*	41 | 6:34
chr12b	232 ; 144	56 (324) | 2428	50,000	9.9–7 | *3.7–6*	*−2.0−5*|*−6.0−5*	50 | 6:27
chr12c	232 ; 144	56 (358) | 2201	50,000	9.9–7 | *4.5–6*	*−1.6−5*|*−6.1−5*	46 | 6:27
chr15a	358 ; 225	84 (648) | 2866	50,000	9.9–7 | *5.7–6*	*−1.6−5*|*−1.0−4*	2:25 | 12:23
chr15b	358 ; 225	90 (584) | 4700	50,000	9.9–7 | *7.3–6*	*−1.3−5*|*−1.3−4*	3:07 | 12:31
chr15c	358 ; 225	65 (425) | 2990	50,000	9.9–7 | *6.8–6*	*−2.4−5*|*−9.7−5*	1:58 | 12:40
chr18a	511 ; 324	256 (1957) | 6003	50,000	7.3–7 | *6.1–6*	*−1.8−5*|*−1.3−4*	14:34 | 22:26
chr18b	511 ; 324	86 (565) | 3907	50,000	9.9–7 | *8.4–6*	*−1.8−5*|*−1.6−4*	5:26 | 22:19
chr20a	628 ; 400	39 (274) | 1751	4133	9.5–7 | *9.7–7*	*−3.3−5*|*−3.4−5*	4:50 | 5:46
chr20b	628 ; 400	72 (490) | 4044	50,000	9.6–7 | *9.1–6*	*−3.7−5*|*−1.4−4*	12:30 | 58:57
chr20c	628 ; 400	144 (981) | 5242	50,000	9.9–7 | *1.4–5*	*−3.1−5*|*−3.1−4*	21:56 | 55:41
chr22a	757 ; 484	67 (473) | 2804	50,000	9.9–7 | *5.0–6*	*−1.0−5*|*−7.6−5*	13:49 | 1:21:01
chr22b	757 ; 484	69 (505) | 3581	50,000	9.9–7 | *6.5–6*	*−1.2−5*|*−1.1−4*	17:12 | 1:19:48
els19	568 ; 361	43 (403) | 2437	50,000	9.8–7 | *1.2–6*	*−4.2−6*|*−8.6−6*	4:39 | 1:04:08
esc16a	406 ; 256	86 (506) | 5446	50,000	9.9–7 | *8.3–6*	*−2.5−5*|*−1.3−4*	4:47 | 16:54
esc16b	406 ; 256	157 (1425) | 9222	50,000	9.9–7 | *1.3–5*	*−3.7−5*|*−2.7−4*	11:18 | 16:56
esc16c	406 ; 256	188 (1404) | 13,806	50,000	9.9–7 | *1.2–5*	*−4.6−5*|*−3.5−4*	14:29 | 16:57
esc16d	406 ; 256	101 (603) | 8043	50,000	9.9–7 | *4.8–6*	*−1.1−5*|*−7.4−5*	6:13 | 16:55
esc16e	406 ; 256	110 (847) | 4286	50,000	9.9–7 | *4.8–6*	*−1.4−5*|*−5.6−5*	5:50 | 16:35
esc16g	406 ; 256	85 (581) | 3818	50,000	9.9–7 | *7.2–6*	*−2.2−5*|*−9.4−5*	4:23 | 16:44
esc16h	406 ; 256	228 (1732) | 11,733	50,000	8.6–7 | *8.6–6*	*−9.3−6*|*−8.7−5*	13:58 | 16:21

Table 3 continued

Problem	m_E | n	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
esc16i	406 ; 256	41 (307) | 3165	50,000	9.6–7 | *4.6 − 6*	*−2.0−5*|*−6.0−5*	2:28 | 16:54
esc16j	406 ; 256	163 (1179) | 5603	50,000	9.9–7 | *6.6−6*	*−2.0−5*|*−1.0−4*	8:03 | 16:24
esc32b	1582 ; 1024	80 (456) | 5026	50,000	9.9–7 | *1.5−4*	*−2.9−5*|*−5.3−4*	1:53:02 | 7:42:25
esc32c	1582 ; 1024	105 (667) | 4203	50,000	8.9–7 | *7.2−6*	*−1.0−5*|*−7.3−5*	2:40:09 | 7:36:49
esc32d	1582 ; 1024	141 (909) | 4852	50,000	9.9–7 | *5.7−6*	*−8.6−6*|*−6.4−5*	3:09:45 | 7:19:17
had12	232 ; 144	53 (320) | 2903	50,000	9.3–7 | *3.3−6*	*−7.2−6*|*−2.7−5*	55 | 6:52
had14	313 ; 196	60 (443) | 3634	50,000	9.9–7 | *4.5−6*	*−6.3−6*|*−2.8−5*	1:50 | 11:21
had16	406 ; 256	208 (1616) | 7604	50,000	8.1–7 | *9.9−6*	*−4.7−6*|*−9.1−5*	10:49 | 16:54
had18	511 ; 324	82 (537) | 4367	50,000	9.9–7 | *9.2−6*	*−1.5−5*|*−7.3−5*	6:10 | 24:46
had20	628 ; 400	121 (848) | 5024	50,000	9.5–7 | *1.0−5*	*−1.2−5*|*−1.0−4*	20:30 | 51:08
kra30a	1393 ; 900	107 (674) | 4665	50,000	9.5–7 | *6.5−6*	*−6.7−5*|*−1.7−4*	1:51:20 | 7:23:56
kra30b	1393 ; 900	107 (674) | 4853	50,000	9.9–7 | *6.5−6*	*−5.7−5*|*−1.7−4*	2:00:03 | 8:02:08
kra32	1582 ; 1024	106 (636) | 6875	50,000	9.9–7 | *7.4−6*	*−3.6−5*|*−1.6−4*	2:47:33 | 10:28:26
lipa30a	1393 ; 900	64 (451) | 2924	50,000	9.9–7 | *5.6−6*	*−6.8−6*|*−3.1−5*	1:10:12 | 6:52:34
lipa30b	1393 ; 900	257 (1918) | 7507	50,000	9.9–7 | *7.0−6*	*−1.9−6*|*−1.9−4*	4:28:38 | 7:18:10
lipa40a	2458 ; 1600	51 (349) | 2193	50,000	7.7–7 | *4.2−6*	*−2.3−6*|*−2.0−5*	3:03:58 | 23:53:41
lipa40b	2458 ; 1600	156 (1339) | 4750	50,000	9.1–7 | *3.9−6*	*6.0−6*|*−8.9−5*	9:36:10 | 18:57:01
nug12	232 ; 144	84 (478) | 4068	50,000	9.8–7 | *5.4−6*	*−3.0−5*|*−1.1−4*	1:32 | 6:34
nug14	313 ; 196	93 (610) | 4953	50,000	9.7–7 | *6.9−6*	*−2.6−5*|*−1.1−4*	3:12 | 10:27
nug15	358 ; 225	102 (660) | 5627	50,000	7.0–7 | *1.1−5*	*−2.2−5*|*−1.7−4*	4:12 | 12:42
nug16a	406 ; 256	86 (530) | 4945	50,000	9.9–7 | *7.2−6*	*−2.3−5*|*−1.1−4*	4:39 | 18:17

Table 3 continued

Problem	m_E | n	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
nug16b	406 ; 256	97 (631) | 4777	50,000	9.9–7 |*1.2−5*	*−2.5−5*|*−2.0−4*	5:19 | 19:06
nug17	457 ; 289	110 (772) | 5365	50,000	9.9–7 |*1.3−5*	*−2.4−5*|*−1.8−4*	7:41 | 22:47
nug18	511 ; 324	85 (559) | 4367	50,000	9.9–7 |*6.1−6*	*−3.3−5*|*−9.9−5*	6:10 | 26:20
nug20	628 ; 400	114 (746) | 5220	50,000	9.9–7 |*8.6−6*	*−2.3−5*|*−1.3−4*	19:25 | 55:36
nug21	691 ; 441	84 (569) | 4322	50,000	9.7–7 |*6.8−6*	*−4.0−5*|*−1.1−4*	18:48 | 1:09:39
nug22	757 ; 484	121 (822) | 5822	50,000	9.6–7 |*8.3−6*	*−4.1−5*|*−1.3−4*	34:03 | 2:08:02
nug24	898 ; 576	89 (542) | 4345	50,000	9.9–7 |*6.5−6*	*−3.2−5*|*−1.1−4*	34:08 | 3:04:07
nug25	973 ; 625	129 (860) | 5801	50,000	9.9–7 |*6.9−6*	*−2.2−5*|*−1.1−4*	1:09:12 | 3:41:22
nug27	1132 ; 729	148 (951) | 8576	50,000	9.9–7 |*8.3−6*	*−2.6−5*|*−1.3−4*	2:09:22 | 4:59:06
nug28	1216 ; 784	119 (758) | 6389	50,000	9.7–7 |*7.8−6*	*−2.9−5*|*−1.1−4*	1:43:52 | 5:50:50
nug30	1393 ; 900	105 (777) | 4912	50,000	9.9–7 |*9.3−6*	*−2.6−5*|*−1.2−4*	2:08:56 | 7:40:43
rou12	232 ; 144	78 (418) | 6600	50,000	9.9–7 |*4.4−6*	*−2.2−5*|*−9.3−5*	2:13 | 6:36
rou15	358 ; 225	106 (639) | 5952	50,000	9.9–7 |*5.4−6*	*−2.7−5*|*−1.0−4*	4:12 | 13:02
rou20	628 ; 400	65 (359) | 4238	50,000	9.9–7 |*3.6−6*	*−2.5−5*|*−6.1−5*	11:53 | 1:00:07
scr12	232 ; 144	56 (295) | 2205	50,000	9.9–7 |*3.2−6*	*−7.5−6*|*−4.0−5*	43 | 6:57
scr15	358 ; 225	121 (769) | 5730	50,000	7.4–7 |*1.0−5*	*−2.1−5*|*−1.9−4*	4:39 | 12:45
scr20	628 ; 400	89 (590) | 5621	50,000	9.9–7 |*8.3−6*	*−4.1−5*|*−1.6−4*	18:49 | 1:01:19
tai12a	232 ; 144	110 (807) | 6090	50,000	9.9–7 |*8.9−6*	*−1.8−5*|*−1.3−4*	2:40 | 6:15
tai12b	232 ; 144	123 (856) | 6323	50,000	8.6–7 |*7.3−6*	*−2.5−5*|*−1.1−4*	2:43 | 6:19
tai15a	358 ; 225	67 (405) | 4301	50,000	9.4–7 |*3.0−6*	*−2.8−5*|*−5.9−5*	2:48 | 13:09
tai17a	457 ; 289	95 (569) | 6142	50,000	9.9–7 |*3.8−6*	*−2.0−5*|*−6.5−5*	6:33 | 19:23

Table 3 continued

Problem	m_E | n	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
tai20a	628 ; 400	87 (498) | 4762	50,000	9.7–7 | *3.0−6*	*−2.2−5*|*−5.4−5*	15:00 | 1:00:55
tai25a	973 ; 625	25 (138) | 3438	10,084	9.9–7 | *9.9−7*	*9.4−6*|*−1.5−5*	17:46 | 47:55
tai25b	973 ; 625	164 (1219) | 7803	50,000	9.8–7 | *1.4−5*	*−4.8−5*|*−2.6−4*	1:33:02 | 3:33:13
tai30a	1393 ; 900	97 (546) | 4270	50,000	7.8–7 | *2.8−6*	*−1.4−5*|*−4.1−5*	1:22:01 | 7:21:02
tai30b	1393 ; 900	162 (1229) | 6845	50,000	9.9–7 | *1.2−5*	*−3.3−5*|*−2.1−4*	3:10:41 | 7:21:29
tai35a	1888 ; 1225	95 (537) | 5781	50,000	9.9–7 | *2.8−6*	*−9.2−6*|*−3.5−5*	3:34:56 | 15:04:40
tai35b	1888 ; 1225	152 (1195) | 5303	50,000	9.6–7 | *1.2−5*	*−3.8−5*|*−1.9−4*	5:56:59 | 13:37:28
tai40a	2458 ; 1600	79 (398) | 6381	50,000	9.1–7 | *3.0−6*	*−1.7−5*|*−3.4−5*	6:01:56 | 23:07:21
tho30	1393 ; 900	116 (761) | 5468	50,000	8.9–7 | *8.0−6*	*−3.1−5*|*−1.3−4*	2:15:55 | 7:39:54
tho40	2458 ; 1600	122 (762) | 3834	50,000	9.9–7 | *7.4−6*	*−2.9−5*|*−1.0−4*	6:28:52 | 26:40:25

Numbers in italic indicate that they are greater than the stopping tolerance

and

$$V = U^T U, \qquad X = [V \; U^T; U \; I_d] \in \mathcal{S}^{(d+l)\times(d+l)}.$$

Following the same approach in [3], we can obtain the following QSDP relaxation model with a regularization term for (38)

$$\begin{array}{l} \min \; \frac{1}{2}\sum_{(i,j)\in\mathcal{N}} \left(g_{ij}^T X g_{ij} - d_{ij}^2\right)^2 - \lambda\langle I_{n+d} - aa^T, \; X\rangle \\ \text{s.t. } g_{ik}^T X g_{ik} = d_{ik}^2, \; (i,k) \in \mathcal{M}, \quad X \succeq 0, \end{array} \tag{39}$$

where λ is a given positive regularization parameter, $a = [\hat{e}; \hat{a}]$ with $\hat{e} = e/\sqrt{l+m}$ and $\hat{a} = \sum_{k=1}^m a_k/\sqrt{l+m}$. Here $e \in \Re^n$ is the vector of all ones.

The test examples are generated in the following manner. We first randomly generate l points $\{\hat{x}_i \in \Re^d \mid i = 1, \ldots, l\}$ in $[-0.5, 0.5]^d$. Then, the edge set $\mathcal{N}$ is generated by considering only pairs of points that have distances less than a given positive number R, i.e.,

$$\mathcal{N} = \{(i,j) \mid \|\hat{u}_i - \hat{u}_j\| \le R, \; 1 \le i < j \le l\}.$$

Given m anchors $\{a_k \in \Re^d \mid k = 1, \ldots, m\}$, the edge set $\mathcal{M}$ is similarly given by

$$\mathcal{M} = \{(i,k) \mid \|\hat{u}_i - a_k\| \le R, \; 1 \le i \le l, \; 1 \le k \le m\}.$$

We also assume that the observed distances d_{ij} are perturbed by random noises ε_{ij} as follows:

$$d_{ij} = \hat{d}_{ij}|1 + \tau\varepsilon_{ij}|, \quad (i,j) \in \mathcal{N},$$

where $\hat{d}_{ij}$ is the true distance between point i and j, ε_{ij} are assumed to be independent standard Normal random variables, τ is the noise parameter. For the numerical experiments, we generate 10 instances where the number of sensors l ranges from 250 to 1500 and the dimension d is set to be 2 or 3. We set the noise factor $\tau = 10\%$. The 4 anchors for the two dimensional case ($d = 2$) are placed at

$$(\pm 0.3, \pm 0.3),$$

and the positions of the anchors for the three dimensional case ($d = 3$) are given by

$$\begin{pmatrix} 1/3 & 2/3 & 2/3 & 1/3 \\ 1/3 & 2/3 & 1/3 & 2/3 \\ 1/3 & 1/3 & 2/3 & 2/3 \end{pmatrix} - 0.5E,$$

where E is the 3×3 matrix of all ones.

Table 4 Same as Table 1 but for the sensor network localization problems [dual of (39)]

d	$m_E \mid n \mid R$	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
2	452 | 252 | 0.50	12 (74) | 652	24,049	7.1–7 | 9.9–7	−9.0–7 | 6.8–7	47 | 6:10
2	548 | 502 | 0.36	12 (62) | 1000	12,057	7.6–7 | 9.9–7	*−9.2−6*|*−9.1−6*	1:49 | 17:25
2	633 | 802 | 0.28	17 (85) | 1000	27,361	3.0–7 | 9.9–7	*−2.4−6*|*−9.9−6*	5:42 | 1:59:16
2	684 | 1002 | 0.25	17 (94) | 1000	50,000	4.1–7 | *1.4−5*	*−2.6−6*|*−3.0−7*	10:18 | 6:16:37
2	781 | 1502 | 0.21	21 (104) | 1000	50,000	3.6–7 | *9.5−4*	*−6.3−6*|*5.1−3*	23:05 | 13:47:39
2	774 | 2002 | 0.18	29 (156) | 1000	50,000	7.3–7 | *2.1−3*	*−3.8−6*|*1.4−2*	49:53 | 23:20:28
3	395 | 253 | 0.49	11 (31) | 408	1487	9.8–7 | 9.7–7	*−1.4−6*|*1.3−7*	06 | 18
3	503 | 503 | 0.39	14 (61) | 877	7882	3.5–7 | 9.9–7	*−1.1−6*|*2.5−6*	1:46 | 7:18
3	512 | 803 | 0.33	15 (85) | 1000	10,579	7.7–7 | 9.9–7	*−1.3−6*|*4.2−7*	7:13 | 26:15
3	513 | 1003 | 0.31	16 (71) | 1000	14,025	2.5–7 | 9.9–7	*−7.6−7*|*4.2−6*	8:46 | 1:02:07
3	509 | 1503 | 0.27	19 (83) | 1000	23,328	8.6–7 | 9.9–7	*−4.5−6*|*7.2−6*	28:34 | 4:13:07
3	505 | 2003 | 0.24	19 (97) | 1000	50,000	9.5–7 | *3.3−4*	*−1.4−5*|*−3.7−4*	49:28 | 16:45:24

Numbers in italic indicate that they are greater than the stopping tolerance

Table 5 Same as Table 1 but for the sensor network localization problems (dual of (40))

d	m_E; m_I | n | R	iter.a it (subs)|itSCB	iter.b	η_{qsdp} a|b	η_{gap} a|b	Time a|b
2	452 ; 14,402 | 252 | 0.50	15 (119) | 603	50,000	6.4–7 | *1.9-6*	−9.1–7 | −5.1–8	1:50 | 24:28
2	548 ; 55,849 | 502 | 0.36	16 (180) | 1357	29,565	4.4–7 | 9.9–7	−2.2–7 | *−2.3−6*	11:28 | 1:09:24
2	633 ; 118,131 | 802 | 0.28	15 (226) | 2330	36,651	6.3–7 | 9.9–7	*−2.4−6*|*−5.4−6*	1:06:04 | 3:43:51
2	684 ; 160,157 | 1002 | 0.25	20 (265) | 3384	50,000	4.5–7 | *2.7−6*	*−1.7−6*|*2.2−6*	2:36:41 | 8:30:28
2	724 ; 201,375 | 1202 | 0.23	21 (487) | 3115	50,000	9.8–7 | *3.8−6*	*5.2−6*|*−2.8−6*	6:36:28 | 11:57:56
3	395 ; 16,412 | 253 | 0.49	12 (88) | 471	2897	3.1–7 | 9.8–7	−3.3–7 | −5.7–7	46 | 1:18
3	503 ; 53,512 | 503 | 0.39	14 (136) | 949	11,003	4.4–7 | 9.9–7	−9.9–7 | −3.1–7	8:23 | 20:26
3	512 ; 104,071 | 803 | 0.33	17 (145) | 1762	14,144	6.6–7 | 9.9–7	*−2.6−6* | 6.0–7	32:17 | 1:09:10
3	513 ; 139,719 | 1003 | 0.31	21 (198) | 2406	31,832	5.1–7 | 9.9–7	8.4–8 | 4.5–8	1:18:56 | 4:48:04
3	526 ; 180,236 | 1203 | 0.29	21 (250) | 2639	19,010	8.3–7 | 9.9–7	−3.8–8 | *1.1−5*	2:15:54 | 4:16:09

Numbers in italic indicate that they are greater than the stopping tolerance

Let $\mathcal{N}_i = \{p \mid (i, p) \in \mathcal{N}\}$ be the set of neighbors of the ith sensor. To further test our algorithm QSDPNAL, we also generate the following valid inequalities and add them to problem (38)

$$\|\hat{u}_i - \hat{u}_j\| \geq R, \ \forall (i, j) \in \widehat{\mathcal{N}},$$

where $\widehat{\mathcal{N}} = \bigcup_{i=1}^{n} \{(i, j) \mid j \in \mathcal{N}_p \backslash \mathcal{N}_i \text{ for some } p \in \mathcal{N}_i\}$. Then, we obtain the following QSDP relaxation model

$$\begin{array}{l} \min \ \frac{1}{2} \sum_{(i,j)\in\mathcal{N}} \left(g_{ij}^T X g_{ij} - d_{ij}^2\right)^2 - \lambda \langle I_{n+d} - aa^T, \ X\rangle \\ \text{s.t.} \ g_{ik}^T X g_{ik} = d_{ik}^2, \ (i,k) \in \mathcal{M}, \\ \quad\ g_{ij}^T X g_{ij} \geq R^2, \ (i,j) \in \widehat{\mathcal{N}}, \quad X \succeq 0. \end{array} \tag{40}$$

In Tables 4 and 5, we present the numerical results for QSDPNAL and QSDPNAL-Phase I in solving some instances of problem (39) and (40), respectively. Clearly, QSDPNAL outperforms the purely first-order algorithm QSDPNAL-Phase I by a significant margin. This superior numerical performance of QSDPNAL over QSDPNAL-Phase I again demonstrates the importance and necessity of our proposed two-phase framework.

7 Conclusions

We have designed a two-phase augmented Lagrangian based method, called QSDPNAL, for solving large scale convex quadratic semidefinite programming problems. The global and local convergence rate analysis of our algorithm is based on the classic results of proximal point algorithms [25,26], together with the recent advances in second order variational analysis of convex composite quadratic semidefinite programming [7]. By devising novel numerical linear algebra techniques, we overcome various challenging numerical difficulties encountered in the efficient implementation of QSDPNAL. Numerical experiments on various large scale QSDPs have demonstrated the efficiency and robustness of our proposed two-phase framework in obtaining accurate solutions. Specifically, for well-posed problems, our QSDPNAL-Phase I is already powerful enough and it is not absolutely necessary to execute QSDPNAL-Phase II. On the other hand, for more difficult problems, the purely first-order QSDPNAL-Phase I algorithm may stagnate because of extremely slow local convergence. In contrast, with the activation of QSDPNAL-Phase II which has second order information wisely incorporated, our QSDPNAL algorithm can still obtain highly accurate solutions efficiently.

Appendix

Along with the paper, we provide a MATLAB implementation of our algorithm. The software package has been issued the Digital Object Identifier https://doi.org/10.5281/zenodo.1206980. Here, as a short users' guide, we present some general description of the structure of our software.

Installation QSDPNAL is a MATLAB software package developed under MATLAB version 8.0 or above. It includes some C subroutines written to carry out certain operations for which MATLAB is not efficient at. These subroutines are called within MATLAB via the Mex interface. The user can simply follow the steps below to install QSDPNAL within MATLAB:

(**a**) unzip `QSDPNAL+v0.1.zip`;
(**b**) run MATLAB in the directory `QSDPNAL+v0.1`;

After that, to see whether you have installed QSDPNAL correctly, try the following steps:

```
>> startup
>> qsdpdemo
```

In the above, `startup.m` sets up the paths for QSDPNAL in MATLAB and `qsdpdemo.m` is a demo file illustrating how to solve a QSDP problem (31) in QSDPNAL.

Main function and data structure QSDPNAL is an extension of the solvers SDPNAL [34] and SDPNAL+ [33]. The internal implementation of QSDPNAL thus follows in part the data structures and design framework of the above two solvers. A casual user does not need to understand the internal implementation of QSDPNAL.

In the QSDPNAL solver, the main routine is `qsdpnal.m`, whose calling syntax is as follows:

```
[obj,Z,W,QW,S,yE,yI,X,runhist,info] =
qsdpnal(blk,Q,AEt,bE,C,AIt,bI,options)
```

Input arguments

- `blk`: a cell array describing the conic structure of the QSDP problem.
- `Q, AEt, bE, C, AIt, bI`: data of QSDP problem (31). If the linear map $\mathcal{A}_I$ is not present, one can set `AIt = [], bI = []`.
- `options`: a structure array of parameters.

Output arguments The names chosen for the output arguments explain their contents. The argument `X` is a solution to problem (31) up to the desired accuracy tolerance. The argument `info` is a structure array which records various performance measures of the solver. For example

`info.etaZ, info.etaI1, info.etaI2, info.etaI3`

correspond to the measures $\eta_Z, \eta_{I_1}, \eta_{I_2}$ and η_{I_3} defined in (35), respectively. The argument `runhist` is a structure array which records the history of various performance measures during the course of running `qsdpnal`. For example,

`runhist.primobj, runhist.dualobj, runhist.primfeasorg, runhist.dualfeasorg`

record the primal and dual objective values, primal and dual infeasibilities at each iteration, respectively.

Data structure The format of the input data in QSDPNAL is similar to those in SDPNAL [34] and SDPNAL+ [33]. For problem (31), we set

```
blk{1,1} = 's', blk{1,2} = n,
Q.QXfun = @(X) some function handle output QX,
AEt = [n̄ × m_E sparse], AIt = [n̄ × m_I sparse],
C = [n × n double or sparse],
```

where $\bar{n} = n(n+1)/2$ and the self-adjoint linear operator $\mathcal{Q}$ is stored as a function handle which takes any matrix $X \in \mathcal{S}^n$ as input and output the matrix $\mathcal{Q}X \in \mathcal{S}^n$. For the sake of computational efficiency, following the same approach used in SDPNAL [34] and SDPNAL+ [33], we store the linear map $\mathcal{A}_E$ in vectorized form as a single $\bar{n} \times m_E$ matrix `AEt`. The same procedure also applies to the linear operator $\mathcal{A}_I$, i.e., `AIt` is stored in the same format as `AEt`. The data of the simple nonempty closed convex polyhedral set $\mathcal{K}$ is encoded in the argument `options`. For example, if $\mathcal{K} = \{X \in \mathcal{S}^n \mid L \le X \le U\}$ with $L, U \in S^n$ being given matrices, we set

```
options.L = [matrix L, sparse or double],
 options.U = [matrix U,
sparse or double].
```

The default is `options.L = [], options.U = []`. If $\mathcal{K} = \{X \in \mathcal{S}^n \mid X \ge 0\}$, then one can simply set `options.nonnegative = 1`.

Apart from the information of $\mathcal{K}$, various other parameters used in our solver `qsdpnal.m` are set in the structure array `options`. We list here the important parameters which the user is likely to reset.

- `options.stoptol`: accuracy tolerance to terminate the algorithm, default is 10^{-6}.
- `options.maxiter`: maximum number of iterations allowed, default is 5000.
- `options.sGSstoptol`: accuracy tolerance to use for QSDPNAL-Phase I (`SCB_qsdp.m`) when generating a starting point for the algorithm in the second phase of `qsdpnal.m` (default = 10^{-4}).
- `options.sGSmaxiter`: maximum number of QSDPNAL-Phase I iterations allowed for generating a starting point (default = 1500).

Examples Next, we present two examples of using QSDPNAL for solving two QSDP problems in detail. The segment below illustrates how one can solve the instance `be250.2` of QSDP-BIQ problems.

```
>> G = biqread(['be250.2.sparse']);
>> [blk,AEt,C,bE,AIt,bI] = biq_sqsdp(G,3);
>> Q.QXfun = @(X) randQXfun(X,blk{1,2});
>> options.nonnegative = 1;
>> [obj,Z,W,QW,S,yE,yI,X,runhist,info] = ...
qsdpnal(blk,Q,AEt,bE,C,AIt,bI,options);
```

The next example is using `qsdpnal.m` to solve the instance `chr15c` of QSDP-QAP problems.

```
>> [AA,BB] = qapread(['chr15c.dat']);
>> [blk,AEt,C,bE] = qapAW(AA,BB,2);
>> AEt = AEt{1}; C = C{1}; AIt = []; bI = [];
>> options.nonnegative = 1;
>> Q.QXfun = @(X) randQXfun(X,blk{1,2});
>> [obj,Z,W,QW,S,yE,yI,X,runhist,info] = ...
qsdpnal(blk,Q,AEt,bE,C,AIt,bI,options);
```

In the above codes, `randQXfun` generates the self-adjoint positive semidefinite linear operator $\mathcal{Q}$ in (37) with randomly generated matrices $A, B \in \mathcal{S}^n_+$.

References

1. Alfakih, A.Y., Khandani, A., Wolkowicz, H.: Solving Euclidean distance matrix completion problems via semidefinite programming. Comput. Optim. Appl. **12**, 13–30 (1999)
2. Bauschke, H.H., Borwein, J.M., Li, W.: Strong conical hull intersection property, bounded linear regularity, Jameson's property (G), and error bounds in convex optimization. Math. Program. **86**, 135–160 (1999)
3. Biswas, P., Liang, T.C., Toh, K.-C., Wang, T.C., Ye, Y.: Semidefinite programming approaches for sensor network localization with noisy distance measurements. IEEE Trans. Autom. Sci. Eng. **3**, 360–371 (2006)
4. Bonnans, J.F., Shapiro, A.: Perturbation Analysis of Optimization Problems. Springer, New York (2000)
5. Burkard, R.E., Karisch, S.E., Rendl, F.: QAPLIB—a quadratic assignment problem library. J. Global Optim. **10**, 391–403 (1997)
6. Chen, L., Sun, D.F., Toh, K.-C.: An efficient inexact symmetric Gauss–Seidel based majorized ADMM for high-dimensional convex composite conic programming. Math. Program. **161**, 237–270 (2017)
7. Cui, Y., Sun, D.F., Toh, K.-C.: On the asymptotic superlinear convergence of the augmented Lagrangian method for semidefinite programming with multiple solutions, arXiv:1610.00875 (2016)
8. Clarke, F.: Optimization and Nonsmooth Analysis. Wiley, New York (1983)
9. Facchinei, F., Pang, J.-S.: Finite-Dimensional Variational Inequalities and Complementarity Problems. Springer, New York (2003)
10. Han, D., Sun, D., Zhang, L.: Linear rate convergence of the alternating direction method of multipliers for convex composite programming. Math. Oper. Res. (2017). https://doi.org/10.1287/moor.2017.0875
11. Higham, N.J.: Computing the nearest correlation matrix—a problem from finance. IMA J. Numer. Anal. **22**, 329–343 (2002)
12. Hiriart-Urruty, J.-B., Strodiot, J.-J., Nguyen, V.H.: Generalized Hessian matrix and second-order optimality conditions for problems with $C^{1,1}$ data. Appl. Math. Optim. **11**, 43–56 (1984)
13. Jiang, K., Sun, D.F., Toh, K.-C.: An inexact accelerated proximal gradient method for large scale linearly constrained convex SDP. SIAM J. Optim. **22**, 1042–1064 (2012)
14. Jiang, K., Sun, D.F., Toh, K.-C.: A partial proximal point algorithm for nuclear norm regularized matrix least squares problems. Math. Program. Comput. **6**, 281–325 (2014)

15. Krislock, N., Lang, J., Varah, J., Pai, D.K., Seidel, H.-P.: Local compliance estimation via positive semidefinite constrained least squares. IEEE Trans. Robot. **20**, 1007–1011 (2004)
16. Li, L., Toh, K.-C.: An inexact interior point method for l1-regularized sparse covariance selection. Math. Program. Comput. **2**, 291–315 (2010)
17. Li, X.D.: A Two-Phase Augmented Lagrangian Method for Convex Composite Quadratic Programming, PhD thesis, Department of Mathematics, National University of Singapore (2015)
18. Li, X.D., Sun, D.F., Toh, K.-C.: A Schur complement based semi-proximal ADMM for convex quadratic conic programming and extensions. Math. Program. **155**, 333–373 (2016)
19. Nie, J.W., Yuan, Y.X.: A predictor-corrector algorithm for QSDP combining Dikin-type and Newton centering steps. Ann. Oper. Res. **103**, 115–133 (2001)
20. Pang, J.-S., Sun, D.F., Sun, J.: Semismooth homeomorphisms and strong stability of semidefinite and Lorentz complementarity problems. Math. Oper. Res. **28**, 39–63 (2003)
21. Povh, J., Rendl, F.: Copositive and semidefinite relaxations of the quadratic assignment problem. Discrete Optim. **6**, 231–241 (2009)
22. Qi, H.D.: Local duality of nonlinear semidefinite programming. Math. Oper. Res. **34**, 124–141 (2009)
23. Qi, H.D., Sun, D.F.: A quadratically convergent Newton method for computing the nearest correlation matrix. SIAM J. Matrix Anal. Appl. **28**, 360–385 (2006)
24. Rockafellar, R.T.: Conjugate Duality and Optimization, CBMS-NSF Regional Conf. Ser. Appl. Math. vol. 16. SIAM, Philadelphia (1974)
25. Rockafellar, R.T.: Monotone operators and the proximal point algorithm. SIAM J. Control Optim. **14**, 877–898 (1976)
26. Rockafellar, R.T.: Augmented Lagrangians and applications of the proximal point algorithm in convex programming. Math. Oper. Res. **1**, 97–116 (1976)
27. Sun, D.F., Sun, J.: Semismooth matrix-valued functions. Math. Oper. Res. **27**, 150–169 (2002)
28. Sun, D.F., Toh, K.-C., Yang, L.: A convergent 3-block semiproximal alternating direction method of multipliers for conic programming with 4-type constraints. SIAM J. Optim. **25**, 882–915 (2015)
29. Sun, D.F., Toh, K.-C., Yang, L.: An efficient inexact ABCD method for least squares semidefinite programming. SIAM J. Optim. **26**, 1072–1100 (2016)
30. Sun, J., Zhang, S.: A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs. Eur. J. Oper. Res. **207**, 1210–1220 (2010)
31. Toh, K.-C.: An inexact primal-dual path following algorithm for convex quadratic SDP. Math. Program. **112**, 221–254 (2008)
32. Toh, K.-C., Tütüncü, R., Todd, M.: Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems. Pac. J. Optim. **3**, 135–164 (2007)
33. Yang, L., Sun, D.F., Toh, K.-C.: SDPNAL+: a majorized semismooth Newton-CG augmented Lagrangian method for semidefinite programming with nonnegative constraints. Math. Program. Comput. **7**, 331–366 (2015)
34. Zhao, X.Y.: A Semismooth Newton-CG Augmented Lagrangian Method for Large Scale Linear and Convex Quadratic SDPs, PhD thesis, Department of Mathematics, National University of Singapore (2009)
35. Zhao, X.Y., Sun, D.F., Toh, K.-C.: A Newton-CG augmented Lagrangian method for semidefinite programming. SIAM J. Optim. **20**, 1737–1765 (2010)