# A DUAL RIEMANNIAN ADMM ALGORITHM FOR LOW-RANK SDPS WITH UNIT DIAGONAL *

JIE WANG†, LIANGBING HU‡, AND BICAN XIA§

**Abstract.** This paper proposes a dual Riemannian alternating direction method of multipliers (ADMM) for solving low-rank semidefinite programs with unit diagonal constraints. We recast the ADMM subproblem as a Riemannian optimization problem over the oblique manifold by performing the Burer-Monteiro factorization. Global convergence of the algorithm is established assuming that the subproblem is solved to certain optimality. Numerical experiments demonstrate the excellent performance of the algorithm. It outperforms, by a significant margin, a few advanced SDP solvers (`MOSEK`, `COPT`, `SDPNAL+`, `ManiSDP`) in terms of accuracy, efficiency, and scalability on second-order SDP relaxations of dense and sparse binary quadratic programs.

**Key words.** semidefinite programming, low-rank solution, polynomial optimization, moment-SOS relaxation, Burer-Monteiro factorization, augmented Lagrangian method, manifold optimization, alternating direction method of multipliers

**AMS subject classifications.** Primary, 90C22; Secondary, 90C23,90C30

## 1. Introduction.

In this paper, we aim to solve the following semidefinite program (SDP) with unit diagonal constraints:

$$\text{(DSDP)} \qquad \begin{cases} \inf\limits_{y\in\mathbb{R}^m} & b^\intercal y \\ \text{s.t.} & S = \mathcal{A}^*(y) - C \succeq 0, \\ & \operatorname{diag}(S) = 1, \end{cases}$$

where $\mathcal{A}^*$ is the adjoint operator of a linear operator $\mathcal{A}: \mathbb{S}_n \to \mathbb{R}^m$ ($\mathbb{S}_n$ denotes the set of $n \times n$ symmetric matrices). We make the following assumptions on (DSDP):

ASSUMPTION 1.1. (DSDP) *admits a low-rank optimal solution* $S^\star$.

ASSUMPTION 1.2. *The linear operator* $\mathcal{A}\mathcal{A}^*$ *is invertible.*

(DSDP) arises as relaxations of a wide range of real or complex or noncommutative polynomial optimization problems, including binary quadratic programs (BQP) [32], the generalized orthogonal Procrustes problem [20], group synchronization [19, 21, 23], phase recovery [31], multiple-input multiple-output detection [22], designing unimodular codes [27], inference in graphical models [10], synchronization and community detection problems [4], the classical Ising problem [9], and the ground-state problem of quantum spin systems [33].

Despite being convex, solving general SDPs to high accuracy is a challenging problem and nowadays is still an active research area. For small/medium-scale SDPs, interior-point methods are usually accurate, efficient, and robust [3, 30]. Nevertheless, interior-point methods are not applicable to large-scale SDPs due to their huge

*Submitted to the editors DATE.

**Funding:** This work was supported by the National Key R&D Program of China under grant No. 2022YFA1005102 and the Natural Science Foundation of China under grant No. 12571333.

†State Key Laboratory of Mathematical Sciences, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China (wangjie212@amss.ac.cn)

‡Nanjing Research Institute of Electronics Technology, Nanjing, China (huliangbing2000@163.com)

§(School of Mathematical Sciences, Peking University, Beijing, China (xbc@math.pku.edu.cn)

memory occupation. First-order methods, e.g., alternating direction methods of multipliers (ADMM) [11, 35, 39], are more memory-friendly, but converge much slowly and typically only return a solution of low or medium accuracy. There are also first- and second-order hybrid algorithms based on augmented Lagrangian methods (ALM) that achieve a better balance between accuracy and scalability [37, 38]. On the other hand, SDPs from practice frequently possess certain structures, e.g., low-rank optimal solutions, data sparsity, and unit diagonal constraints. Those structures could be exploited to design more efficient and scalable algorithms [8, 12, 13, 14, 15, 18, 24, 29, 34, 36].

The structure of low-rank optimal solutions is usually exploited via the Burer-Monteiro factorization [6, 7]. A series of works have extended the approach to SDPs with distinguished constraints that define manifolds [16, 32, 34]. Particularly in [32], the first two authors proposed a Riemannian ALM algorithm for solving low-rank SDPs in the primal form which has superior performance on a variety of non-degenerate and degenerate SDPs with low-rank optimal solutions, including those from the moment-sum-of-squares (SOS) hierarchy of polynomial optimization problems. This work follows the research stream along [16, 32, 34] and presents an efficient algorithm for solving (DSDP) in the dual form that can be viewed as a dual version of the algorithm given in [32]. The motivation to consider the dual version stems from the fact that in many cases the rank of the linear operator $\mathcal{A}$ is much smaller when taking the dual SDP rather than the primal SDP. One important such case comes from the moment-SOS hierarchy of polynomial optimization problems: the SOS problem typically has significantly less linear constraints than the moment problem when relaxation orders are greater than one. Our contributions are summarized as follows.

• We propose a dual Riemannian ADMM algorithm for solving low-rank SDPs with unit diagonal constraints. First, we apply the ADMM framework to (DSDP). Then, to exploit the low-rank property, we perform the Burer-Monteiro factorization $S = YY^\intercal$ ($Y \in \mathbb{R}^{n\times p}, p \ll n$) to the ADMM subproblem. Due to the presence of unit diagonal constraints, we may recast the ADMM subproblem as a Riemannian optimization problem over the oblique manifold. To circumvent the non-convexity introduced by the Burer-Monteiro factorization, we employ a similar strategy as in [32] to escape from saddle points. Global convergence of the proposed algorithm is established assuming that the subproblem is solved to certain optimality.

• Assumption (1.2) implies a closed-form expression of $y$ in terms of $S$: $y = (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S + C)$. As a preprocessing step, we propose to eliminate $y$ from the objective of (DSDP) using $y = (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S + C)$. We empirically observe that it is more efficient to solve this modified version of (DSDP) with our algorithm.

• Numerical experiments demonstrate that the proposed algorithm outperforms, by a significant margin, several advanced SDP solvers (`MOSEK`, `COPT`, `SDPNAL+`) in terms of accuracy, efficiency, and scalability on second-order SDP relaxations of dense and sparse BQPs. Besides, it typically runs several times faster than the recent low-rank SDP solver `ManiSDP` [32]. The experiments also indicate an intriguing feature of our algorithm — the residue diving phenomenon, that is, the maximal KKT residue sharply decreases to far less than $10^{-8}$ at some ADMM iteration. Moreover, it is worth mentioning that our algorithm typically returns a solution of extremely high accuracy in a few tens of ADMM iterations. In comparison, traditional ADMM-based algorithms often require more than ten thousand iterations to arrive at a solution of high accuracy for the same type of SDPs [17].

The rest of the paper is organized as follows. In Section 2, we collect notation and some preliminary results. In Section 3, we present the ADMM framework for solving

(DSDP). In Section 4, we show how to solve the ADMM subproblem through the Burer-Monteiro factorization and provide computational details that are necessary for the Riemannian trust-region method. In Section 5, we present the whole algorithm and prove its global convergence. Numerical experiments are provided in Section 6. Conclusions are made in Section 7.

**2. Notation and preliminaries.** $\mathbb{R}$ (resp. $\mathbb{R}^+$, $\mathbb{N}$) denotes the set of real numbers (resp. positive real numbers, nonnegative integers). For a positive integer $n$, let $[n] \coloneqq \{1, 2, \ldots, n\}$. Let $\mathbb{S}_n$ (resp. $\mathbb{S}_n^+$) denote the set of (resp. positive semidefinite/PSD) symmetric matrices of size $n$. We use $\mathrm{Tr}(A)$ (resp. $A^\intercal$) to denote the trace (resp. transpose) of a matrix $A \in \mathbb{R}^{n\times m}$. For two matrices $A, B \in \mathbb{R}^{n\times m}$, the inner product is defined as $\langle A, B\rangle = \mathrm{Tr}(A^\intercal B)$. For $A \in \mathbb{R}^{n\times n}$, $\mathrm{diag}(A)$ denotes the diagonal of $A$, $\mathrm{Diag}(A)$ denotes the diagonal matrix with the same diagonal as $A$, and $\lambda_{\min}(A), \lambda_{\max}(A)$ denote the smallest, largest eigenvalues of $A$, respectively. For a vector $v$, let $|v|$ be the dimension of $v$, $\|v\|$ the 2-norm of $v$, and $\mathrm{Diag}(v)$ the diagonal matrix with $v$ being its diagonal. For a matrix $A$, $\|A\|$ denotes the Frobenius norm of $A$. For a function $f(x)$, we write $\nabla f(x)$ (resp. $\mathrm{grad}\, f(x)$) for the Euclidean (resp. Riemannian) gradient, and write $\nabla^2 f(x)[u]$ (resp. $\mathrm{Hess}\, f(x)[u]$) for the Euclidean (resp. Riemannian) Hessian acting on $u$. For a set $E$, $|E|$ denotes its cardinality.

Let $\mathcal{A}(X) \coloneqq (\langle A_i, X\rangle)_{i=1}^m$ for $A_i \in \mathbb{S}_n$ and $\mathcal{A}^* : \mathbb{R}^m \to \mathbb{S}_n$ be the adjoint operator of $\mathcal{A}$ defined as $\mathcal{A}^*(y) \coloneqq \sum_{i=1}^m y_i A_i$ for $y \in \mathbb{R}^m$. The linear operator $\mathcal{A}\mathcal{A}^* : \mathbb{R}^m \to \mathbb{R}^m$ is defined as $\mathcal{A}\mathcal{A}^*(y) = \mathcal{A}(\mathcal{A}^*(y))$. Let $\mathcal{B}(S) \coloneqq (\langle B_i, S\rangle)_{i=1}^n$, where $B_i \in \mathbb{S}_n$ is the matrix with the $i$-th diagonal element being one and all other elements being zeros, and $\mathcal{B}^* : \mathbb{R}^n \to \mathbb{S}_n$ be the adjoint operator of $\mathcal{B}$ defined as $\mathcal{B}^*(z) \coloneqq \sum_{i=1}^n z_i B_i = \mathrm{Diag}(z)$ for $z \in \mathbb{R}^n$.

The dual of (DSDP) reads as

$$\text{(PSDP)} \qquad \begin{cases} \sup\limits_{X\in\mathbb{S}_n^+, z\in\mathbb{R}^n} & \langle C, X + \mathcal{B}^*(z)\rangle + \sum_{i=1}^n z_i \\ \text{s.t.} & \mathcal{A}(X + \mathcal{B}^*(z)) = b. \end{cases}$$

Throughout the paper, we assume that strong duality holds for (PSDP)-(DSDP).

**3. The dual ADMM approach.** In this section, we introduce the dual ADMM framework for solving (DSDP). Let us define

$$\mathcal{M} \coloneqq \{S \in \mathbb{S}_n^+ \mid \mathrm{diag}(S) = 1\} = \{S \in \mathbb{S}_n^+ \mid \mathcal{B}(S) = 1\}. \tag{3.1}$$

Throughout the paper, we define the constant matrix $D$ as

$$D \coloneqq \mathcal{A}^*\left((\mathcal{A}\mathcal{A}^*)^{-1} b\right). \tag{3.2}$$

From $S = \mathcal{A}^*(y) - C$ together with Assumption 1.2, we have $y = (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S + C)$, and consequently,

$$b^\intercal y = b^\intercal (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S + C) = \left\langle \mathcal{A}^*\left((\mathcal{A}\mathcal{A}^*)^{-1} b\right), S + C\right\rangle = \langle D, S + C\rangle.$$

Therefore, we may reformulate (DSDP) as

$$\text{(DSDP')} \qquad \begin{cases} \inf\limits_{y\in\mathbb{R}^m, S\in\mathcal{M}} & \langle D, S + C\rangle \\ \text{s.t.} & S = \mathcal{A}^*(y) - C. \end{cases}$$

*Remark* 3.1. From now on, we work on the reformulation (DSDP') instead of (DSDP) as the former empirically exhibits better performance with the algorithm that will be developed in this paper.

We now employ the ADMM framework to solve (DSDP'). The augmented Lagrangian function associated with (DSDP') is defined by

$$L_\sigma(S, y, \widetilde{X}) = \langle D, S + C\rangle - \langle \widetilde{X}, \mathcal{A}^*(y) - S - C\rangle + \frac{\sigma}{2}\|\mathcal{A}^*(y) - S - C\|^2. \tag{3.3}$$

At the $k$-th ADMM iteration, one needs to solve the subproblems:

$$S^{k+1} := \arg\min_{S\in\mathcal{M}} L_{\sigma_k}\left(S, y^k, \widetilde{X}^k\right), \tag{3.4a}$$

$$y^{k+1} := \arg\min_{y\in\mathbb{R}^m} L_{\sigma_k}\left(S^{k+1}, y, \widetilde{X}^k\right). \tag{3.4b}$$

Then, $\widetilde{X}^k$ is updated as

$$\widetilde{X}^{k+1} := \widetilde{X}^k - \sigma_k\left(\mathcal{A}^*\left(y^{k+1}\right) - S^{k+1} - C\right). \tag{3.5}$$

LEMMA 3.2. *For any $k \geq 0$, $\mathcal{A}(\widetilde{X}^{k+1}) = 0$. Moreover, for any $k \geq 1$, the subproblem (3.4b) has a closed-form solution $y^{k+1} = (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S^{k+1} + C)$.*

*Proof.* The first-order optimality condition of (3.4b) reads as

$$-\mathcal{A}(\widetilde{X}^k) + \sigma_k\mathcal{A}(\mathcal{A}^*(y^{k+1}) - S^{k+1} - C) = 0, \tag{3.6}$$

from which we obtain

$$\mathcal{A}(\widetilde{X}^{k+1}) = \mathcal{A}(\widetilde{X}^k - \sigma_k(\mathcal{A}^*(y^{k+1}) - S^{k+1} - C)) = 0. \tag{3.7}$$

For $k \geq 1$, plugging $\mathcal{A}(\widetilde{X}^k) = 0$ into (3.6) yields

$$(\mathcal{A}\mathcal{A}^*)(y^{k+1}) = \mathcal{A}(S^{k+1} - C), \tag{3.8}$$

which gives $y^{k+1} = (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S^{k+1} + C)$. ∎

Let $\Phi_k(S) := L_{\sigma_k}(S, y^k, \widetilde{X}^k)$. The following lemma characterizes the optimality conditions of (3.4a).

LEMMA 3.3. *An element $S \in \mathcal{M}$ is a minimizer of (3.4a) if and only if*

$$X := \nabla\Phi_k(S) - \mathrm{Diag}(\nabla\Phi_k(S)S) \succeq 0, \tag{3.9a}$$

$$XS = 0, \tag{3.9b}$$

*where $\nabla\Phi_k(S) = \widetilde{X}^k - \sigma_k(\mathcal{A}^*(y^k) - S - C) + D$.*

*Proof.* Since (3.4a) is convex, $S \in \mathcal{M}$ is a minimizer if and only if the KKT conditions hold. The Lagrangian function of (3.4a) is

$$\Phi_k(S) - z^\intercal(\mathcal{B}(S) - 1) - \langle X, S\rangle, \tag{3.10}$$

where $z \in \mathbb{R}^n$ and $X \in \mathbb{S}^+_n$ are the dual variables. The KKT conditions of (3.4a) thus read as

$$S \in \mathcal{M}, \tag{3.11a}$$

$$\nabla\Phi_k(S) - \mathcal{B}^*(z) - X = 0, \tag{3.11b}$$

$$X \succeq 0, \tag{3.11c}$$

$$XS = 0. \tag{3.11d}$$

To get the closed-form expression of $z$, plugging $X = \nabla\Phi_k(S) - \mathcal{B}^*(z) = \nabla\Phi_k(S) - \mathrm{Diag}(z)$ into $XS = 0$ yields

$$\nabla\Phi_k(S)S = \mathrm{Diag}(z)S, \tag{3.12}$$

which gives $z = \mathrm{diag}(\nabla\Phi_k(S)S)$ by matching the diagonal elements of both sides. □

According to Lemma 3.3, $X^k$ is updated as

$$X^{k+1} := \widetilde{X}^{k+1} + D - \mathrm{Diag}\left(\left(\widetilde{X}^{k+1} + D\right) S^{k+1}\right). \tag{3.13}$$

We present the dual ADMM algorithm for solving (DSDP') in Algorithm 3.1.

**Algorithm 3.1** The dual ADMM algorithm for solving (DSDP')

**Input:** $\mathcal{A}, b, C, \sigma_0 > 0, \sigma_{\max} > \sigma_{\min} > 0$

$k \leftarrow 0$, $y^0 \leftarrow 0$;
**while** The stopping criterion is not fulfilled **do**
  Obtain $S^{k+1}$ by solving the ADMM subproblem (3.4a);
  $y^{k+1} \leftarrow (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S^{k+1} + C)$;
  $\widetilde{X}^{k+1} \leftarrow \widetilde{X}^k - \sigma_k\left(\mathcal{A}^*\left(y^{k+1}\right) - S^{k+1} - C\right)$;
  $z^{k+1} \leftarrow \mathrm{diag}\left(\left(\widetilde{X}^{k+1} + D\right) S^{k+1}\right)$;
  $X^{k+1} \leftarrow \widetilde{X}^{k+1} + D - \mathrm{Diag}\left(z^{k+1}\right)$;
  Determine $\sigma_{k+1} \in [\sigma_{\min}, \sigma_{\max}]$ according to certain policy;
  $k \leftarrow k + 1$;
**end while**
**return** $(S^k, y^k, X^k, z^k)$;

**4. Exploiting the manifold structure.** By Assumption 1.1, (DSDP) admits a low-rank optimal solution. To take into account the low-rank property, we apply the Burer-Monteiro factorization to $\Phi_k(S)$ and define $\Psi_k(Y) = \Phi_k(YY^\intercal)$ for $Y \in \mathbb{R}^{n\times p}$ ($p \ll n$). Doing so, the convex subproblem (3.4a) becomes the non-convex factorized subproblem:

$$\min_{Y\in\mathcal{N}_p} \Psi_k(Y) = L_{\sigma_k}(YY^\intercal, y^k, \widetilde{X}^k), \tag{Sub-$k$}$$

where

$$\mathcal{N}_p := \{Y \in \mathbb{R}^{n\times p} \mid YY^\intercal \in \mathcal{M}\} = \{Y \in \mathbb{R}^{n\times p} \mid \|Y(i,:)\| = 1,\ i = 1,\dots,n\}, \tag{4.1}$$

is the oblique manifold ($Y(i,:)$ stands for the $i$-th row of $Y$). We call $p$ the *factorization size*. Thus, it remains to solve (Sub-$k$) on the manifold $\mathcal{N}_p$, which could be done by off-the-shelf efficient Riemannian optimization methods. Here, we choose the Riemannian trust-region method to solve (Sub-$k$) due to its superior performance, as shown in [32]. We refer the reader to [1] for details on the Riemannian trust-region method.

We next calculate the Riemannian gradient and Riemannian Hessian of (Sub-$k$) that are necessary to perform optimization on the manifold $\mathcal{N}_p$ via the Riemannian trust-region method. First of all, we note that at a point $Y \in \mathcal{N}_p$, the tangent space of $\mathcal{N}_p$ is given by

$$T_Y\mathcal{N}_p = \{U \in \mathbb{R}^{n\times p} \mid \mathrm{diag}(UY^\intercal) = 0\}, \tag{4.2}$$

and the normal space to $\mathcal{N}_p$ is given by

$$N_Y\mathcal{N}_p = \{\mathrm{Diag}(u)Y \mid u \in \mathbb{R}^n\}. \tag{4.3}$$

LEMMA 4.1 ([32], Lemma 4.1). *Let $Y$ be a point on $\mathcal{N}_p$. The orthogonal projector $P_Y : \mathbb{R}^{n\times p} \to T_Y\mathcal{N}_p$ is given by*

$$P_Y(U) = U - \mathrm{Diag}(UY^\intercal)Y, \quad \text{for } U \in \mathbb{R}^{n\times p}. \tag{4.4}$$

PROPOSITION 4.2. *Consider the non-convex subproblem (Sub-$k$). Let $S = YY^\intercal$ and $X = \nabla\Phi_k(S) - \mathrm{Diag}(\nabla\Phi_k(S)S)$. Then the* Riemannian gradient *at $Y$ is given by*

$$\mathrm{grad}\,\Psi_k(Y) = 2XY. \tag{4.5}$$

*For $U \in T_Y\mathcal{N}_p$, let*

$$\widetilde{H} = \nabla^2\Psi_k(Y)[U] = 2\nabla\Phi_k(S)U + 2\sigma_k\left(Z - \mathcal{A}^*\left((\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(Z)\right)\right)Y,$$

*where $Z \coloneqq YU^\intercal + UY^\intercal$. Then the* Riemannian Hessian *is given by*

$$\mathrm{Hess}\,\Psi_k(Y)[U] = \widetilde{H} - \mathrm{Diag}(\widetilde{H}Y^\intercal)Y - 2\mathrm{Diag}(\nabla\Phi_k(S)S)U. \tag{4.6}$$

*Proof.* By (3) of [2], $\mathrm{grad}\,\Psi_k(Y) = P_Y(\nabla\Psi_k(Y)) = P_Y(2\nabla\Phi_k(S)Y)$. By Lemma 4.1, there exists $z \in \mathbb{R}^n$ such that

$$\mathrm{grad}\,\Psi_k(Y) = 2\nabla\Phi_k(S)Y - 2\mathrm{Diag}(z)Y.$$

Moreover, since $\mathrm{grad}\,\Psi_k(Y) \in T_Y\mathcal{N}_p$, we have

$$\mathrm{diag}\left(\mathrm{grad}\,\Psi_k(Y)Y^\intercal\right) = 2\,\mathrm{diag}\left((\nabla\Phi_k(S) - \mathrm{Diag}(z))S\right) = 0, \tag{4.7}$$

which gives $z = \mathrm{diag}(\nabla\Phi_k(S)S)$.

By (10) of [2], we have

$$\mathrm{Hess}\,\Psi_k(Y)[U] = P_Y(\nabla^2\Psi_k(Y)[U]) + \mathfrak{A}_Y(U, P_Y^\perp(\nabla\Psi_k(Y))), \tag{4.8}$$

where $\mathfrak{A}_Y$ is the Weingarten map at $Y$ and $P_Y^\perp = I - P_Y$ is the orthogonal projector at $Y$ to $N_Y\mathcal{N}_p$. Let $D_Y(\cdot)[U]$ be the directional derivative at $Y$ along $U$. Then,

$$\begin{aligned}
\mathfrak{A}_Y(U, P_Y^\perp(\nabla\Psi_k(Y))) &= \mathfrak{A}_Y(U, 2\mathrm{Diag}(z)Y) \\
&= -P_Y(D_Y(2\mathrm{Diag}(z)Y)[U]) \\
&= -2P_Y(\mathrm{Diag}(z)U) - 2P_Y(\mathrm{Diag}(D_Y(z)[U])Y) \\
&= -2\mathrm{Diag}(z)U,
\end{aligned}$$

where we have used the fact that $P_Y(\mathrm{Diag}(z)U) = \mathrm{Diag}(z)U$ and $P_Y(\mathrm{Diag}(u)Y) = 0$ for any $u \in \mathbb{R}^n$. (4.6) then follows. ∎

The global optimality of (Sub-$k$) is solely determined by the positive semidefiniteness of $X$.

PROPOSITION 4.3. *Let $Y \in \mathcal{N}_p$, $S = YY^\intercal$, and $X = \nabla\Phi_k(S) - \mathrm{Diag}(\nabla\Phi_k(S)S)$. Then a stationary point $Y$ of the non-convex factorized subproblem (Sub-$k$) is a global minimizer if and only if $X \succeq 0$.*

*Proof.* First, notice that $Y$ is a global minimizer of (Sub-$k$) if and only if $S$ is a minimizer of the convex subproblem (3.4a). The condition that $Y$ is a stationary point means $\operatorname{grad}\Psi_k(Y) = 2XY = 0$ implying $XS = 0$. The conclusion then follows from Lemma 3.3. ∎

We then obtain the following theorem as an immediate corollary of Proposition 4.3.

THEOREM 4.4. *A stationary point $Y \in \mathcal{N}_p$ of the non-convex factorized subproblem* (Sub-$k$) *provides a minimizer $S = YY^\intercal$ of* (DSDP') *if and only if $\mathcal{A}^*(y^k) = S + C$ and $X = \nabla\Phi_k(S) - \operatorname{Diag}(\nabla\Phi_k(S)S) \succeq 0$.*

As the subproblem (Sub-$k$) is non-convex, it is very important to escape from saddle points in order to solve (Sub-$k$) to certain optimality. We shall see that a second-order descent direction $U$ is readily available whenever $X = \nabla\Phi_k(S) - \operatorname{Diag}(\nabla\Phi_k(S)S) \not\succeq 0$.

THEOREM 4.5. *Suppose $X = \nabla\Phi_k(S) - \operatorname{Diag}(\nabla\Phi_k(S)S) \not\succeq 0$. Let $\delta \in \mathbb{N}$ be a positive number and let $V \in \mathbb{R}^{n\times\delta}$ be a matrix whose columns consist of eigenvectors corresponding to negative eigenvalues of $X$. Then $U \coloneqq [0_{n\times p}, V]$ is a second-order descent direction of* (Sub-$k$) *with $p \coloneqq p + \delta$ at the point $Y \coloneqq [Y, 0_{n\times\delta}]$, namely, $U$ satisfies*

$$\langle U, \operatorname{grad}\Psi_k(Y)\rangle = 0, \quad \langle U, \operatorname{Hess}\Psi_k(Y)[U]\rangle < 0. \tag{4.9}$$

*Proof.* By construction, we have $YU^\intercal = 0$ which implies $U \in T_Y\mathcal{N}_p$. Therefore,

$$\langle U, \operatorname{grad}\Psi_k(Y)\rangle = 2\operatorname{Tr}(U^\intercal XY) = 2\operatorname{Tr}(XYU^\intercal) = 0.$$

On the other hand, by (4.6) and Lemma 4.1, we have $\widetilde{H} = 2\nabla\Phi_k(S)U$ and

$$\begin{aligned}\operatorname{Hess}\Psi_k(Y)[U] &= \widetilde{H} - \operatorname{Diag}(\widetilde{H}Y^\intercal)Y - 2\operatorname{Diag}(\nabla\Phi_k(S)S)U\\ &= 2\nabla\Phi_k(S)U - 2\operatorname{Diag}(\nabla\Phi_k(S)S)U\\ &= 2XU.\end{aligned}$$

Thus,

$$\langle U, \operatorname{Hess}\Psi_k(Y)[U]\rangle = 2\operatorname{Tr}(U^\intercal XU) = 2\operatorname{Tr}(V^\intercal XV) < 0.$$ ∎

*Remark* 4.6. In analogy with Theorem 4.8 of [32], one can show that a rank deficient second-order critical point of (Sub-$k$) is a global minimizer. Therefore, by escaping from saddle points we necessarily reach a global minimizer of (Sub-$k$).

**5. The `ManiDSDP` algorithm.** Our dual Riemannian ADMM algorithm for solving (DSDP') is presented in Algorithm 5.1.

We make a few remarks on Algorithm 5.1.

- The initial factorization size $p_0$ can be set to 2. However, we have found that sometimes choosing $p_0 \sim O(\lceil\log(m)\rceil)$ would bring some speed-up; see [26].
- At Step 9, one could perform a partial eigenvalue decomposition when the full eigenvalue decomposition is too expensive.
- To minimize the cost of solving the ADMM subproblem (Sub-$k$), we follow the strategy described in [32, Section 5.1] to dynamically adjust the factorization size $p$;

**Algorithm 5.1** `ManiDSDP`

**Input:** $\mathcal{A}, b, C, \sigma_0 > 0, \sigma_{\max} > \sigma_{\min} > 0$, $p_0 \in \mathbb{N}$, $\gamma > 1, \tau_2 \geq \tau_1 > 0$

$k \leftarrow 0$, $y^0 \leftarrow 0$, $p \leftarrow p_0$, $Y^0 \leftarrow 0_{n\times p}$, $U \leftarrow 0_{n\times p}$;
**while** The stopping criterion is not fulfilled **do**
  Solve the ADMM subproblem (Sub-$k$) inexactly using the Riemannian trust-region method with $U$ as a descent direction, obtaining an approximate minimizer $Y^{k+1}$;
  $S^{k+1} \leftarrow Y^{k+1}(Y^{k+1})^\intercal$;
  $y^{k+1} \leftarrow (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}(S^{k+1} + C)$;
  $\widetilde{X}^{k+1} \leftarrow \widetilde{X}^k - \sigma_k\left(\mathcal{A}^*\left(y^{k+1}\right) - S^{k+1} - C\right)$;
  $z^{k+1} \leftarrow \operatorname{diag}\left(\left(\widetilde{X}^{k+1} + D\right) S^{k+1}\right)$;
  $X^{k+1} \leftarrow \widetilde{X}^{k+1} + D - \operatorname{Diag}\left(z^{k+1}\right)$;
  Compute a descent direction $U$ from $X^{k+1}$ according to Theorem 4.5;
  Update $p$ following the strategy described in [32, Section 5.1];
  Update $\sigma_k$ according to the rules (5.1);
  $k \leftarrow k + 1$;
**end while**
**return** $(S^k, y^k, X^k, z^k)$;

- The following self-adaptive strategy is adopted to update the penalty factor $\sigma$:

(5.1)
$$\sigma_{k+1} = \begin{cases} \max\{\sigma_k/\gamma, \sigma_{\min}\}, & \text{if } \|R^{k+1}\|/(1+\|b\|) < \tau_1 \|\operatorname{grad}\Psi_k(Y^{k+1})\|, \\ \min\{\gamma\sigma_k, \sigma_{\max}\}, & \text{if } \|R^{k+1}\|/(1+\|b\|) > \tau_2 \|\operatorname{grad}\Psi_k(Y^{k+1})\|, \\ \sigma_k, & \text{otherwise,} \end{cases}$$

  where $R^{k+1} = \mathcal{A}^*(y^{k+1}) - S^{k+1} - C$ and $\gamma > 1, \tau_2 \geq \tau_1 > 0, \sigma_{\max} > \sigma_{\min} > 0$ are constants.

We now establish the global convergence of `ManiDSDP` assuming that the subproblem (Sub-$k$) is solved to certain optimality.

THEOREM 5.1. *Let $\{\varepsilon_k\}_{k\in\mathbb{N}}, \{\tau_k\}_{k\in\mathbb{N}} \subseteq \mathbb{R}^+$ satisfy $\sum_{i=0}^{\infty}\varepsilon_k < \infty$ and $\sum_{i=0}^{\infty}\tau_k < \infty$. Suppose that we use the following stopping criterion for the subproblem (Sub-$k$):*

$$\|X^{k+1}Y^{k+1}\| \leq \varepsilon_k \text{ and } \lambda_{\min}(X^{k+1}) \geq -\tau_k. \tag{5.2}$$

*Let $(\hat{S}, \hat{y}, \hat{X}, \hat{z})$ be a limit point of $\{(S^k, y^k, X^k, z^k)\}_{k\geq 1}$. Then $(\hat{S}, \hat{y}, \hat{X}, \hat{z})$ is a KKT point of (DSDP).*

Let us begin by proving two lemmas.

LEMMA 5.2. *Suppose that $Y^{k+1}$ is an approximate minimizer of the subproblem (Sub-$k$) so that the criterion (5.2) is fulfilled. Let $S^{k+1} := Y^{k+1}(Y^{k+1})^\intercal$ and $(S^\star, y^\star, X^\star, z^\star)$ be a KKT point of (DSDP'). Let $\widetilde{X}^\star = X^\star - D + \mathcal{B}^*(z^\star)$ and $R^{k+1} = \mathcal{A}^*(y^{k+1}) - S^{k+1} - C$. Then*

$$\langle \widetilde{X}^k - \widetilde{X}^\star, R^{k+1}\rangle \geq \sigma_k\|R^{k+1}\|^2 - n\tau_k - \sqrt{n}\varepsilon_k. \tag{5.3}$$

*Proof.* By the stopping criterion (5.2), we have

$$\langle X^{k+1}, S^\star - S^{k+1}\rangle = \langle X^{k+1}, S^\star\rangle - \langle X^{k+1}Y^{k+1}, Y^{k+1}\rangle \geq -n\tau_k - \sqrt{n}\varepsilon_k. \tag{5.4}$$

From the definitions $S^\star = \mathcal{A}^*(y^\star) - C$ and $S^{k+1} = \mathcal{A}^*(y^{k+1}) - R^{k+1} - C$, we obtain

$$S^\star - S^{k+1} = \mathcal{A}^*(y^\star - y^{k+1}) + R^{k+1}. \tag{5.5}$$

It follows that

$$\begin{aligned}\langle \widetilde{X}^{k+1}, S^\star - S^{k+1}\rangle &= \langle \widetilde{X}^k - \sigma_k R^{k+1}, \mathcal{A}^*(y^\star - y^{k+1}) + R^{k+1}\rangle \\ &= \langle \mathcal{A}(\widetilde{X}^{k+1}), y^\star - y^{k+1}\rangle + \langle \widetilde{X}^k, R^{k+1}\rangle - \sigma_k\|R^{k+1}\|^2 \\ &= \langle \widetilde{X}^k, R^{k+1}\rangle - \sigma_k\|R^{k+1}\|^2. \end{aligned} \tag{5.6}$$

Moreover,

$$\begin{aligned}\langle D, S^\star - S^{k+1}\rangle &= \langle D, \mathcal{A}^*(y^\star - y^{k+1}) + R^{k+1}\rangle \\ &= \langle \mathcal{A}(D), y^\star - y^{k+1}\rangle + \langle D, R^{k+1}\rangle \\ &= \langle \mathcal{A}(X^\star + \mathcal{B}^*(z^\star)), y^\star - y^{k+1}\rangle + \langle D, R^{k+1}\rangle \\ &= \langle X^\star + \mathcal{B}^*(z^\star), \mathcal{A}^*(y^\star) - \mathcal{A}^*(y^{k+1})\rangle + \langle D, R^{k+1}\rangle \\ &= \langle X^\star + \mathcal{B}^*(z^\star), (S^\star + C) - (S^{k+1} + C + R^{k+1})\rangle + \langle D, R^{k+1}\rangle \\ &= \langle X^\star, S^\star\rangle - \langle X^\star, S^{k+1}\rangle - \langle X^\star + \mathcal{B}^*(z^\star), R^{k+1}\rangle + \langle D, R^{k+1}\rangle \\ &\le \langle D - X^\star - \mathcal{B}^*(z^\star), R^{k+1}\rangle \\ &= -\langle \widetilde{X}^\star, R^{k+1}\rangle, \end{aligned} \tag{5.7}$$

where we have used $\mathcal{A}(D) = b = \mathcal{A}(X^\star + \mathcal{B}^*(z^\star))$, $\langle X^\star, S^\star\rangle = 0$, and $\langle X^\star, S^{k+1}\rangle \ge 0$. Besides,

$$\langle -\mathcal{B}^*(z^{k+1}), S^\star - S^{k+1}\rangle = \langle -z^{k+1}, \mathcal{B}(S^\star) - \mathcal{B}(S^{k+1})\rangle = 0. \tag{5.8}$$

Substituting $\widetilde{X}^{k+1} + D - \mathcal{B}^*(z^{k+1})$ for $X^{k+1}$ in (5.4) and using (5.6)–(5.8), we obtain

$$\begin{aligned}\langle X^{k+1}, S^\star - S^{k+1}\rangle &= \langle \widetilde{X}^{k+1} + D - \mathcal{B}^*(z^{k+1}), S^\star - S^{k+1}\rangle \\ &\le \langle \widetilde{X}^k - \widetilde{X}^\star, R^{k+1}\rangle - \sigma_k\|R^{k+1}\|^2. \end{aligned} \tag{5.9}$$

Thus, combining (5.4) with (5.9) yields (5.3). ∎

LEMMA 5.3. *It holds that* $\lim_{k\to\infty}\|R^k\| = 0$.

*Proof.* Let $(S^\star, y^\star, X^\star, z^\star)$ be a KKT point of (DSDP'). For all $k \ge 1$, by invoking Lemma 5.2, we have

$$\begin{aligned}\|\widetilde{X}^{k+1} - \widetilde{X}^\star\|^2 &= \|\widetilde{X}^k - \widetilde{X}^\star\|^2 - 2\sigma_k\langle \widetilde{X}^k - \widetilde{X}^\star, R^{k+1}\rangle + \sigma_k^2\|R^{k+1}\|^2 \\ &\le \|\widetilde{X}^k - \widetilde{X}^\star\|^2 - \sigma_k^2\|R^{k+1}\|^2 + 2\sigma_k\left(n\tau_k + \sqrt{n}\varepsilon_k\right). \end{aligned}$$

For an arbitrary $N \ge 1$, summing the above inequality for $k = 1, \ldots, N$ yields

$$\|\widetilde{X}^{N+1} - \widetilde{X}^\star\|^2 \le \|\widetilde{X}^1 - \widetilde{X}^\star\|^2 - \sigma_{\min}^2\sum_{k=1}^N\|R^{k+1}\|^2 + 2\sigma_{\max}\sum_{k=1}^N\left(n\tau_k + \sqrt{n}\varepsilon_k\right).$$

It follows

$$\sigma_{\min}^2\sum_{k=1}^N\|R^{k+1}\|^2 \le \|\widetilde{X}^1 - \widetilde{X}^\star\|^2 + 2\sigma_{\max}\sum_{k=1}^N\left(n\tau_k + \sqrt{n}\varepsilon_k\right). \tag{5.10}$$

Now, because the right-hand side of (5.10) is bounded and $N$ is arbitrary, we see that $\|R^{k+1}\|$ must converge to 0. ∎

*Proof of Theorem 5.1.* Because $\lim_{k\to\infty}\tau_k = 0$, we have $\hat{X} \succeq 0$. Moreover, noting $\langle X^k, S^k\rangle = |\langle X^k Y^k, Y^k\rangle| \le \varepsilon_{k-1}\cdot\|Y^k\|$, since $\lim_{k\to\infty}\varepsilon_k = 0$, we have $\langle \hat{X}, \hat{S}\rangle = 0$. By Lemma 5.3, $\mathcal{A}^*(\hat{y}) - \hat{S} - C = 0$. In addition, by Lemma 3.2, $\mathcal{A}(\hat{X} + \mathcal{B}^*(\hat{z})) = \mathcal{A}(\hat{\tilde{X}} + D) = b$ where $\hat{\tilde{X}} := \hat{X} + \mathcal{B}^*(\hat{z}) - D$. Therefore, $(\hat{S}, \hat{y}, \hat{X}, \hat{z})$ is a KKT point of (DSDP).

*Remark* 5.4. Algorithm 5.1 can be extended to handle SDPs with multi-blocks which often arise as relaxations of sparse polynomial optimization problems.

**6. Numerical experiments.** In this section, we benchmark the performance of the `ManiDSDP` algorithm which was implemented in the low-rank SDP solver `ManiSDP`[1]. `Manopt 7.0` [5] is employed to solve the ADMM subproblem (Sub-$k$). When presenting the results, the column labelled by 'time' records running time in seconds. Moreover, the symbol '-' indicates that the SDP solver runs out of memory, the symbol '$*$' indicates that running time exceeds 10000 seconds, and the symbol '$**$' indicates that the solver encounters some numerical issue.

**Hardware.** All numerical experiments were performed on a desktop computer with Intel(R) Core(TM) i9-10900 CPU@2.80GHz and 64G RAM.

**Baseline Solvers.** We compare the performance of `ManiDSDP` with that of four advanced SDP solvers: `MOSEK 11.0` [25], `COPT 7.2.3`, `SDPNAL+` [28], `ManiSDP` [32]. We explain why to choose these four baseline solvers: `MOSEK` and `COPT` are chosen as they are popular commercial solvers; `SDPNAL+` is chosen as it is designed to solve large-scale SDPs; `ManiSDP` is chosen as it is a representative solver that exploits the low-rank property via the Burer-Monteiro factorization.

**The Stopping Criterion.** To measure the feasibility and optimality of an approximate solution $(X, y, S) \in \mathbb{S}^+_n \times \mathbb{R}^m \times \mathbb{S}^+_n$, we define the following KKT residues:

$$\eta_p = \frac{\|\mathcal{A}^*(y) - S - C\|}{1 + \|C\|}, \tag{6.1}$$

$$\eta_d = \frac{\max\{0, -\lambda_{\min}(X)\}}{1 + |\lambda_{\max}(X)|}, \tag{6.2}$$

$$\eta_g = \frac{|\langle C, X\rangle - b^\intercal y|}{1 + |\langle C, X\rangle| + |b^\intercal y|}. \tag{6.3}$$

Given a tolerance tol $> 0$, the SDP solver terminates when $\eta_{\max} := \max\{\eta_p, \eta_d, \eta_g\} \le$ tol, and we set tol = 1e-8 for all our experiments.

**6.1. Dense BQPs.** Let us consider the dense BQP given by

$$\text{(BQP)} \qquad \begin{cases} \inf\limits_{\mathbf{x}\in\mathbb{R}^q} & \mathbf{x}^\intercal Q\mathbf{x} + \mathbf{c}^\intercal\mathbf{x} \\ \text{s.t.} & x_i^2 = 1, \quad i = 1, \ldots, q, \end{cases}$$

where $Q \in \mathbb{S}_q$ and $\mathbf{c} \in \mathbb{R}^q$. (BQP) covers many combinatorial optimization problems (e.g., the Max-Cut problem), the $\mathbb{Z}_2$ synchronization problem in signal processing, and the classical Ising problem in physics. Let

$$v(\mathbf{x}) := [1, x_1, \ldots, x_q, x_1x_2, x_1x_3, \ldots, x_{q-1}x_q]^\intercal$$

[1] `ManiSDP` is freely available at https://github.com/wangjie212/ManiSDP-matlab.

be the vector of monomials in $\mathbf{x}$ up to degree two (excluding squares $x_i^2, i = 1, \ldots, q$). The second-order SOS relaxation of (BQP) is the following SDP:

$$\text{(6.4)} \qquad \begin{cases} \sup\limits_{X,\lambda} & \lambda \\ \text{s.t.} & \mathbf{x}^\intercal Q\mathbf{x} + \mathbf{c}^\intercal \mathbf{x} - \lambda \equiv v(\mathbf{x})^\intercal X v(\mathbf{x}), \mod \mathcal{I}, \\ & X \succeq 0, \end{cases}$$

where $\mathcal{I}$ is the ideal generated by $\{x_i^2 - 1\}_{i=1}^q$ in the polynomial ring $\mathbb{R}[\mathbf{x}]$. For each linear constraint of (6.4) indexed by a monomial

$$a \in \{1\} \cup \{x_i\}_{i=1}^q \cup \{x_i x_j\}_{1 \le i < j \le q} \cup \{x_i x_j x_k\}_{1 \le i < j < k \le q} \cup \{x_i x_j x_k x_l\}_{1 \le i < j < k < l \le q},$$

we associate it with a dual variable $y_a \in \mathbb{R}$. Then the dual of (6.4) fits exactly into (DSDP) with $C = 0$.

For each $q \in \{10, 20, 30, 40, 50, 60, 70, 80\}$, we generate three random instances of (BQP) by taking $Q \in \mathbb{S}_q$ with $Q_{ij} \sim \mathcal{N}(0, 1)$ and $\mathbf{c} \in \mathbb{R}^q$ with $c_i \sim \mathcal{N}(0, 1)$. For each instance, we solve the SDP relaxation (6.4). The results are presented in Table 1 from which we could draw the following conclusions: (i) `MOSEK` is very efficient for small problems (e.g. $q = 10$), but the running time significantly grows as $q$ increases. Notice that `MOSEK` runs out of memory when $q \ge 40$. (ii) `COPT` encounters some numerical issue on one instance with $q = 30$, and is the slowest among all solvers especially when $q$ becomes large. (iii) Both `ManiSDP` and `ManiDSDP` can solve all instances to extremely high accuracy, while `ManiDSDP` is the most efficient. (iv) `SDPNAL+` can solve the instances to medium/high accuracy, but the running time is pretty significant.

**6.2. Sparse BQPs.** We now conduct numerical experiments on sparse BQPs which are constructed as follows. Suppose that $t$ and $q$ are given positive integer numbers. For $k \in \{1, 2 \ldots, t\}$, let $\mathbf{x}_k = \{x_{(q-2)(k-1)+1}, \ldots, x_{(q-2)k+2}\}$, and let $\mathbf{x} = \cup_{k=1}^t \mathbf{x}_k = \{x_1, \ldots, x_{(q-2)t+2}\}$. Let us consider the following sparse BQP:

$$\text{(BQP-sparse)} \qquad \begin{cases} \inf\limits_{\mathbf{x} \in \mathbb{R}^{(q-2)t+2}} & \sum_{k=1}^t (\mathbf{x}_k^\intercal Q_k \mathbf{x}_k + \mathbf{c}_k^\intercal \mathbf{x}_k) \\ \text{s.t.} & x_i^2 = 1, \quad i = 1, 2, \ldots, (q-2)t + 2, \end{cases}$$

where $Q_k \in \mathbb{S}_q$ and $\mathbf{c}_k \in \mathbb{R}^q$, $k = 1, 2, \ldots, t$. The second-order sparse SOS relaxation of (BQP-sparse) is the following multi-block SDP:

$$\text{(6.5)} \qquad \begin{cases} \sup\limits_{X_k,\lambda} & \lambda \\ \text{s.t.} & \sum_{k=1}^t (\mathbf{x}_k^\intercal Q_k \mathbf{x}_k + \mathbf{c}_k^\intercal \mathbf{x}_k) - \lambda \equiv \sum_{k=1}^t v(\mathbf{x}_k)^\intercal X_k v(\mathbf{x}_k), \mod \mathcal{I}, \\ & X_k \succeq 0, \quad k = 1, 2, \ldots, t, \end{cases}$$

where $\mathcal{I}$ is the ideal generated by $\{x_i^2 - 1\}_{i=1}^{(q-2)t+2}$. The dual of (6.5) fits into a multi-block version of (DSDP) with $C = 0$:

$$\text{(DSDP-sparse)} \qquad \begin{cases} \inf\limits_{y \in \mathbb{R}^m} & b^\intercal y \\ \text{s.t.} & S = \mathcal{A}^*(y) - C \in \mathbb{S}^+_{|v(\mathbf{x}_1)|} \times \cdots \times \mathbb{S}^+_{|v(\mathbf{x}_t)|}, \\ & \operatorname{diag}(S) = 1. \end{cases}$$

First, we fix the parameter $q = 20$ and vary the parameter $t$: For each $t \in \{20, 40, 60, 80, 100, 120\}$, we generate three random instances of (BQP-sparse) by taking $Q_k \in \mathbb{S}_q$ with $[Q_k]_{ij} \sim \mathcal{N}(0, 1)$ and $\mathbf{c}_k \in \mathbb{R}^q$ with $c_{k,i} \sim \mathcal{N}(0, 1)$, $k = 1, 2, \ldots, t$.

TABLE 1
*Results for dense BQPs.*

| $q$ | trial | MOSEK | | COPT | | SDPNAL+ | | ManiSDP | | ManiDSDP | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time |
| 10 | #1 | 2.8e-09 | **0.02** | 4.6e-10 | 0.08 | 1.5e-09 | 0.10 | 7.3e-09 | 0.07 | 8.4e-15 | 0.07 |
| | #2 | 7.4e-10 | **0.02** | 2.3e-09 | 0.09 | 4.4e-10 | 0.08 | 4.1e-15 | 0.06 | 8.8e-15 | 0.06 |
| | #3 | 3.0e-11 | **0.02** | 1.8e-09 | 0.08 | 6.0e-09 | 0.08 | 3.7e-15 | 0.07 | 8.9e-15 | 0.07 |
| 20 | #1 | 3.1e-13 | 3.49 | 1.8e-09 | 5.71 | 3.8e-10 | 1.21 | 1.6e-14 | 0.25 | 4.9e-14 | **0.20** |
| | #2 | 1.0e-09 | 3.64 | 4.7e-11 | 6.22 | 3.1e-09 | 1.09 | 1.3e-14 | 0.33 | 4.8e-14 | **0.24** |
| | #3 | 1.6e-10 | 3.44 | 2.6e-10 | 5.79 | 2.0e-09 | 1.46 | 1.4e-14 | 0.42 | 4.8e-14 | **0.28** |
| 30 | #1 | 1.3e-12 | 439 | ** | ** | 9.4e-10 | 6.74 | 2.8e-14 | 3.14 | 1.1e-13 | **1.64** |
| | #2 | 5.2e-12 | 426 | 7.1e-08 | 496 | 7.2e-09 | 14.2 | 2.9e-14 | 1.67 | 1.1e-13 | **1.15** |
| | #3 | 1.8e-10 | 491 | 1.4e-07 | 876 | 3.4e-10 | 20.9 | 3.0e-14 | 2.57 | 1.1e-13 | **1.52** |
| 40 | #1 | - | - | 4.1e-08 | 881 | 2.4e-09 | 79.2 | 4.9e-14 | 7.93 | 2.3e-13 | **3.36** |
| | #2 | - | - | 7.7e-08 | 336 | 4.1e-10 | 92.6 | 4.6e-14 | 4.83 | 2.4e-13 | **2.01** |
| | #3 | - | - | 2.8e-08 | 437 | 5.0e-10 | 65.6 | 4.8e-14 | 7.71 | 2.3e-13 | **2.71** |
| 50 | #1 | - | - | * | * | 6.7e-08 | 558 | 7.0e-14 | 25.9 | 4.3e-13 | **11.1** |
| | #2 | - | - | * | * | 6.1e-11 | 480 | 2.9e-13 | 34.2 | 4.3e-13 | **14.3** |
| | #3 | - | - | * | * | 3.8e-09 | 607 | 6.8e-14 | 58.1 | 4.3e-13 | **19.6** |
| 60 | #1 | - | - | * | * | 2.8e-07 | 1832 | 9.6e-14 | 80.6 | 7.5e-13 | **58.6** |
| | #2 | - | - | * | * | 9.1e-05 | 7464 | 9.7e-14 | 597 | 7.5e-13 | **234** |
| | #3 | - | - | * | * | 1.6e-07 | 1898 | 9.6e-14 | 146 | 7.5e-13 | **53.7** |
| 70 | #1 | - | - | * | * | * | * | 1.4e-13 | 986 | 7.7e-13 | **366** |
| | #2 | - | - | * | * | 9.8e-07 | 6934 | 1.4e-13 | 272 | 7.7e-13 | **142** |
| | #3 | - | - | * | * | * | * | 1.2e-13 | 421 | 7.7e-13 | **172** |
| 80 | #1 | - | - | * | * | * | * | 1.6e-13 | 1037 | 1.3e-12 | **492** |
| | #2 | - | - | * | * | * | * | 1.7e-13 | 1844 | 2.5e-11 | **656** |
| | #3 | - | - | * | * | * | * | 1.6e-13 | 2145 | 3.1e-12 | **1181** |

For each instance, we solve the sparse SDP relaxation (DSDP-sparse). The results are presented in Table 2. We could make the following observations from the table. (i) `COPT` is much slower than the other solvers, especially when $t \geq 40$. (ii) `MOSEK`, `ManiSDP`, and `ManiDSDP` can solve all instances to very high accuracy, while `ManiDSDP` is the most efficient. (iii) `SDPNAL+` can also solve all instances, but it takes significantly more time.

Next, we fix the parameter $t = 20$ and vary the parameter $q$: For each $q \in \{10, 20, 30, 40\}$, we generate three random instances of (BQP-sparse) by taking $Q_k \in \mathbb{S}_q$ with $[Q_k]_{ij} \sim \mathcal{N}(0,1)$ and $\mathbf{c}_k \in \mathbb{R}^q$ with $c_{k,i} \sim \mathcal{N}(0,1)$, $k = 1, 2, \ldots, t$. For each instance, we solve the sparse SDP relaxation (DSDP-sparse). The results are presented in Table 3. The following conclusions can be drawn from Table 3. (i) `MOSEK` is very efficient for small problems (say, $q = 10$), and becomes slower as $q$ grows. It runs out of memory for $q \geq 30$. (ii) Both `COPT` and `SDPNAL+` are much slower than `ManiSDP` and `ManiDSDP` for $q \geq 20$. In particular, `COPT` encounters some numerical issue when $q = 30$. (iii) `ManiSDP` can typically solve the instances to extremely high accuracy but fails on one instance with $q = 40$. (iv) `ManiDSDP` not only can solve all instances to very high accuracy, but is also the most efficient and robust.

TABLE 2
*Results for sparse BQPs ($q = 20$).*

| $t$ | trial | MOSEK | | COPT | | SDPNAL+ | | ManiSDP | | ManiDSDP | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time |
| 20 | #1 | 1.4e-13 | 151 | 8.2e-10 | 313 | 5.5e-10 | 226 | 1.9e-17 | 26.0 | 1.3e-12 | **12.5** |
| | #2 | 6.3e-14 | 146 | 8.6e-10 | 296 | 1.6e-08 | 143 | 4.7e-14 | 20.4 | 1.3e-12 | **6.97** |
| | #3 | 1.2e-13 | 160 | 1.8e-09 | 298 | 6.7e-08 | 275 | 1.4e-17 | 34.3 | 1.3e-12 | **11.4** |
| 40 | #1 | 8.1e-14 | 283 | * | * | 3.5e-09 | 622 | 4.1e-14 | 77.5 | 7.0e-12 | **51.0** |
| | #2 | 8.2e-14 | 340 | * | * | 4.2e-08 | 1072 | 1.9e-17 | 130 | 7.0e-12 | **45.1** |
| | #3 | 1.2e-13 | 289 | * | * | 7.1e-08 | 1206 | 1.9e-17 | 237 | 7.0e-12 | **49.2** |
| 60 | #1 | 1.4e-13 | 400 | * | * | 4.9e-10 | 2206 | 5.7e-13 | 79.7 | 1.5e-11 | **67.9** |
| | #2 | 2.2e-13 | 448 | * | * | 4.5e-07 | 3322 | 3.9e-14 | 313 | 1.5e-11 | **80.3** |
| | #3 | 1.5e-13 | 507 | * | * | 6.7e-10 | 1449 | 9.8e-14 | 167 | 1.5e-11 | **154** |
| 80 | #1 | 1.7e-12 | 651 | * | * | 7.2e-10 | 2933 | 9.3e-17 | 824 | 1.6e-11 | **304** |
| | #2 | 2.4e-13 | 718 | * | * | 2.7e-09 | 4542 | 9.3e-17 | 342 | 1.6e-11 | **119** |
| | #3 | 1.0e-13 | 719 | * | * | 2.0e-09 | 3195 | 1.8e-16 | 153 | 1.6e-11 | **144** |
| 100 | #1 | 2.3e-12 | 633 | * | * | 3.4e-09 | 3581 | 7.4e-17 | 705 | 2.5e-11 | **287** |
| | #2 | 5.0e-13 | 866 | * | * | 1.9e-08 | 6811 | 1.5e-16 | 1595 | 2.5e-11 | **80.9** |
| | #3 | 5.9e-13 | 956 | * | * | 1.6e-09 | 3249 | 7.0e-14 | 231 | 2.5e-11 | **131** |
| 120 | #1 | 4.4e-13 | 1269 | * | * | 1.8e-10 | 6931 | 1.2e-16 | 1175 | 3.4e-11 | **573** |
| | #2 | 2.5e-13 | 1039 | * | * | 8.6e-11 | 4942 | 1.1e-13 | 962 | 3.4e-11 | **135** |
| | #3 | 8.4e-13 | 910 | * | * | 5.0e-09 | 4863 | 6.1e-17 | 278 | 3.4e-11 | **228** |

TABLE 3
*Results for sparse BQPs ($t = 10$).*

| $q$ | trial | MOSEK | | COPT | | SDPNAL+ | | ManiSDP | | ManiDSDP | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time | $\eta_{\max}$ | time |
| 10 | #1 | 1.2e-12 | **0.13** | 2.0e-09 | 0.33 | 1.0e-09 | 0.75 | 4.5e-14 | 0.31 | 4.3e-14 | 0.14 |
| | #2 | 2.3e-11 | **0.13** | 4.6e-10 | 0.32 | 7.0e-09 | 0.87 | 6.7e-17 | 0.23 | 3.3e-13 | 0.19 |
| | #3 | 5.2e-13 | **0.14** | 1.5e-09 | 0.32 | 4.7e-09 | 1.04 | 5.5e-17 | 0.34 | 2.1e-11 | 0.21 |
| 20 | #1 | 3.2e-13 | 58.6 | 2.7e-10 | 117 | 5.7e-08 | 90.6 | 2.1e-17 | 13.4 | 1.2e-12 | **4.37** |
| | #2 | 1.3e-13 | 64.1 | 1.1e-10 | 118 | 4.4e-09 | 39.9 | 8.8e-17 | 11.7 | 1.2e-12 | **3.34** |
| | #3 | 9.4e-14 | 60.0 | 2.4e-10 | 114 | 1.7e-09 | 67.2 | 2.2e-17 | 19.6 | 1.6e-12 | **4.93** |
| 30 | #1 | - | - | ** | ** | 9.1e-09 | 1092 | 1.1e-17 | 145 | 1.7e-12 | **58.0** |
| | #2 | - | - | ** | ** | 5.3e-09 | 619 | 9.6e-17 | 250 | 1.7e-12 | **59.4** |
| | #3 | - | - | ** | ** | 9.3e-09 | 1087 | 8.4e-18 | 208 | 1.7e-12 | **71.2** |
| 40 | #1 | - | - | * | * | * | * | 6.4e-17 | 3596 | 7.9e-12 | **756** |
| | #2 | - | - | * | * | 4.0e-09 | 3287 | 1.8e-16 | 1639 | 7.9e-12 | **566** |
| | #3 | - | - | * | * | 2.9e-07 | 6480 | 1.4e+00 | 4193 | 7.9e-12 | **1114** |

**6.3. ManiDSDP versus ManiSDP.** In this subsection, let us compare ManiDSDP with ManiSDP in more detail. In Fig. 1, we display the maximal factorization size reached through outer iterations of ManiDSDP and ManiSDP (each point is averaged over three random instances of (BQP)). In Fig. 2, we display the number of outer iterations taken by ManiDSDP and ManiSDP (each point is averaged over three random instances of (BQP)). We could see that ManiDSDP reaches smaller maximal factorization sizes and takes less numbers of outer iterations on the average compared to ManiSDP.

**6.4. (DSDP') versus (DSDP).** In this subsection, we compare the running time in solving (DSDP') and (DSDP) with our dual Riemannian ADMM algorithm. We display the results in Fig. 3 (each point is averaged over three random instances of (BQP)). One can see that it is more efficient to solve (DSDP') than to solve (DSDP), especially when $q$ becomes large.

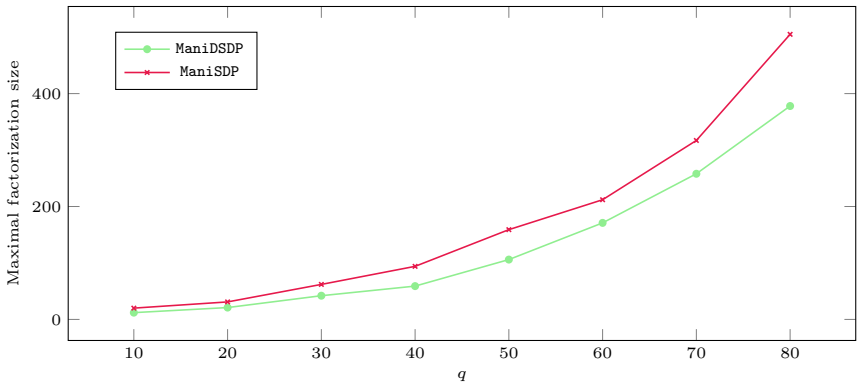


FIG. 1. *Comparison of maximal factorization sizes reached through outer iterations of* `ManiDSDP` *and* `ManiSDP`.

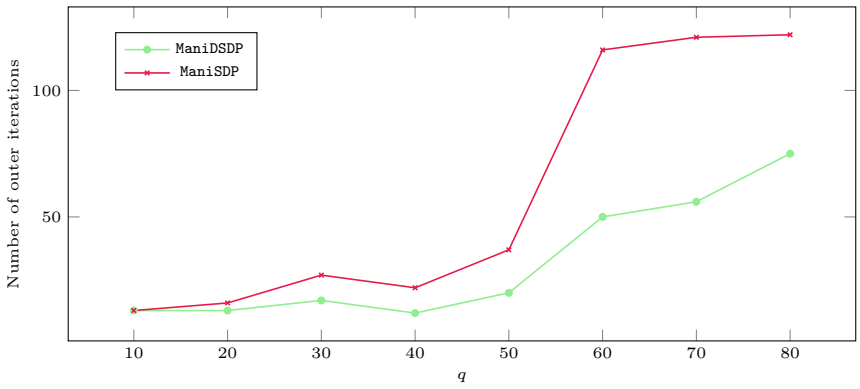


FIG. 2. *Comparison of numbers of outer iterations taken by* `ManiDSDP` *and* `ManiSDP`.

**6.5. The residue diving phenomenon.** When solving the SDP relaxation (6.4) of BQPs, `ManiDSDP` exhibits a residue diving phenomenon, that is, the maximal KKT residue sharply decreases to far less than $10^{-8}$ at some ADMM iteration. We illustrate this intriguing phenomenon with a random instance of (BQP) for $q \in \{10, 20, 30, 40, 50, 60\}$; see Fig. 4. Actually, the same phenomenon also occurs when solving BQPs with `ManiSDP`.

**7. Conclusions and future work.** In this paper, we have presented a dual Riemannian ADMM algorithm for solving low-rank SDPs with unit diagonal constraints. Global convergence of the algorithm is established assuming that the subproblem is solved to certain optimality. Numerical experiments demonstrate the superior performance of the algorithm. There are several directions for future research:

- We have empirically observed that our algorithm typically takes a few tens of ADMM iterations to converge to a solution of extremely high accuracy. This super fast convergence rate in comparison with traditional ADMM-based algorithms remains mysterious to us and should be carefully analyzed in future work. In particular, it is important to figure out which role the low-rank factorization plays for the acceleration, which will deepen our understanding of the algorithm.

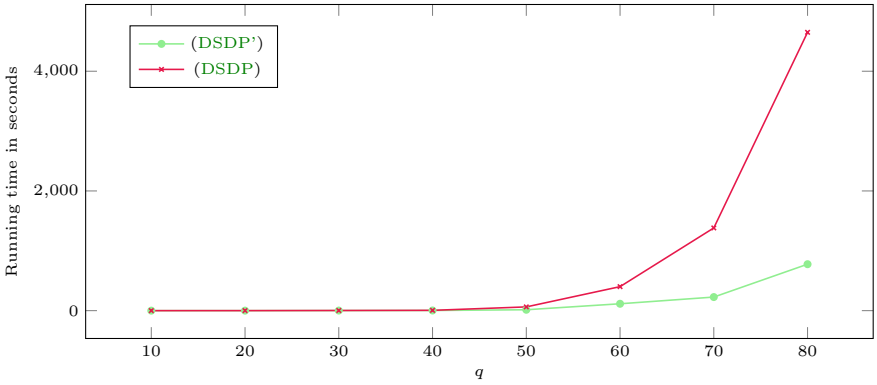


FIG. 3. *Comparison of running time in solving* (DSDP') *and* (DSDP).

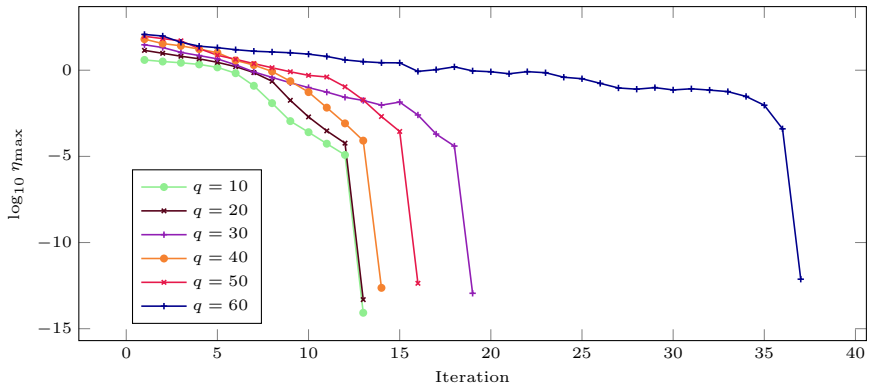


FIG. 4. *The maximal KKT residue per iteration with* `ManiDSDP`.

- An intriguing feature of our algorithm is the residue diving phenomenon as illustrated in Section 6.5. A clear theoretical explanation of this phenomenon is lacking and will be pursued in future work.
- Finally, in view of the strong performance, it would be very interesting to extend the algorithm to handle more types of SDPs.

## REFERENCES

[1] P.-A. ABSIL, C. G. BAKER, AND K. A. GALLIVAN, *Trust-region methods on Riemannian manifolds*, Found. Comput. Math., 7 (2007), pp. 303–330, https://doi.org/10.1007/s10208-005-0179-9.

[2] P. A. ABSIL, R. MAHONY, AND J. TRUMPF, *An extrinsic look at the Riemannian Hessian*, in Geometric Science of Information: First International Conference, GSI 2013, Paris, France, August 28-30, 2013. Proceedings, Springer, 2013, pp. 361–368.

[3] E. D. ANDERSEN, C. ROOS, AND T. TERLAKY, *On implementing a primal-dual interior-point method for conic quadratic optimization*, Mathematical Programming, 95 (2003), pp. 249–277.

[4] A. S. BANDEIRA, N. BOUMAL, AND V. VORONINSKI, *On the low-rank approach for semidefinite programs arising in synchronization and community detection*, in Conference on learning theory, PMLR, 2016, pp. 361–382.

[5] N. BOUMAL, B. MISHRA, P.-A. ABSIL, AND R. SEPULCHRE, *Manopt, a Matlab toolbox for optimization on manifolds*, Journal of Machine Learning Research, 15 (2014), pp. 1455–

1459, https://www.manopt.org.
[6] S. Burer and R. D. Monteiro, *A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization*, Mathematical Programming, 95 (2003), pp. 329–357.
[7] S. Burer and R. D. Monteiro, *Local minima and convergence in low-rank semidefinite programming*, Mathematical programming, 103 (2005), pp. 427–444.
[8] Y. Chen and P. Goulart, *Burer-monteiro admm for large-scale diagonally constrained sdps*, in 2022 European Control Conference (ECC), IEEE, 2022, pp. 66–71.
[9] S. F. Edwards and P. W. Anderson, *Theory of spin glasses*, Journal of Physics F: Metal Physics, 5 (1975), p. 965.
[10] M. A. Erdogdu, Y. Deshpande, and A. Montanari, *Inference in graphical models via semidefinite programming hierarchies*, Advances in Neural Information Processing Systems, 30 (2017).
[11] M. Garstka, M. Cannon, and P. Goulart, *Cosmo: A conic operator splitting method for convex conic problems*, Journal of Optimization Theory and Applications, 190 (2021), pp. 779–810.
[12] Q. Han, C. Li, Z. Lin, C. Chen, Q. Deng, D. Ge, H. Liu, and Y. Ye, *A low-rank admm splitting approach for semidefinite programming*, INFORMS Journal on Computing, (2025).
[13] D. Hou, T. Tang, and K.-C. Toh, *A low-rank augmented lagrangian method for doubly nonnegative relaxations of mixed-binary quadratic programs*, Operations Research, (2025).
[14] D. Hou, T. Tang, and K.-C. Toh, *Rinnal+: a riemannian alm solver for sdp-rlt relaxations of mixed-binary quadratic programs*, arXiv preprint arXiv:2507.13776, (2025).
[15] Y. Huang and D. F. Gleich, *Suboptimality bounds for trace-bounded sdps enable a faster and scalable low-rank sdp solver sdplr+*, arXiv preprint arXiv:2406.10407, (2024).
[16] M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre, *Low-rank optimization on the cone of positive semidefinite matrices*, SIAM Journal on Optimization, 20 (2010), pp. 2327–2351.
[17] S. Kang, X. Jiang, and H. Yang, *Local linear convergence of the alternating direction method of multipliers for semidefinite programming under strict complementarity*, arXiv preprint arXiv:2503.20142, (2025).
[18] A. Lemon, A. M.-C. So, Y. Ye, et al., *Low-rank semidefinite programming: Theory and applications*, Foundations and Trends® in Optimization, 2 (2016), pp. 1–156.
[19] S. Ling, *Solving orthogonal group synchronization via convex and low-rank optimization: Tightness and landscape analysis*, Mathematical Programming, 200 (2023), pp. 589–628.
[20] S. Ling, *Generalized orthogonal procrustes problem under arbitrary adversaries*, SIAM Journal on Matrix Analysis and Applications, 46 (2025), pp. 561–583.
[21] S. Ling, *Local geometry determines global landscape in low-rank factorization for synchronization*, Foundations of Computational Mathematics, (2025), pp. 1–33.
[22] C. Lu, Y.-F. Liu, W.-Q. Zhang, and S. Zhang, *Tightness of a new and enhanced semidefinite relaxation for mimo detection*, SIAM Journal on Optimization, 29 (2019), pp. 719–742.
[23] A. D. McRae, *Benign landscapes for synchronization on spheres via normalized laplacian matrices*, arXiv preprint arXiv:2503.18801, (2025).
[24] R. D. Monteiro, A. Sujanani, and D. Cifuentes, *A low-rank augmented lagrangian method for large-scale semidefinite programming based on a hybrid convex-nonconvex approach*, arXiv preprint arXiv:2401.12490, (2024).
[25] A. Mosek, *The MOSEK optimization Suite. Version 9.0.*, 2019, https://docs.mosek.com/9.0/intro/index.html.
[26] A. M.-C. So, Y. Ye, and J. Zhang, *A unified theorem on sdp rank reduction*, Mathematics of Operations Research, 33 (2008), pp. 910–920.
[27] M. Soltanalian and P. Stoica, *Designing unimodular codes via quadratic optimization*, IEEE Transactions on Signal Processing, 62 (2014), pp. 1221–1234.
[28] D. Sun, K.-C. Toh, Y. Yuan, and X.-Y. Zhao, *SDPNAL+: A Matlab software for semidefinite programming with bound constraints (version 1.0)*, Optimization Methods and Software, 35 (2020), pp. 87–115.
[29] T. Tang and K.-C. Toh, *A feasible method for general convex low-rank sdp problems*, SIAM Journal on Optimization, 34 (2024), pp. 2169–2200.
[30] K.-C. Toh, M. J. Todd, and R. H. Tütüncü, *Sdpt3—a matlab software package for semidefinite programming, version 1.3*, Optimization methods and software, 11 (1999), pp. 545–581.
[31] I. Waldspurger, A. d'Aspremont, and S. Mallat, *Phase recovery, maxcut and complex semidefinite programming*, Mathematical Programming, 149 (2015), pp. 47–81.
[32] J. Wang and L. Hu, *Solving low-rank semidefinite programs via manifold optimization*, Journal of Scientific Computing, 104 (2025), p. 33.
[33] J. Wang, J. Surace, I. Frérot, B. Legat, M.-O. Renou, V. Magron, and A. Ac´in,

*Certifying ground-state properties of quantum many-body systems*, arXiv preprint arXiv:2310.05844, (2023).

[34] Y. Wang, K. Deng, H. Liu, and Z. Wen, *A decomposition augmented lagrangian method for low-rank semidefinite programming*, SIAM Journal on Optimization, 33 (2023), pp. 1361–1390.

[35] Z. Wen, D. Goldfarb, and W. Yin, *Alternating direction augmented lagrangian methods for semidefinite programming*, Mathematical Programming Computation, 2 (2010), pp. 203–230.

[36] H. Yang, L. Liang, L. Carlone, and K.-C. Toh, *An inexact projected gradient method with rounding and lifting by nonlinear programming for solving rank-one semidefinite relaxation of polynomial optimization*, Mathematical Programming, (2022), pp. 1–64.

[37] L. Yang, D. Sun, and K.-C. Toh, *SDPNAL+: a majorized semismooth Newton-CG augmented lagrangian method for semidefinite programming with nonnegative constraints*, Mathematical Programming Computation, 7 (2015), pp. 331–366.

[38] X.-Y. Zhao, D. Sun, and K.-C. Toh, *A newton-cg augmented lagrangian method for semidefinite programming*, SIAM Journal on Optimization, 20 (2010), pp. 1737–1765.

[39] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn, *Chordal decomposition in operator-splitting methods for sparse semidefinite programs*, Mathematical Programming, 180 (2020), pp. 489–532.