Bregman primal–dual first-order method and application to sparse semidefinite programming

Xin Jiang* Lieven Vandenberghe*

August 9, 2021

Abstract

We present a new variant of the Chambolle–Pock primal–dual method with Bregman distances, analyze its convergence, and apply it to the centering problem in sparse semidefinite programming. The novelty in the method is a line search procedure for selecting suitable step sizes. The line search obviates the need for estimating the norm of the constraint matrix and the strong convexity constant of the Bregman kernel. As an application, we discuss the centering problem in large-scale semidefinite programming with sparse coefficient matrices. The logarithmic barrier function for the cone of positive semidefinite completable sparse matrices is used as the distance-generating kernel. For this distance, the complexity of evaluating the Bregman proximal operator is shown to be roughly proportional to the cost of a sparse Cholesky factorization. This is much cheaper than the standard proximal operator with Euclidean distances, which requires an eigenvalue decomposition.

1 Introduction

Optimization methods based on Bregman distances offer the possibility of matching the Bregman distance to the structure in the problem, with the goal of reducing the complexity per iteration. In this paper, we apply this idea to the centering problem in sparse semidefinite programming. The paper is motivated by the difficulty of exploiting sparsity in large-scale semidefinite programming in general and, for proximal methods, the need for eigendecompositions to compute Euclidean projections on the positive semidefinite matrix cone. By replacing the Euclidean projection with a generalized Bregman projection, we take advantage of the efficiency and scalability of algorithms for sparse Cholesky factorization and several related computations [3, 54].

We consider semidefinite programs (SDPs) in the standard form

$$\begin{array}{lllll} \text{primal:} & \text{minimize} & \mathbf{tr}(CX) & \text{dual:} & \text{maximize} \quad b^T y \\ & \text{subject to} & \mathcal{A}(X) = b & & \text{subject to} \quad \mathcal{A}^*(y) + S = C \\ & & X \in \mathbf{S}^n_+ & & \qquad\qquad S \in \mathbf{S}^n_+, \end{array} \tag{1}$$

with primal variable $X \in \mathbf{S}^n$ and dual variables $S \in \mathbf{S}^n$, $y \in \mathbf{R}^m$, where $\mathbf{S}^n$ is the set of symmetric $n \times n$ matrices. The linear operator $\mathcal{A}\colon \mathbf{S}^n \to \mathbf{R}^m$ is defined as

$$\mathcal{A}(X) = \big(\mathbf{tr}(A_1 X), \mathbf{tr}(A_2 X), \ldots, \mathbf{tr}(A_m X)\big)$$

*Department of Electrical and Computer Engineering, UCLA. Email: jiangxjames@ucla.edu, vandenbe@ucla.edu.
Research supported in part by NSF grant ECCS 1509789.

and $\mathcal{A}^*(y) = \sum_{i=1}^m y_i A_i$ is its adjoint operator. The coefficients $C, A_1, \ldots, A_m$ are symmetric $n \times n$ matrices. The notation $\mathbf{S}^n_+$ is used for the cone of positive semidefinite (PSD) matrices in $\mathbf{S}^n$.

In many large-scale applications of semidefinite programming, the coefficient matrices are sparse. The sparsity pattern of a symmetric $n \times n$ matrix can be represented by an undirected graph $G = (V, E)$ with vertex set $V = \{1, 2, \ldots, n\}$ and edge set E. The set of matrices with sparsity pattern E is then defined as

$$\mathbf{S}^n_E = \{Y \in \mathbf{S}^n \mid Y_{ij} = Y_{ji} = 0 \text{ if } i \neq j \text{ and } \{i, j\} \notin E\}.$$

In this paper, E will denote the common (or *aggregate*) sparsity pattern of the coefficient matrices in the SDP, *i.e.*, we assume that $C, A_1, \ldots, A_m \in \mathbf{S}^n_E$. Note that the sparsity pattern E is not uniquely defined (unless it is dense, *i.e.*, the sparsity graph G is complete): if the coefficients are in $\mathbf{S}^n_E$ then they are also in $\mathbf{S}^n_{E'}$ where $E \subset E'$. In particular, E can always be extended to make the graph $G = (V, E)$ *chordal* or *triangulated* [14, 54]. Without loss of generality, we will assume that this is the case.

The primal variable X in (1) generally needs to be dense to be feasible. However, the cost function and the linear equality constraints only depend on the diagonal entries X_{ii} and the off-diagonal entries $X_{ij} = X_{ji}$ for $\{i, j\} \in E$. For the other entries the only requirement is to make the matrix positive semidefinite. In the dual problem, $S \in \mathbf{S}^n_E$ holds at all dual feasible points. These observations imply that the SDPs (1) can be equivalently rewritten as a pair of primal and dual conic linear programs

$$\begin{array}{lllll} \text{primal:} & \text{minimize} & \mathbf{tr}(CX) & \text{dual:} & \text{maximize} \quad b^T y \\ & \text{subject to} & \mathcal{A}(X) = b & & \text{subject to} \quad \mathcal{A}^*(y) + S = C \\ & & X \in K & & \qquad\qquad S \in K^*, \end{array} \tag{2}$$

with *sparse* matrix variables $X, S \in \mathbf{S}^n_E$, and a vector variable $y \in \mathbf{R}^m$. The primal cone K in this problem is the set of matrices in $\mathbf{S}^n_E$ which have a positive semidefinite completion, *i.e.*, $K = \Pi_E(\mathbf{S}^n_+)$ where Π_E stands for projection on $\mathbf{S}^n_E$. The dual cone K^* of K is the set of positive semidefinite matrices with sparsity pattern E, *i.e.*, $K^* = \mathbf{S}^n_+ \cap \mathbf{S}^n_E$. The formulation (2) is attractive when the aggregate sparsity pattern E is very sparse, in which case $\mathbf{S}^n_E$ is a much lower-dimensional space than $\mathbf{S}^n$.

The centering problem for the sparse SDP (2) is

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) + \mu\phi(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{3}$$

where ϕ is the logarithmic barrier function for the cone K, defined as

$$\phi(X) = \sup_{S \in \mathbf{int}\, K^*} (-\mathbf{tr}(XS) + \log\det S).$$

The centering parameter $\mu > 0$ controls the duality gap at the solution. Since the barrier function ϕ is n-logarithmically homogeneous, the optimal solution of the centering problem is a (μn)-suboptimal solution for the original SDP (2). The centering problem (3) is useful as an approximation to the original problem, because it yields more easily computed suboptimal solutions, with an accuracy that can be controlled by the choice of barrier parameter. The centering problem is also a key component of barrier methods, in which a sequence of centering problems with decreasing

values of the barrier parameter are solved. Traditionally, the centering problem in interior-point methods is solved by Newton's algorithm, possibly accelerated via the preconditioned conjugate gradient method [10,55], but recent work has started to examine the use of proximal methods such as the alternating direction method of multipliers (ADMM) or the proximal method of multipliers for this purpose [37,48].

Contributions The contribution of this paper is two-fold. First, we formulate a non-Euclidean (Bregman) proximal method for the centering problem of the sparse SDP. In the proposed method, the proximal operators are replaced by generalized proximal operators defined in terms of a Bregman generalized distance or divergence. We show that if the Bregman divergence generated by the barrier function ϕ for the cone K is used, the generalized projections can be computed very efficiently, with a complexity dominated by the cost of a sparse Cholesky factorization with sparsity pattern E. This is much cheaper than the eigenvalue decomposition needed to compute a Euclidean projection on the positive semidefinite cone. Hence, while the method only solves an approximation of the SDP (2), it can handle problem sizes that are orders of magnitude larger than the problems solved by standard interior-point and proximal first-order methods.

For the solution of the centering problem, we apply a variant of the primal–dual method proposed by Chambolle and Pock [22]. The version of the algorithm described in [22] requires careful tuning of primal and dual step size parameters. Acceptable values of the step sizes depend on the norm of the linear operator $\mathcal{A}$ and the strong convexity constants for the distance function. These parameters are often difficult to estimate in practice. As a second contribution, we propose a new version of the algorithm, in which the step sizes are not fixed parameters, but are selected using an easily implemented line search procedure. We give a detailed convergence analysis of the algorithm with line search and show an $O(1/k)$ ergodic convergence rate, which is consistent with previous results in [22,39].

Related work Sparse structure in semidefinite programming has been extensively studied by many authors. The scalability of interior-point methods is limited by the need to form and solve a set of m linear equations in m variables, known as the *Schur complement system*, at each iteration. This system is usually dense. Sparsity in the coefficients A_i can be exploited to reduce the cost of assembling the Schur complement equations. This process is efficient especially in extremely sparse problems, where the coefficients A_i may also have low rank. In dual barrier methods, one can also take advantage of sparsity of dual feasible variables S. These properties are leveraged in the dual interior-point methods described in [9–13].

In another line of research, techniques based on properties and algorithms for chordal sparsity patterns have been applied to semidefinite programming since the late 1990s [3, 13, 18, 29, 30, 34, 35, 42, 46, 50, 51, 58]; see [54, 60] for recent surveys. An important tool from this literature is the *conversion* or *clique decomposition method* proposed by Fukuda *et al.* [30, 42]. It is based on a fundamental result from linear algebra, stating that for a chordal pattern E, a matrix $X \in \mathbf{S}^n_E$ has a positive semidefinite completion if and only if $X_{\gamma_k\gamma_k} \succeq 0$ for $k = 1, \ldots, r$, where γ_1, $\ldots$, γ_r are the maximal cliques in the graph [31]. In the conversion method, the large sparse variable matrix X in (2) is replaced with smaller dense matrix variables $X_k = X_{\gamma_k\gamma_k}$. Each of these new variables is constrained to be positive semidefinite. Linear equality constraints need to be added to couple the variables X_k, as they represent overlapping subblocks of a single matrix X. Thus, a large sparse SDP is converted in an equivalent problem with several smaller, dense variables X_k,

and additional sparse equality constraints. This equivalent problem may be considerably easier to solve by interior-point methods than the original SDP (1). Recent examples where the clique decomposition is applied to solve large sparse SDPs can be found in [27, 58].

Proximal splitting methods, such as (accelerated) proximal gradient methods [7, 8, 43], ADMM [16], and the primal–dual hybrid gradient (PDHG) or Chambolle–Pock method [20, 28, 47], are perhaps the most popular alternatives to interior-point methods in machine learning, image processing, and other applications involving large-scale convex programming. When applied to the SDPs (1), they require at each iteration a Euclidean projection on the positive semidefinite cone $\mathbf{S}_+^n$, hence, a symmetric eigenvalue decomposition of order n. This contributes an order n^3 term to the per-iteration complexity. In the nonsymmetric formulation (2) of the sparse SDP, the projections on K^* or (equivalently) K cannot be computed directly, and must be handled by introducing splitting variables and alternating projection on $\mathbf{S}_E^n$, which is trivial, and on $\mathbf{S}_+^n$, which requires an eigenvalue decomposition. The clique decomposition used in the conversion method described above, which was originally developed for interior-point methods, lends itself naturally to splitting algorithms as well. It allows us to replace the matrix constraint $X \in K$ with several smaller dense inequalities $X_k \succeq 0$, one for each maximal clique in the sparsity graph. In a proximal method, this means that projection on the $n \times n$ positive semidefinite cone can be replaced by less expensive projections on lower-dimensional positive semidefinite cones [38, 52, 59, 61]. This advantage of the conversion method is tempered by the large number of consistency constraints that must be introduced to link the splitting variables X_k. First-order methods typically do not compute very accurate solutions and if the residual error in the consistency constraints is not small, it may be difficult to convert the computed solution of the decomposed problem back to an accurate solution of the original SDP [27].

Outline The rest of the paper is organized as follows. In Section 2 we describe the Bregman distance generated by the barrier function and show how generalized projections can be efficiently computed without expensive eigenvalue decomposition. The primal–dual proximal algorithm and its convergence are discussed in Section 3. Section 4 contains results of numerical experiments.

2 Barrier proximal operator for sparse PSD matrix cone

2.1 Centering problem

We will assume that the equality constraints in (2) include a constraint $\mathbf{tr}(NX) = 1$, where $N \in \mathbf{S}_{++}^n \cap \mathbf{S}_E^n$. To make this explicit we write the centering problem (2) as

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) + \mu\phi(X) \\ \text{subject to} & \mathcal{A}(X) = b, \\ & \mathbf{tr}(NX) = 1. \end{array} \tag{4}$$

For $N = I$, the normalized cone $\{X \in K \mid \mathbf{tr}(NX) = 1\}$ is a matrix extension of the probability simplex $\{x \succeq 0 \mid \mathbf{1}^T x = 1\}$, sometimes referred to as the *spectraplex*. With minor changes, the techniques we discuss extend to a normalization in the inequality form $\mathbf{tr}(NX) \leq 1$, with $N \in \mathbf{S}_{++}^n \cap \mathbf{S}_E^n$. However, we will discuss (4) to retain the standard form of the centering problem.

The constraints $\mathbf{tr}(NX) = 1$ and $\mathbf{tr}(NX) \leq 1$ guarantee the boundedness of the primal feasible set, a common assumption in first-order methods. The added constraint does not diminish the

generality of our approach. In many applications an equality $\mathbf{tr}(NX) = 1$ is implied by the contraints $\mathcal{A}(X) = b$ and easily derived from the problem data (see Section 4 for two typical examples). When an equality constraint of this form is not readily available, one can add a bounding inequality $\mathbf{tr}(NX) \leq 1$ with N sufficiently small to ensure that the optimal solution is not modified.

To apply first-order proximal methods, we view the problem (4) as a linearly constrained optimization problem

$$\begin{array}{ll} \text{minimize} & f(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{5}$$

where f is defined as

$$f(X) = \mathbf{tr}(CX) + \mu\phi(X) + \delta_{\mathcal{H}}(X), \qquad \mathcal{H} = \{X \in \mathbf{S}_E^n \mid \mathbf{tr}(NX) = 1\}, \tag{6}$$

and δ_H is the indicator function of the hyperplane $\mathcal{H}$. The algorithm we apply to (5) can be summarized as

$$\bar{z}_{k+1} = z_k + \theta_k(z_k - z_{k-1}) \tag{7a}$$

$$X_{k+1} = \operatorname*{argmin}_X \left(f(X) + \bar{z}_{k+1}^T \mathcal{A}(X) + \frac{1}{\tau_k} d(X, X_k)\right) \tag{7b}$$

$$z_{k+1} = z_k + \sigma_k(\mathcal{A}(X_{k+1}) - b) \tag{7c}$$

where d is the Bregman distance generated by the barrier function ϕ:

$$d(X, Y) = \phi(X) - \phi(Y) - \mathbf{tr}(\nabla\phi(Y)(X - Y)).$$

The choices of θ_k, σ_k, and τ_k, together with the details and origins of the algorithm, will be discussed in Section 3. In the remainder of this section we focus on the most expensive step in the algorithm, the optimization problem in the X-update (7b).

In Sections 2.2 and 2.3 we first review some facts from the theory of generalized distances and the logarithmic barrier functions for the primal and dual cones K and K^*. Sections 2.4 and 2.5 describe the details of the barrier kernel and the associated generalized proximal operator applied in (7b).

2.2 Bregman distance

Let h be a convex function, defined on a domain that has nonempty interior, and suppose h is continuously differentiable on $\mathbf{int}\,(\mathbf{dom}\,h)$. The *generalized distance* generated by h is defined as the function

$$d(x, y) = h(x) - h(y) - \langle \nabla h(y), x - y\rangle,$$

with domain $\mathbf{dom}\,d = \mathbf{dom}\,h \times \mathbf{int}\,(\mathbf{dom}\,h)$. The function h is called the *kernel function* that generates the generalized distance d. For $h(x) = \|x\|_2^2/2$ and the standard inner product $\langle u, v\rangle = u^T v$, we obtain $d(x, y) = \|x - y\|_2^2/2$. The best known non-quadratic example is the relative entropy

$$d(x, y) = \sum_{i=1}^n (x_i \log(x_i/y_i) - x_i + y_i), \qquad \mathbf{dom}\,d = \mathbf{R}_+^n \times \mathbf{R}_{++}^n.$$

This generalized distance is generated by the kernel $h(x) = \sum_i x_i \log x_i$, if we use the standard inner product.

Generalized distances are not necessarily symmetric ($d(x,y) \neq d(y,x)$ in general) but share some other important properties with the squared Euclidean norm. An important example is the *triangle identity* [23, Lemma 3.1]

$$\langle \nabla h(y) - \nabla h(z), x - y \rangle = d(x,z) - d(x,y) - d(y,z) \tag{8}$$

which holds for all $x \in \mathbf{dom}\, h$ and $y, z \in \mathbf{int}\,(\mathbf{dom}\, h)$. This generalizes the identity

$$(y-z)^T(x-y) = \frac{1}{2}\left(\|x-z\|_2^2 - \|x-y\|_2^2 - \|y-z\|_2^2\right).$$

Additional conditions may have to be imposed on the kernel function h, depending on the application and the algorithm in which the generalized distance is used [19]. For now we only assume convexity and continuous differentiability on the interior of the domain. Other properties will be mentioned when needed.

The *proximal operator* of a closed convex function f is defined as

$$\mathrm{prox}_f(y) = \operatorname*{argmin}_x \left(f(x) + \frac{1}{2}\|x-y\|_2^2\right).$$

If f is closed and convex, then the minimizer in the definition exists and is unique for all y [40]. We will use the following extension to generalized distances. Suppose f is a convex function with the property that for every a and every $y \in \mathbf{int}\,(\mathbf{dom}\, h)$, the optimization problem

$$\text{minimize} \quad f(x) + \langle a, x \rangle + d(x,y) \tag{9}$$

has a unique solution $\hat{x}$ in $\mathbf{int}\,(\mathbf{dom}\, h)$. Then we denote the minimizer $\hat{x}$ by

$$\begin{aligned} \mathrm{prox}_f^d(y,a) &= \operatorname*{argmin}_x \left(f(x) + \langle a, x \rangle + d(x,y)\right) \\ &= \operatorname*{argmin}_x \left(f(x) + \langle a, x \rangle + h(x) - \langle \nabla h(y), x \rangle\right) \end{aligned} \tag{10}$$

and call the mapping prox_f^d the *generalized proximal operator* of f. From the second expression we see that $\hat{x} = \mathrm{prox}_f^d(y,a)$ satisfies

$$\nabla h(y) - \nabla h(\hat{x}) - a \in \partial f(\hat{x}). \tag{11}$$

If $d = \|x-y\|_2^2/2$, it is easily verified that $\mathrm{prox}_f^d(y,a) = \mathrm{prox}_f(y-a)$, where prox_f is the standard proximal operator.

In contrast to the Euclidean case, it is difficult to give simple general conditions that guarantee that for every a and every $y \in \mathbf{int}\,(\mathbf{dom}\, h)$ the problem (9) has a unique solution in $\mathbf{int}\,(\mathbf{dom}\, h)$. However, we will use the definition only for specific combinations of f and d, for which problem (9) is particularly easy to solve. In those applications, existence and uniqueness of the solution follow directly from the availability of a fast algorithm for computing it. A classical example is the relative entropy distance with f given by the indicator function of the hyperplane $\{x \mid \mathbf{1}^T x = 1\}$. Problem (9) can be written as

$$\begin{array}{ll} \text{minimize} & a^T x + \sum_{i=1}^n (x_i \log(x_i/y_i) - x_i) \\ \text{subject to} & \mathbf{1}^T x = 1. \end{array}$$

For any a and any positive y, the solution of (9) is unique and equal to the positive vector

$$\mathrm{prox}_f^d(y,a) = \frac{1}{\sum_{i=1}^n y_i e^{-a_i}} \begin{bmatrix} y_1 e^{-a_1} \\ \vdots \\ y_n e^{-a_n} \end{bmatrix}.$$

Research on proximal methods for semidefinite programming has been largely based on the standard Euclidean proximal operators and the distance defined by the matrix entropy [6]. For these distances, projections on the positive semidefinite cone require eigenvalue decompositions, which limits the size of the variables that can be handled and precludes applications to large sparse SDPs. In the following sections, we introduce a generalized proximal operator designed for sparse semidefinite programming. The generalized proximal operator can be evaluated via a simple iterative algorithm with a complexity dominated by the cost of a sparse Cholesky factorization.

2.3 Primal and dual barrier

The logarithmic barrier functions for the cones $K^* = \mathbf{S}_+^n \cap \mathbf{S}_E^n$ and $K = \Pi_E(\mathbf{S}_+^n)$ are defined as

$$\phi_*(S) = -\log\det S, \qquad \phi(X) = \sup_S \big(-\mathbf{tr}(XS) - \phi_*(S)\big), \tag{12}$$

with domains $\mathbf{dom}\,\phi_* = \mathbf{int}\,K^*$ and $\mathbf{dom}\,\phi = \mathbf{int}\,K$, respectively. Note that $\phi(X)$ is the conjugate of ϕ_* evaluated at $-X$.

In [3, 54] efficient algorithms are presented for evaluating the two barrier functions, their gradients, and their directional second derivatives, when the sparsity pattern E is chordal. The value of the dual barrier $\phi_*(S) = -\log\det S$ is easily computed from the diagonal entries in a sparse Cholesky factor of S. The gradient and Hessian are given by

$$\nabla\phi_*(S) = -\Pi_E(S^{-1}), \qquad \nabla^2\phi_*(S)[V] = \frac{d}{dt}\nabla\phi_*(S+tV) = \Pi_E(S^{-1}VS^{-1}). \tag{13}$$

Given a Cholesky factorization of S, these expressions can be evaluated via one or two recursions on the elimination tree [3, 54], without explicitly computing the entire inverse S^{-1} or the matrix product $S^{-1}VS^{-1}$. The cost of these recursions is roughly the same as the cost of a sparse Cholesky factorization with the sparsity pattern E [3, 54].

The primal barrier function ϕ and its gradient can be evaluated by solving the optimization problem in the definition of $\phi(X)$. The optimal solution $\hat{S}_X$ is the matrix in $\mathbf{S}_{++}^n \cap \mathbf{S}_E^n$ that satisfies

$$\Pi_E(\hat{S}_X^{-1}) = X. \tag{14}$$

Its inverse $\hat{S}_X^{-1}$ is also the maximum determinant positive definite completion of X, *i.e.*, $Z = \hat{S}_X^{-1}$ is the solution of

$$\begin{array}{ll} \text{maximize} & \log\det Z \\ \text{subject to} & \Pi_E(Z) = X \end{array} \tag{15}$$

(where we take $\mathbf{S}_{++}^n$ as the domain of the cost function). From $\hat{S}_X$, one obtains

$$\phi(X) = \log\det\hat{S}_X - n, \qquad \nabla\phi(X) = -\hat{S}_X, \qquad \nabla^2\phi(X) = \nabla^2\phi_*(\hat{S}_X)^{-1}. \tag{16}$$

Comparing the expressions for the gradients of ϕ and ϕ_* in (16) and (13), and using (14), we see that $\nabla\phi$ and $\nabla\phi_*$ are inverse mappings, up to a change in sign:

$$\nabla\phi(X) = -\hat{S}_X = -(\nabla\phi_*)^{-1}(-X), \qquad \nabla\phi_*(S) = -(\nabla\phi)^{-1}(-S).$$

For general sparsity patterns, the determinant maximization problem (15) or the convex optimization problem in the definition of ϕ must be solved by an iterative optimization algorithm. If the pattern is chordal, these optimization problems can be solved by finite recursive algorithms, again at a cost that is comparable with the cost of a sparse Cholesky factorization for the same pattern [3, 54].

2.4 Barrier kernel

The primal barrier function ϕ is convex, continuously differentiable on the interior of the cone, and strongly convex on $\mathbf{int}\, K \cap \{X \mid \mathbf{tr}(NX) = 1\}$. It generates the Bregman divergence

$$\begin{aligned} d(X,Y) &= \phi(X) - \phi(Y) - \mathbf{tr}\left(\nabla\phi(Y)(X-Y)\right) \\ &= \phi(X) - \log\det\hat{S}_Y + n + \mathbf{tr}\left(\hat{S}_Y(X-Y)\right) \\ &= \phi(X) - \log\det\hat{S}_Y + \mathbf{tr}\left(\hat{S}_Y X\right). \end{aligned}$$

On line 2 we used the properties (16) to express $\phi(Y)$ and $\nabla\phi(Y)$. The generalized proximal operator (10) for the function f defined in (6), which is the key step in the X-update (7b) of algorithm (7), then becomes

$$\begin{aligned} \hat{X} &= \mathrm{prox}_f^d(Y,A) \\ &= \operatorname*{argmin}_{\mathbf{tr}(NX)=1} \left(\mathbf{tr}(CX) + \mu\phi(X) + \mathbf{tr}(AX) + d(X,Y)\right) \\ &= \operatorname*{argmin}_{\mathbf{tr}(NX)=1} \left(\mathbf{tr}\left((C + A - \nabla\phi(Y))X\right) + (\mu+1)\phi(X)\right) \\ &= \operatorname*{argmin}_{\mathbf{tr}(NX)=1} \left(\mathbf{tr}(BX) + \phi(X)\right) \end{aligned}$$

where

$$B = \frac{1}{1+\mu}(C + A + \hat{S}_Y).$$

To compute $\hat{X}$ we therefore need to solve an optimization problem

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(BX) + \phi(X) \\ \text{subject to} & \mathbf{tr}(NX) = 1, \end{array} \tag{17}$$

where $B \in \mathbf{S}_E^n$ and $N \in \mathbf{S}_{++}^n \cap \mathbf{S}_E^n$. If we introduce a Lagrange multiplier ν for the equality constraint in (17), the optimality condition can be written as

$$\nabla\phi(X) + B + \nu N = 0, \qquad \mathbf{tr}(NX) = 1.$$

Equivalently, since $\nabla\phi_*(S) = -(\nabla\phi)^{-1}(-S)$,

$$X = -\nabla\phi_*(B + \nu N) = \Pi_E((B + \nu N)^{-1}), \qquad \mathbf{tr}(NX) = 1.$$

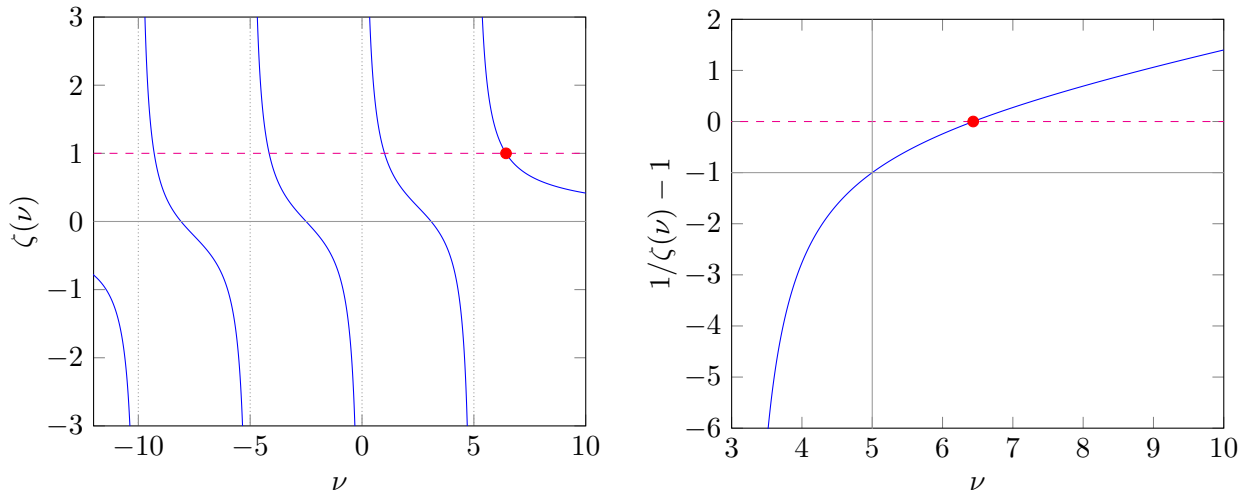

Figure 1: *Left.* The function $\zeta(\nu) = \sum_i 1/(\nu + \lambda_i)$ for $\lambda = (-5, 0, 5, 10)$. We are interested in the solution of $\zeta(\nu) = 1$ larger than $-\lambda_{\min} = 5$. *Right.* The function $1/\zeta(\nu) - 1$.

Eliminating X we obtain a nonlinear equation in ν:

$$\mathbf{tr}(N(B + \nu N)^{-1}) = 1. \tag{18}$$

(The projection in $\mathbf{tr}(N\Pi_E((B + \nu N)^{-1}))$ can be omitted because the matrix N has the sparsity pattern E.) The unique solution ν that satisfies $B + \nu N \succ 0$ defines the solution $X = \Pi_E((B + \nu N)^{-1})$ of (17).

The equation (18) is also the optimality condition for the Lagrange dual of (17), which is a smooth unconstrained convex optimization problem in the scalar variable ν:

$$\text{maximize} \quad -\phi_*(B + \nu N) - \nu. \tag{19}$$

2.5 Newton method for barrier proximal operator

In this section we discuss in detail Newton's method applied to the dual problem (19) and the equivalent nonlinear equation (18). We write the equation as $\zeta(\nu) = 1$ where

$$\zeta(\nu) = \mathbf{tr}(N(B + \nu N)^{-1}), \qquad \zeta'(\nu) = -\mathbf{tr}(N(B + \nu N)^{-1}N(B + \nu N)^{-1}). \tag{20}$$

The function ζ and its derivative can be expressed in terms of the generalized eigenvalues λ_i of (B, N) as

$$\zeta(\nu) = \sum_{i=1}^{n} \frac{1}{\nu + \lambda_i}, \qquad \zeta'(\nu) = -\sum_{i=1}^{n} \frac{1}{(\nu + \lambda_i)^2}. \tag{21}$$

Figure 1 shows an example with $n = 4$, $N = I$, and eigenvalues $10, 5, 0, -5$.

We are interested in computing the solution of $\zeta(\nu) = 1$ that satisfies $B + \nu N \succ 0$, *i.e.*, $\nu > -\lambda_{\min}$, where $\lambda_{\min} = \min_i \lambda_i$ is the smallest generalized eigenvalue of (B, N). We denote this interval by $J = (-\lambda_{\min}, \infty)$. The equation $\zeta(\nu) = 1$ is guaranteed to have a unique solution in J because ζ is monotonic and continuous on this interval, with

$$\lim_{\nu \to -\lambda_{\min}} \zeta(\nu) = \infty, \qquad \lim_{\nu \to \infty} \zeta(\nu) = 0.$$

Furthermore, on the interval J, the function ζ and its derivative can be expressed as

$$\zeta(\nu) = -\,\mathbf{tr}(N\nabla\phi_*(B+\nu N)), \qquad \zeta'(\nu) = -\,\mathbf{tr}(N(\nabla^2\phi_*(B+\nu N)[N])).$$

Therefore $\zeta(\nu)$ and $\zeta'(\nu)$ can be evaluated by taking the inner product of N with

$$\begin{aligned} \nabla\phi_*(B+\nu N) &= -\Pi_E\left((B+\nu N)^{-1}\right) \\ \nabla^2\phi_*(B+\nu N)[N] &= -\Pi_E\left((B+\nu N)^{-1}N(B+\nu N)^{-1}\right). \end{aligned}$$

Since $B, N \in \mathbf{S}^n_E$, these quantities can be computed by the efficient algorithms for computing the gradient and directional second derivative of ϕ_* described in [3, 54].

We note a few other properties of ζ. First, the expressions in (21) show that ζ is convex, decreasing, and positive on J. Second, if $\nu \in J$, then $\tilde{\nu} \in J$ for all $\tilde{\nu}$ that satisfy

$$\tilde{\nu} > \nu - \frac{1}{\sqrt{|\zeta'(\nu)|}}. \tag{22}$$

This follows from

$$|\zeta'(\nu)| = \sum_{i=1}^n \frac{1}{(\nu+\lambda_i)^2} \geq \frac{1}{(\nu+\lambda_{\min})^2},$$

and is also a simple consequence of the Dikin ellipsoid theorem for self-concordant functions [44, Theorem 2.1.1.b].

The Newton iteration for the equation $\zeta(\nu) - 1 = 0$ is

$$\nu^+ = \nu + \alpha\frac{1-\zeta(\nu)}{\zeta'(\nu)}, \tag{23}$$

where α is a step size. The same iteration can be interpreted as a damped Newton method for the unconstrained problem (19). If $\nu^+ \in J$ for a unit step $\alpha = 1$, then

$$\zeta(\nu^+) > \zeta(\nu) + \zeta'(\nu)(\nu^+ - \nu) = 1,$$

from strict convexity of ζ. Hence after one full Newton step, the Newton iteration with unit steps approaches the solution monotonically from the left. If $\zeta(\nu) < 1$ then in general a non-unit step size must be taken to keep the iterates in J. From the Dikin ellipsoid inequality (22), we see that $\nu^+ \in J$ for all positive α that satisfy

$$\alpha < \frac{\sqrt{|\zeta'(\nu)|}}{1-\zeta(\nu)}.$$

The theory of self-concordant functions provides a step size rule that satisfies this condition and guarantees convergence:

$$\alpha = \frac{\sqrt{|\zeta'(\nu)|}}{\sqrt{|\zeta'(\nu)|}+1-\zeta(\nu)} \quad \text{if} \quad \frac{1-\zeta(\nu)}{\sqrt{|\zeta'(\nu)|}} < \eta, \qquad \alpha = 1 \quad \text{otherwise},$$

where η is a constant in $(0, 1)$. As an alternative to this fixed step size rule, a standard backtracking line search can be used to determine a suitable step size α in (23). Checking whether $\nu^+ \in J$ can be done by attempting a sparse Cholesky factorization of $B + \nu^+ N$.

Figure 1 shows that the function ζ can be quite nonlinear around the solution of the equation if the solution is near $-\lambda_{\min}$. Instead of applying Newton's method directly to (20), it is useful to rewrite the nonlinear equation as $\psi(\nu) = 0$ where

$$\psi(\nu) = \frac{1}{\zeta(\nu)} - 1. \tag{24}$$

The negative smallest eigenvalue $-\lambda_{\min}$ is a pole of $\zeta(\nu)$, but a zero of $1/\zeta(\nu)$. Also the derivative of ψ changes slowly near this zero point; in Figure 1, the function ψ is almost linear in the region of interest. This implies that Newton's method applied to (24), *i.e.*,

$$\nu^+ = \nu + \beta\frac{\psi(\nu)}{\psi'(\nu)} = \nu + \beta\frac{\zeta(\nu)(1-\zeta(\nu))}{\zeta'(\nu)},$$

should be extremely efficient in this case. Starting the line search at $\beta = 1$ is equivalent to starting at $\alpha = \zeta(\nu)$ in (23). This often requires fewer backtracking steps than starting at $\alpha = 1$.

Newton's method requires a feasible initial point $\nu_0 \in J$. Suppose we know a positive lower bound γ on the smallest eigenvalue of N. Then $\hat{\nu}_0 \in J$ where

$$\hat{\nu}_0 > \max\{0, \frac{-\lambda_{\min}(B)}{\gamma}\}.$$

A lower bound on $\lambda_{\min}(B)$ can be obtained from the Gershgorin circle theorem, which states that the eigenvalues of B are contained in the disks

$$\Big\{ s \;\Big|\; |s - B_{ii}| \le \sum_{j\neq i} |B_{ij}| \Big\}, \quad i = 1, \ldots, n.$$

Thus, $\lambda_{\min}(B) \ge \min_i (B_{ii} - \sum_{j\neq i} |B_{ij}|)$. Apart from the above initialization, we find another practically useful initial point $\tilde{\nu}_0 = n - \mathbf{tr}\, B / \mathbf{tr}\, N$, which is the solution for $\mathbf{tr}(N(B+\nu N)^{-1}) = 1$ when B happens to be a multiple of N. This choice is efficient in many practical examples but, unfortunately, not guaranteed to be feasible. Thus, in the implementation, we use $\tilde{\nu}_0$ if it is feasible and $\hat{\nu}_0$ otherwise.

3 Bregman primal–dual method

The proposed algorithm (7) is applicable not only to sparse SDPs, but to more general optimization problems. To emphasize its generality and to simplify notation, we switch in this section to the vector form of the optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array} \tag{25}$$

where f is a closed convex function. Most of the discussion in this section extends to the more general standard form

$$\text{minimize} \quad f(x) + g(Ax), \tag{26}$$

where f and g are closed convex functions. Problem (25) is a special case with $g = \delta_{\{b\}}$, the indicator function of the singleton $\{b\}$. While the standard form (26) offers more flexibility, it

should be noted that methods for the equality constrained problem (25) also apply to (26) if this problem is reformulated as

$$\begin{array}{ll} \text{minimize} & f(x) + g(y) \\ \text{subject to} & Ax - y = 0. \end{array}$$

We also note that (25) includes conic optimization problems in standard form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \in C \end{array}$$

if we define $f(x) = c^T x + \delta_C(x)$, where δ_C is the indicator function of the cone C.

In Section 3.1 we review some facts from convex duality theory. Section 3.2 describes the algorithm we propose for solving (25), and in Section 3.3 we analyze its convergence.

3.1 Duality theory

The Lagrangian for problem (25) will be denoted by

$$L(x, z) = f(x) + z^T(Ax - b). \tag{27}$$

This function is convex in x and affine in z, and satisfies

$$\sup_z L(x, z) = f(x) + \delta_{\{b\}}(Ax) = \begin{cases} f(x) & Ax = b \\ +\infty & \text{otherwise,} \end{cases} \qquad \inf_x L(x, z) = -f^*(-A^T z) + b^T z,$$

where $f^*(y) = \sup_x (y^T x - f(x))$ is the conjugate of f. The function $f^*(-A^T z)$ is the objective in the dual problem

$$\text{maximize} \quad -f^*(-A^T z) + b^T z. \tag{28}$$

A point $(x^\star, z^\star)$ is a *saddle point* of the Lagrangian if

$$\sup_z L(x^\star, z) = L(x^\star, z^\star) = \inf_x L(x, z^\star). \tag{29}$$

Existence of a saddle point is equivalent to the property that the primal and dual optimal values are equal and attained. The left-hand equality in (29) holds if and only if $Ax^\star = b$. The right-hand equality holds if and only if $-A^T z^\star \in \partial f(x^\star)$. Hence $(x^\star, z^\star)$ is a saddle point if and only if it satisfies the optimality conditions

$$Ax^\star = b, \qquad -A^T z^\star \in \partial f(x^\star).$$

Throughout this section we assume that there exists a saddle point $(x^\star, z^\star)$.

Some of the convergence results in Section 3.3 are expressed in terms of the merit function

$$f(x) + \gamma \|Ax - b\|_2. \tag{30}$$

It is well known that for sufficiently large γ, the term $\gamma\|Ax - b\|_2$ is an *exact penalty*. Specifically, if $\gamma > \|z^\star\|_2$, where $z^\star$ is a solution of the dual problem (28), then optimal solutions of (30) are also optimal for (25).

3.2 Algorithm

The algorithm for (25) presented in this section involves a generalized distance d in the primal space, generated by a kernel function ϕ. It will be assumed that ϕ is strongly convex on $\mathbf{dom}\, f$. This property can be expressed as

$$d(x,y) \geq \frac{1}{2}\|x-y\|^2 \quad \text{for all } x \in \mathbf{dom}\,\phi \cap \mathbf{dom}\, f,\ y \in \mathbf{int}\,(\mathbf{dom}\,\phi) \cap \mathbf{dom}\, f \tag{31}$$

where $\|\cdot\|$ is a norm, scaled so that the strong convexity constant in (31) is one. (More generally, if ϕ is ρ-strongly convex with respect to $\|\cdot\|$, then the factor $1/2$ is replaced with $\rho/2$. By scaling the norm, one can assume $\rho = 1$.) We denote by $\|A\|$ the matrix norm

$$\|A\| = \sup_{x\neq 0} \frac{\|Ax\|_2}{\|x\|} = \sup_{z\neq 0,\, x\neq 0} \frac{z^T A x}{\|z\|_2\|x\|}. \tag{32}$$

The algorithm is summarized as follows. Select starting points $x_0 \in \mathbf{int}(\mathbf{dom}\,\phi) \cap \mathbf{dom}\, f$ and $z_{-1} = z_0$. For $k = 0, 1, \ldots$, repeat the following steps:

$$\begin{aligned} \bar{z}_{k+1} &= z_k + \theta_k(z_k - z_{k-1}) && \text{(33a)}\\ x_{k+1} &= \mathrm{prox}^d_{\tau_k f}(x_k, \tau_k A^T \bar{z}_{k+1}) && \text{(33b)}\\ z_{k+1} &= z_k + \sigma_k(Ax_{k+1} - b). && \text{(33c)} \end{aligned}$$

Step (33b) can be written more explicitly as

$$x_{k+1} = \operatorname*{argmin}_x \left(f(x) + \bar{z}_{k+1}^T Ax + \frac{1}{\tau_k} d(x, x_k)\right). \tag{34}$$

The parameters τ_k, σ_k, θ_k are determined by one of two methods.

- *Constant parameters:* $\theta_k = 1$, $\tau_k = \tau$, $\sigma_k = \sigma$, where

  $$\sqrt{\sigma\tau}\|A\| \leq \delta. \tag{35}$$

  The parameter δ satisfies $0 < \delta \leq 1$. In practice, $\delta = 1$ can be used, but some convergence results will require $\delta < 1$; see Section 3.3.4.

- *Varying parameters.* The parameters τ_k, σ_k, θ_k are determined by a backtracking search. At the start of the algorithm, we set τ_{-1} and σ_{-1} to some positive values. To start the search in iteration k we choose $\bar{\theta}_k \geq 1$. For $i = 0, 1, 2, \ldots$, we set $\theta_k = 2^{-i}\bar{\theta}_k$, $\tau_k = \theta_k\tau_{k-1}$, $\sigma_k = \theta_k\sigma_{k-1}$, and compute $\bar{z}_{k+1}$, x_{k+1}, z_{k+1} using (33). If

  $$(z_{k+1} - \bar{z}_{k+1})^T A(x_{k+1} - x_k) \leq \frac{\delta^2}{\tau_k} d(x_{k+1}, x_k) + \frac{1}{2\sigma_k}\|\bar{z}_{k+1} - z_{k+1}\|_2^2, \tag{36}$$

  we accept the computed iterates $\bar{z}_{k+1}$, x_{k+1}, z_{k+1} and step sizes τ_k, σ_k, and terminate the backtracking search. If (36) does not hold, we increment i and continue the backtracking search.

The constant parameter choice is simple, but it is often overly pessimistic. Moreover it requires an estimate or tight upper bound for $\|A\|$, which is difficult to obtain in large-scale problems. Using a loose bound for $\|A\|$ in (35) may result in unnecessarily small values of τ and σ, and can dramatically slow down the convergence. The definition of $\|A\|$ further depends on the strong convexity constant for the kernel ϕ; see (31) and (32). This quantity is also difficult to estimate for most kernels.

The varying parameters option does not require estimates or bounds on $\|A\|$ or the strong convexity constant of the kernel. It is more expensive because in each backtracking iteration the three updates in (33) are computed. However, the extra cost is well justified in practice. If the line search process takes more than a few backtracking iterations, it indicates that the inequality (36) is much weaker than the conservative step size condition (35), and the algorithm with line search takes much larger steps than would be used by the constant parameter algorithm. In practice, the parameter $\bar{\theta}_k$ can be set to one in most iterations. The backtracking search then first checks whether the previous step sizes τ_{k-1} and σ_{k-1} are acceptable, and decreases them only when needed to satisfy (36). The option of choosing $\bar{\theta}_k > 1$ allows one to occasionally increase the step sizes.

Algorithm (33) is related to several existing algorithms. With constant parameters, it is a special case of the primal–dual algorithm in [22, Algorithm 1], which solves the more general problem (26) and uses generalized distances for the primal and dual variables. Here we take $g(y) = \delta_{\{b\}}$ and use a generalized distance only in the primal space. The line search condition (36) for selecting step sizes does not appear in [22].

With standard proximal operators (for squared Euclidean distances), the primal–dual algorithm of [22] is also known as the *primal–dual hybrid gradient (PDHG) algorithm*, and has been extensively studied as a versatile and efficient algorithm for large-scale convex optimization; see [20, 21, 24, 25, 28, 33, 45, 47, 49, 56, 57] for applications, analysis, and extensions. The line search technique for the primal–dual algorithm proposed by Malitsky and Pock [39] is similar to the one described above, but not identical, even when squared Euclidean distances are used.

The algorithm can also be interpreted as a variation on the Bregman *proximal point algorithm* [19, 26, 32], applied to the optimality conditions

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} \partial f(x) \\ b \end{bmatrix}.$$

In each iteration of the proximal point algorithm the iterates x_{k+1}, z_{k+1} are defined by the inclusion

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x_{k+1} \\ z_{k+1} \end{bmatrix} + \begin{bmatrix} \partial f(x_{k+1}) \\ b \end{bmatrix} + \nabla\phi_{\mathrm{pd}}(x_{k+1}, z_{k+1}) - \nabla\phi_{\mathrm{pd}}(x_k, z_k), \tag{37}$$

where $\phi_{\mathrm{pd}}(x, z)$ is a Bregman kernel. If we choose a kernel of the form

$$\phi_{\mathrm{pd}}(x, z) = \frac{1}{\tau}\phi(x) + \frac{1}{2\sigma}\|z\|_2^2,$$

then (37) reduces to

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x_{k+1} \\ z_{k+1} \end{bmatrix} + \begin{bmatrix} \partial f(x_{k+1}) \\ b \end{bmatrix} + \begin{bmatrix} (\nabla\phi(x_{k+1}) - \nabla\phi(x_k))/\tau \\ (z_{k+1} - z_k)/\sigma \end{bmatrix}.$$

In the generalized proximal operator notation defined of (10) and (11), this condition can be expressed as two equations

$$x_{k+1} = \mathrm{prox}^d_{\tau f}(x_k, \tau A^T z_{k+1}), \qquad z_{k+1} = z_k + \sigma(Ax_{k+1} - b).$$

These two equations are coupled and difficult to solve because x_{k+1} and z_{k+1} each appear on the right-hand side of an equality. The updates (33b) and (33c) are almost identical but replace z_{k+1} with $\bar{z}_{k+1}$ in the primal update. The iterate $\bar{z}_{k+1}$ can therefore be interpreted as a prediction of z_{k+1}. This interpretation also provides some intuition for the step size condition (36). If $\bar{z}_{k+1}$ happens to be equal to z_{k+1}, then (36) imposes no upper bound on the step sizes τ_k and σ_k. This makes sense because when $\bar{z}_{k+1} = z_{k+1}$ the update is equal to the proximal point update, and the convergence theory for the proximal point method does not impose upper bounds on the step size.

He and Yuan [33] have given an interesting interpretation of the primal–dual algorithm of [20] as a "pre-conditioned" proximal point algorithm. For the algorithm considered here, their interpretation corresponds to choosing

$$\phi_{\mathrm{pd}}(x, z) = \frac{1}{\tau}\phi(x) + \frac{1}{2\sigma}\|z\|_2^2 + z^T A x \tag{38}$$

as the generalized distance in (37). It can be shown that under the strong convexity assumptions for ϕ mentioned at the beginning of the section, the function (38) is convex if $\sqrt{\sigma\tau}\|A\| \leq 1$. With this choice of Bregman kernel, the inclusion (37) reduces to

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x_k \\ 2z_{k+1} - z_k \end{bmatrix} + \begin{bmatrix} \partial f(x_{k+1}) \\ b \end{bmatrix} + \begin{bmatrix} (\nabla\phi(x_{k+1}) - \nabla\phi(x_k))/\tau \\ (z_{k+1} - z_k)/\sigma \end{bmatrix},$$

which can be written as

$$z_{k+1} = z_k + \sigma(Ax_k - b), \qquad x_{k+1} = \mathrm{prox}^d_{\tau f}(x_k, \tau A^T(2z_{k+1} - z_k)).$$

Except for the indexing of the iterates, this is identical to (33) with constant step sizes ($\theta_k = 1$, $\tau_k = \tau$, $\sigma_k = \sigma$).

3.3 Convergence analysis

In this section we analyze the convergence of the algorithm following the ideas in [22, 39, 49]. The main result is an ergodic convergence rate, given in equation (49).

3.3.1 Algorithm parameters

We first prove two facts about the step sizes in the two versions of the algorithm.

Constant parameters If $\theta_k = 1$, $\tau_k = \tau$, $\sigma_k = \sigma$, where τ and σ satisfy (35), then the iterates $\bar{z}_{k+1}$, x_{k+1}, z_{k+1} satisfy (36).

Proof. We use the definition of the matrix norm $\|A\|$, the arithmetic–geometric mean inequality,

and strong convexity of the Bregman kernel:

$$\begin{aligned}
(z_{k+1}-\bar{z}_{k+1})^T A(x_{k+1}-x_k) &\leq \|A\|\|x_{k+1}-x_k\|\|z_{k+1}-\bar{z}_{k+1}\|_2 \\
&= \frac{\sqrt{\sigma_k\tau_k}\|A\|}{\delta}\left(\frac{\delta^2\|x_{k+1}-x_k\|^2}{\tau_k}\frac{\|z_{k+1}-\bar{z}_{k+1}\|_2^2}{\sigma_k})\right)^{1/2} \\
&\leq \frac{\sqrt{\sigma_k\tau_k}\|A\|}{\delta}\left(\frac{\delta^2\|x_{k+1}-x_k\|^2}{2\tau_k}+\frac{\|z_{k+1}-\bar{z}_{k+1}\|_2^2}{2\sigma_k}\right) \\
&\leq \frac{\sqrt{\sigma_k\tau_k}\|A\|}{\delta}\left(\frac{\delta^2 d(x_{k+1},x_k)}{\tau_k}+\frac{\|z_{k+1}-\bar{z}_{k+1}\|_2^2}{2\sigma_k}\right) \\
&\leq \frac{\delta^2 d(x_{k+1},x_k)}{\tau_k}+\frac{\|\bar{z}_{k+1}-z_{k+1}\|_2^2}{2\sigma_k}.
\end{aligned}$$

The last inequality follows from (35). □

The result implies that we can restrict the analysis to the algorithm with varying parameters. The constant parameter variant is a special case with $\bar{\theta}_k = 1$, $\tau_{-1} = \tau$, and $\sigma_{-1} = \sigma$.

Varying parameters In the varying parameter variant of the algorithm the step sizes are bounded below by

$$\tau_k \geq \tau_{\min} \triangleq \min\{\tau_{-1}, \frac{\delta}{2\sqrt{\beta}\|A\|}\}, \qquad \sigma_k \geq \sigma_{\min} \triangleq \beta\tau_{\min}, \tag{39}$$

where $\beta = \sigma_{-1}/\tau_{-1}$.

Proof. We proved in the previous paragraph that the exit condition (36) in the backtracking search certainly holds if

$$\sqrt{\sigma_k\tau_k}\|A\| \leq \delta.$$

From this observation one can use induction to prove the lower bounds (39). Suppose $\tau_{k-1} \geq \tau_{\min}$ and $\sigma_{k-1} \geq \sigma_{\min}$. This holds at $k = 0$ by definition of $\tau_{\min}$ and $\sigma_{\min}$. The first value of θ_k tested in the search is $\theta_k = \bar{\theta}_k \geq 1$. If this value is accepted, then

$$\tau_k = \bar{\theta}_k\tau_{k-1} \geq \tau_{k-1} \geq \tau_{\min}, \qquad \sigma_k = \bar{\theta}_k\sigma_{k-1} \geq \sigma_{k-1} \geq \sigma_{\min}.$$

If $\theta_k = \bar{\theta}_k$ is rejected, one or more backtracking steps are taken. Denote by $\tilde{\theta}_k$ the last rejected value. Then $\tilde{\theta}_k\sqrt{\sigma_{k-1}\tau_{k-1}}\|A\| > \delta$, and the accepted θ_k satisfies

$$\theta_k = \frac{\tilde{\theta}_k}{2} > \frac{\delta}{2\sqrt{\sigma_{k-1}\tau_{k-1}}\|A\|} = \frac{\delta}{2\tau_{k-1}\sqrt{\beta}\|A\|}.$$

Therefore

$$\tau_k = \theta_k\tau_{k-1} > \frac{\delta}{2\sqrt{\beta}\|A\|} \geq \tau_{\min}, \qquad \sigma_k = \beta\tau_k \geq \beta\tau_{\min}.$$

□

3.3.2 Analysis of one iteration

We now analyze the progress in one iteration of the varying parameter variant of algorithm (33).

Duality gap For $i \geq 1$, the iterates x_i, z_i, $\bar{z}_i$ satisfy

$$\begin{aligned} L(x_i, z) - L(x, \bar{z}_i) &\leq \frac{1}{\tau_{i-1}}\left(d(x, x_{i-1}) - d(x, x_i) - (1-\delta^2)d(x_i, x_{i-1})\right) \\ &\quad + \frac{1}{2\sigma_{i-1}}\left(\|z - z_{i-1}\|_2^2 - \|z - z_i\|_2^2 - \|\bar{z}_i - z_{i-1}\|_2^2\right) \end{aligned} \tag{40}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi$ and all z.

Proof. The second step (33b) defines x_{k+1} as the minimizer of

$$f(x) + \bar{z}_{k+1}^T Ax + \frac{1}{\tau_k} d(x, x_k) = f(x) + \bar{z}_{k+1}^T Ax + \frac{1}{\tau_k}\big(\phi(x) - \phi(x_k) - \langle \nabla\phi(x_k), x - x_k\rangle\big).$$

By assumption the solution is uniquely defined and in the interior of $\mathbf{dom}\, \phi$. Therefore x_{k+1} satisfies the optimality condition

$$\frac{1}{\tau_k}(\nabla\phi(x_k) - \nabla\phi(x_{k+1})) - A^T\bar{z}_{k+1} \in \partial f(x_{k+1}).$$

Equivalently, the following holds for all $x \in \mathbf{dom}\, \phi \cap \mathbf{dom}\, f$:

$$\begin{aligned} f(x) - f(x_{k+1}) &\geq -\bar{z}_{k+1}^T A(x - x_{k+1}) + \frac{1}{\tau_k}\langle \nabla\phi(x_k) - \nabla\phi(x_{k+1}), x - x_{k+1}\rangle \\ &= -\bar{z}_{k+1}^T A(x - x_{k+1}) - \frac{1}{\tau_k}(d(x, x_k) - d(x, x_{k+1}) - d(x_{k+1}, x_k)). \end{aligned} \tag{41}$$

(The triangle identity (8) is used on the second line.) The dual update (33c) implies that

$$(z - z_{k+1})^T(Ax_{k+1} - b) = \frac{1}{\sigma_k}(z - z_{k+1})^T(z_{k+1} - z_k) \quad \text{for all } z. \tag{42}$$

This equality at $k = i - 1$ is

$$\begin{aligned} (z - z_i)^T(Ax_i - b) &= \frac{1}{\sigma_{i-1}}(z - z_i)^T(z_i - z_{i-1}) \\ &= \frac{1}{2\sigma_{i-1}}\left(\|z - z_{i-1}\|_2^2 - \|z - z_i\|_2^2 - \|z_i - z_{i-1}\|_2^2\right). \end{aligned} \tag{43}$$

The equality (42) at $k = i - 2$ is

$$\begin{aligned} (z - z_{i-1})^T(Ax_{i-1} - b) &= \frac{1}{\sigma_{i-2}}(z - z_{i-1})^T(z_{i-1} - z_{i-2}) \\ &= \frac{\theta_{i-1}}{\sigma_{i-1}}(z - z_{i-1})^T(z_{i-1} - z_{i-2}) \\ &= \frac{1}{\sigma_{i-1}}(z - z_{i-1})^T(\bar{z}_i - z_{i-1}). \end{aligned}$$

We evaluate this at $z = z_i$ and add it to the equality at $z = z_{i-2}$ multiplied by θ_{i-1}:

$$\begin{aligned} (z_i - \bar{z}_i)^T(Ax_{i-1} - b) &= \frac{1}{\sigma_{i-1}}(z_i - \bar{z}_i)^T(\bar{z}_i - z_{i-1}) \\ &= \frac{1}{2\sigma_{i-1}}\left(\|z_i - z_{i-1}\|_2^2 - \|z_i - \bar{z}_i\|_2^2 - \|\bar{z}_i - z_{i-1}\|_2^2\right). \end{aligned} \tag{44}$$

Now we combine (41) for $k = i-1$, with (43) and (44). For $i \geq 1$,

$$\begin{aligned}
&L(x_i, z) - L(x, \bar{z}_i) \\
&\quad = \quad f(x_i) + z^T(Ax_i - b) - f(x) - \bar{z}_i^T(Ax - b) \\
&\quad \leq \quad \frac{1}{\tau_{i-1}}\left(d(x, x_{i-1}) - d(x, x_i) - d(x_i, x_{i-1})\right) + \bar{z}_i^T A(x - x_i) + z^T(Ax_i - b) - \bar{z}_i^T(Ax - b) \\
&\quad = \quad \frac{1}{\tau_{i-1}}\left(d(x, x_{i-1}) - d(x, x_i) - d(x_i, x_{i-1})\right) + (z - \bar{z}_i)^T(Ax_i - b) \\
&\quad = \quad \frac{1}{\tau_{i-1}}\left(d(x, x_{i-1}) - d(x, x_i) - d(x_i, x_{i-1})\right) + (z_i - \bar{z}_i)^T A(x_i - x_{i-1}) \\
&\qquad + (z - z_i)^T(Ax_i - b) + (z_i - \bar{z}_i)^T(Ax_{i-1} - b) \\
&\quad = \quad \frac{1}{\tau_{i-1}}(d(x, x_{i-1}) - d(x, x_i) - d(x_i, x_{i-1})) + (z_i - \bar{z}_i)^T A(x_i - x_{i-1}) \\
&\qquad + \frac{1}{2\sigma_{i-1}}(\|z - z_{i-1}\|_2^2 - \|z - z_i\|_2^2 - \|\bar{z}_i - z_{i-1}\|_2^2 - \|\bar{z}_i - z_i\|_2^2). \qquad (45)
\end{aligned}$$

The first inequality follows from (41). In the last step we substitute (43) and (44). Next we note that the line search exit condition (36) implies that

$$(z_i - \bar{z}_i)^T A(x_i - x_{i-1}) \leq \frac{\delta^2}{\tau_{i-1}} d(x_i, x_{i-1}) + \frac{1}{2\sigma_{i-1}}\|\bar{z}_i - z_i\|_2^2.$$

Substituting this in (45) gives the bound (40). □

Monotonicity properties Suppose $x^\star \in \mathbf{dom}\,\phi$, and $x^\star$, $z^\star$ satisfy the saddle point property (29). Then

$$d(x^\star, x_i) + \frac{1}{2\beta}\|z^\star - z_i\|_2^2 \leq d(x^\star, x_{i-1}) + \frac{1}{2\beta}\|z^\star - z_{i-1}\|_2^2 \qquad (46)$$

where $\beta = \sigma_{-1}/\tau_{-1}$. Moreover

$$\sum_{i=1}^{k}\left((1-\delta^2)d(x_i, x_{i-1}) + \frac{1}{2\beta}\|\bar{z}_i - z_{i-1}\|_2^2\right) \leq d(x^\star, x_0) + \frac{1}{2\beta}\|z^\star - \bar{z}_0\|_2^2. \qquad (47)$$

These inequalities hold for any value $\delta \in (0, 1]$ in the line search condition (36). The second inequality implies that $\bar{z}_i - z_{i-1} \to 0$. If $\delta < 1$ it also implies that $d(x_i, x_{i-1}) \to 0$ and, by the strong convexity assumption on ϕ, that $x_i - x_{i-1} \to 0$.

Proof. We substitute $x = x^\star$, $z = z^\star$ in (40) and note that $L(x_i, z^\star) - L(x^\star, \bar{z}_i) \geq 0$ (from the saddle-point property (29)):

$$\begin{aligned}
0 \quad &\leq \quad L(x_i, z^\star) - L(x^\star, \bar{z}_i) \\
&\leq \quad \frac{1}{\tau_{i-1}}(d(x^\star, x_{i-1}) - d(x^\star, x_i) - (1-\delta^2)d(x_i, x_{i-1})) \\
&\qquad + \frac{1}{2\sigma_{i-1}}\left(\|z^\star - z_{i-1}\|_2^2 - \|z^\star - z_i\|_2^2 - \|\bar{z}_i - z_{i-1}\|_2^2\right).
\end{aligned}$$

With $\beta = \sigma_{i-1}/\tau_{i-1} = \sigma_{-1}/\tau_{-1}$, this gives the inequality

$$(1-\delta^2)d(x_i, x_{i-1}) + \frac{1}{2\beta}\|\bar{z}_i - z_{i-1}\|_2^2 \leq d(x^\star, x_{i-1}) - d(x^\star, x_i) + \frac{1}{2\beta}(\|z^\star - z_{i-1}\|_2^2 - \|z^\star - z_i\|_2^2).$$

Since the left-hand side is nonnegative, the inequality (46) follows. Summing from $i = 1$ to k gives (47). □

3.3.3 Ergodic convergence

We define averaged primal and dual sequences

$$x_k^{\text{avg}} = \frac{1}{\sum_{i=1}^k \tau_{i-1}} \sum_{i=1}^k \tau_{i-1} x_i, \qquad z_k^{\text{avg}} = \frac{1}{\sum_{i=1}^k \tau_{i-1}} \sum_{i=1}^k \tau_{i-1} \bar{z}_i.$$

We first show that the averaged sequences satisfy

$$L(x_k^{\text{avg}}, z) - L(x, z_k^{\text{avg}}) \leq \frac{1}{\sum_{i=1}^k \tau_{i-1}} (d(x, x_0) + \frac{1}{2\beta}\|z - z_0\|_2^2) \tag{48}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi$ and all z. This holds for every choice for $\delta \in (0, 1]$ in (36).

Proof. From (40),

$$L(x_i, z) - L(x, \bar{z}_i) \leq \frac{1}{\tau_{i-1}} \left(d(x, x_{i-1}) - d(x, x_i) + \frac{1}{2\beta}\|z - z_{i-1}\|_2^2 - \frac{1}{2\beta}\|z - z_i\|_2^2 \right).$$

Since L is convex in x and affine in z,

$$\begin{aligned}
(\sum_{i=1}^k \tau_{i-1})(L(x_k^{\text{avg}}, z) - L(x, z_k^{\text{avg}})) &\leq \sum_{i=1}^k \tau_{i-1}(L(x_i, z) - L(x, \bar{z}_i)) \\
&\leq d(x, x_0) - d(x, x_k) + \frac{1}{2\beta}(\|z - z_0\|_2^2 - \|z - z_k\|_2^2) \\
&\leq d(x, x_0) + \frac{1}{2\beta}\|z - z_0\|_2^2.
\end{aligned}$$

Dividing by $\sum_{i=1}^k \tau_{i-1}$ gives (48). □

If we substitute in (48) an optimal $x = x^\star$ (which satisfies $Ax^\star = b$), we obtain that

$$f(x_k^{\text{avg}}) + z^T(Ax_k^{\text{avg}} - b) - f(x^\star) \leq \frac{1}{\sum_{i=1}^k \tau_{i-1}} (d(x^\star, x_0) + \frac{1}{2\beta}\|z - z_0\|_2^2)$$

for all z. Maximizing both sides over z subject to $\|z\|_2 \leq \gamma$ shows that

$$\begin{aligned}
f(x_k^{\text{avg}}) + \gamma\|Ax_k^{\text{avg}} - b\|_2 - f(x^\star) &\leq \frac{1}{\sum_{i=1}^k \tau_{i-1}} \left(d(x^\star, x_0) + \frac{1}{2\beta} \sup_{\|z\|_2 \leq \gamma} \|z - z_0\|_2^2 \right) \\
&= \frac{1}{\sum_{i=1}^k \tau_{i-1}} \left(d(x^\star, x_0) + \frac{1}{2\beta}(\gamma + \|z_0\|_2)^2 \right).
\end{aligned} \tag{49}$$

The first two terms on the left-hand side form the merit function (30). For $\gamma > \|z^\star\|_2$, the penalty function in the merit function is exact, so $f(x) + \gamma\|Ax - b\|_2 - f(x^\star) \geq 0$ with equality only if x is optimal. (The use of an exact penalty function to express a convergence result is inspired by [49, page 287].) Since $\tau_i \geq \tau_{\min}$, the inequality shows that the merit function decreases as $O(1/k)$.

3.3.4 Convergence of the iterates

We now make two additional assumptions about the Bregman kernel ϕ [19].

1. For fixed x, the sublevel sets $\{y \mid d(x,y) \leq \alpha\}$ are closed. In other words, the distance $d(x,y)$ is a closed function of y.

2. If $y_k \in \mathbf{int}\,(\mathbf{dom}\,\phi)$ converges to $x \in \mathbf{dom}\,\phi$, then $d(x,y_k) \to 0$.

These two assumptions are not restrictive, and in particular, they are satisfied by the logarithmic barrier ϕ (12). We also make the (minor) assumptions that $\delta < 1$ in (36) and that θ_k is bounded above (which is easily satisfied, since the user chooses $\bar{\theta}_k$). With these additional assumptions it can be shown that the sequences x_k, z_k converge to optimal solutions.

Proof. The inequality (46) and strong convexity of ϕ show that the sequences x_k, z_k are bounded. Let (x_{k_i}, z_{k_i}) be a convergent subsequence with limit $(\hat{x}, \hat{z})$. With $\delta < 1$, (47) shows that $d(x_{k_i+1}, x_{k_i})$ converges to zero. By strong convexity of the kernel, $x_{k_i+1} - x_{k_i} \to 0$ and therefore the subsequence x_{k_i+1} also converges to $\hat{x}$. Since $z_{k_i+1} - z_{k_i} \to 0$, the subsequence z_{k_i+1} converges to $\hat{z}$. Since θ_k is bounded above, $\bar{z}_{k_{i+1}} = z_{k_i} + \theta_k(z_{k_i} - z_{k_i-1})$ also converges to $\hat{z}$.

The dual update (33c) can be written as

$$Ax_{k_i+1} - b = \frac{1}{\sigma_{k_i}}(z_{k_i+1} - z_{k_i}). \tag{50}$$

Since $z_{k_i+1} - z_{k_i} \to 0$ and $\sigma_{k_i} \geq \sigma_{\min}$, the left-hand side converges to zero, so $A\hat{x} = b$.

From (46), $d(x^\star, x_{k_i})$ is bounded above. Since the sublevel sets $\{y \mid d(x^\star, y) \leq \alpha\}$ are closed subsets of $\mathbf{int}\,(\mathbf{dom}\,\phi)$, the limit $\hat{x}$ is in $\mathbf{int}\,(\mathbf{dom}\,\phi)$. The left-hand side of the optimality condition

$$\frac{1}{\tau_{k_i}}(\nabla\phi(x_{k_i}) - \nabla\phi(x_{k_i+1})) - A^T\bar{z}_{k_i+1} \in \partial f(x_{k_i+1}) \tag{51}$$

converges to $-A^T\hat{z}$, because $\tau_k \geq \tau_{\min}$ and $\nabla\phi$ is continuous on $\mathbf{int}\,(\mathbf{dom}\,\phi)$. By maximal monotonicity of ∂f, this implies that $-A^T\hat{z} \in \partial f(\hat{x})$ (see [17, page 27] [53, lemma 3.2]). We conclude that $\hat{x}$, $\hat{z}$ satisfy the optimality conditions $A\hat{x} = b$ and $-A^T\hat{z} \in \partial f(\hat{x})$.

To show that the entire sequence converges, we substitute $x = \hat{x}$, $z = \hat{z}$ in (40):

$$L(x_k, \hat{z}) - L(\hat{x}, \bar{z}_k) \leq \frac{1}{\tau_{k-1}}(d(\hat{x}, x_{k-1}) - d(\hat{x}, x_k)) + \frac{1}{2\beta\tau_{k-1}}(\|\hat{z} - z_{k-1}\|_2^2 - \|\hat{z} - z_k\|_2^2).$$

The left-hand side is nonnegative by the saddle point property (29). Therefore

$$d(\hat{x}, x_k) + \frac{1}{2\beta}\|\hat{z} - z_k\|_2^2 \leq d(\hat{x}, x_{k-1}) + \frac{1}{2\beta}\|\hat{z} - z_{k-1}\|_2^2$$

for all k. This shows that

$$d(\hat{x}, x_k) + \frac{1}{2\beta}\|\hat{z} - z_k\|_2^2 \leq d(\hat{x}, x_{k_i}) + \frac{1}{2\beta}\|\hat{z} - z_{k_i}\|_2^2$$

for all $k \geq k_i$. By the second additional kernel property mentioned above, the right-hand side converges to zero. Therefore $d(\hat{x}, x_k) \to 0$ and $z_k \to \hat{z}$. If $d(\hat{x}, x_k) \to 0$, then the strong convexity property of the kernel implies that $x_k \to \hat{x}$. □

4 Numerical experiments

In this section we evaluate the performance of algorithm (7), the Bregman PDHG algorithm (33) applied to the centering problem (5). The numerical results illustrate that the cost for evaluating the Bregman proximal operator (17) is comparable to the cost of a sparse Cholesky factorization with sparsity pattern E. This prox-evaluation dominates the computational cost in each iteration of (7), since $\mathcal{A}$ and $\mathcal{A}^*$ are usually easy to evaluate for large-scale problems with sparse or other types of structure. In particular, the proposed method does not need to solve linear equations involving $\mathcal{A}$ or $\mathcal{A}^*$, an important advantage over ADMM and interior-point methods.

In this section we consider the centering problem for two sets of sparse SDPs, the maximum cut problem and the graph partitioning problem. The experiments are carried out in Python 3.6 on a laptop with an Intel Core i5 2.4GHz CPU and 8GB RAM. The Python library for chordal matrix computations CHOMPACK [4] is used to compute chordal extensions (with the AMD reordering [1]), sparse Cholesky factorizations, the primal barrier ϕ, and the gradient and directional second derivative of the dual barrier ϕ_*. Other sparse matrix computations are implemented using CVXOPT [2].

In the experiments, we terminate the iteration (33) when the relative primal and dual residuals are less than 10^{-6}. These two stopping conditions are sufficient for our algorithm, as suggested by the convergence proof, in particular, equations (50) and (51). The two residuals are defined as

$$\text{primal residual} = \frac{\|z_k - z_{k-1}\|_2}{\sigma_k \max\{1, \|z_k\|_\infty\}}, \qquad \text{dual residual} = \frac{\|\nabla\phi(X_k) - \nabla\phi(X_{k-1})\|_2}{\tau_k \max\{1, \|X_k\|_{\max}\}},$$

where $\|Y\|_{\max} = \max_{i,j} |Y_{ij}|$.

4.1 Maximum cut problem

Given an undirected graph $G = (V, E)$, the maximum cut problem is to partition the set of vertices into two sets in order to maximize the total number of edges between the two sets. (If every edge $\{i, j\} \in E$ is associated with a nonnegative weight w_{ij}, then the maximum cut problem is to maximize the total weight of the edges between the two sets.) One can show that the maximum cut problem can be represented as a binary quadratic optimization problem

$$\begin{array}{ll} \text{maximize} & (1/4)x^T L x \\ \text{subject to} & x \in \{\pm 1\}^n, \end{array}$$

where $L \in \mathbf{S}^n$ is the Laplacian of an undirected graph $G = (V, E)$ with vertices $V = \{1, 2, \ldots, n\}$. The SDP relaxation of the maximum cut problem is

$$\begin{array}{ll} \text{maximize} & (1/4)\,\mathbf{tr}(LX) \\ \text{subject to} & \mathbf{diag}(X) = \mathbf{1} \\ & X \succeq 0, \end{array} \tag{52}$$

with variable $X \in \mathbf{S}^n$. The operator $\mathbf{diag}\colon \mathbf{S}^n \to \mathbf{R}^n$ returns the diagonal elements of the input matrix as a vector: $\mathbf{diag}(X) = (X_{11}, X_{22}, \ldots, X_{nn})$. If moderate accuracy is allowed, we can solve the centering problem of the SDP relaxation

$$\begin{array}{ll} \text{minimize} & -(1/4)\,\mathbf{tr}(LX) + \mu\phi(X) \\ \text{subject to} & \mathbf{diag}(X) = \mathbf{1} \\ & X \in \Pi_{E'}(\mathbf{S}^n_+) \end{array} \tag{53}$$

	n	$p^\star_{\rm sdp}$	$p^\star_{\rm sdp} - \frac{1}{4}\,\mathbf{tr}(LX)$	primal residual	dual residual
maxG51	1000	4.0039×10^3	3.12×10^{-4}	2.24×10^{-7}	6.43×10^{-8}
maxG32	2000	1.5676×10^3	6.95×10^{-4}	6.48×10^{-7}	2.23×10^{-7}
maxG55	5000	1.1039×10^4	1.02×10^{-4}	5.32×10^{-7}	7.13×10^{-7}
maxG60	7000	1.5222×10^4	9.91×10^{-5}	1.21×10^{-7}	2.33×10^{-7}

Table 1: Results for four instances of the MAXCUT problem from SDPLIB [15]. Column 3 is the optimal value computed by MOSEK. Column 4 is the difference with the optimal value of the centering problem computed by algorithm (33). The last two columns give the primal and dual residuals in the computed solution.

with optimization variable $X \in \mathbf{S}^n_{E'}$ where E' is a chordal extension of E. Note that $\mathbf{tr}(X) = n$ for all feasible X. The centering problem has the form of (5) with

$$C = -\frac{1}{4}L, \qquad N = \frac{1}{n}I, \qquad \mathcal{A}(X) = \mathbf{diag}(X).$$

The Lagrangian of (53) is in the form of (27) where f is defined in (6), and z is the Lagrange multiplier associated with the equality constraint $\mathbf{diag}(X) = \mathbf{1}$. Thus we have

$$\frac{1}{4}\,\mathbf{tr}(LX^\star) \leq p^\star_{\rm sdp} \leq \mathbf{1}^T z^\star, \qquad -\frac{1}{4}\,\mathbf{tr}(LX^\star) + \mathbf{1}^T z^\star = \mu n, \tag{54}$$

where $X^\star$ and $z^\star$ are the primal and dual optimal solutions of the centering problem (53), and $p^\star_{\rm sdp}$ is the optimal value of the SDP (52).

Numerical results We first collect four MAXCUT problems of moderate size from SDPLIB [15]. The SDP relaxation (52) is solved using MOSEK [41] and the optimal value computed by MOSEK is denoted by $p^\star_{\rm sdp}$. (Note that the source file for the graph maxcutG55 was unfortunately incorrectly converted into SDPA sparse format. Thus the objective value for the maxG55 problem obtained from the original data file is 1.1039×10^4 instead of 9.9992×10^3 as reported in SDPLIB.)

In (53), we set $\mu = 0.001/n$, and report in column 4 of Table 1 the difference between $p^\star_{\rm sdp}$ and the cost function $(1/4)\,\mathbf{tr}(LX)$ at the suboptimal solution returned by the algorithm. The last two columns of Table 1 give the relative primal and dual residuals. These results show that the proposed algorithm is able to solve the centering SDP (53) with the desired accuracy. A comparison of the third and fourth columns of Table 1 confirms (54), *i.e.*, the objective value of the SDP at X is within $\mu n = 10^{-3}$ of the optimal value. Considering the values of $p^\star_{\rm sdp}$, we see that the computed points on the central path are close to the optimal solutions of the SDPs.

To test the scalability of algorithm (33), we add four larger graphs from the SuiteSparse collection [36]. In Table 2 we report the time per Cholesky factorization, the number of Newton steps per iteration, the time per PDHG iteration, and the number of iterations in the primal–dual (PDHG) algorithm for the eight test problems. As can be seen from the table, the number of Newton iterations per prox-evaluation remains small even when the size of the problem increases. Also, we observe that the time per PDHG iteration is roughly the cost of a sparse Cholesky factorization times the number of Newton steps. This means that the backtracking in Newton's method does not cause a significant overhead. Since the evaluations of $\mathcal{A}$ and $\mathcal{A}^*$ in this problem are very cheap, the cost per prox-evaluation is the dominant term in the per-iteration complexity.

	n	time per Cholesky factorization	Newton steps per iteration	time per PDHG iteration	PDHG iterations
maxG51	1000	0.05	2.45	0.12	267
maxG32	2000	0.12	1.56	0.18	240
maxG55	5000	0.29	2.10	0.58	249
maxG60	7000	0.60	2.55	1.22	279
barth4	6019	0.42	3.57	1.55	346
tuma2	12992	0.48	4.36	1.89	375
biplane-9	21701	0.95	2.58	2.12	287
c-67	57975	0.76	3.58	3.56	378

Table 2: The four MAXCUT problems from SDPLIB plus four larger graphs from the SuiteSparse collection [36]. The last column ('PDHG iterations') gives the number of iterations in the primal–dual algorithm. Columns 3–5 describe the complexity of one iteration of the algorithm. The CPU time is measured in seconds.

4.2 Graph partitioning

The problem of partitioning the vertices of a graph $G = (V, E)$ in two subsets of equal size (here we assume an even number of vertices), while minimizing the number of edges between the two subsets, can be expressed as

$$\begin{array}{ll} \text{minimize} & (1/4)x^T L x \\ \text{subject to} & \mathbf{1}^T x = 0 \\ & x \in \{-1, 1\}^n, \end{array}$$

where L is the graph Laplacian. The ith entry of the n-vector x indicates the set that vertex i is assigned to. To obtain an SDP relaxation we introduce a matrix variable $Y = xx^T$ and write the problem in the equivalent form

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(LY) \\ \text{subject to} & \mathbf{1}^T Y \mathbf{1} = 0 \\ & \mathbf{diag}(Y) = \mathbf{1} \\ & Y = xx^T, \end{array}$$

and then relax the constraint $Y = xx^T$ as $Y \succeq 0$. This gives the SDP

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(LY) \\ \text{subject to} & \mathbf{1}^T Y \mathbf{1} = 0 \\ & \mathbf{diag}(Y) = \mathbf{1} \\ & Y \succeq 0. \end{array} \tag{55}$$

The dual SDP is

$$\begin{array}{ll} \text{maximize} & \mathbf{1}^T z \\ \text{subject to} & \mathbf{diag}(z) + \xi \mathbf{1}\mathbf{1}^T \preceq (1/4)L, \end{array}$$

with variables $\xi \in \mathbf{R}$ and $z \in \mathbf{R}^n$.

The aggregate sparsity pattern of the SDP (55) is completely dense, because the equality constraint $\mathbf{1}^T Y \mathbf{1} = 0$ has a coefficient matrix of all ones. We therefore eliminate the dense constraint

using the technique described in [30, page 668]. Let P be the $n \times (n-1)$ matrix

$$P = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -1 & 1 & \cdots & 0 & 0 \\ 0 & -1 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & -1 & 1 \\ 0 & 0 & \cdots & 0 & -1 \end{bmatrix}.$$

The columns of P form a sparse basis for the orthogonal complement of the multiples of the vector $\mathbf{1}$. Suppose Y is feasible in (55) and define

$$\begin{bmatrix} X & u \\ u^T & v \end{bmatrix} = \begin{bmatrix} P & \mathbf{1} \end{bmatrix}^{-1} Y \begin{bmatrix} P & \mathbf{1} \end{bmatrix}^{-T}. \tag{56}$$

From $\mathbf{1}^T Y \mathbf{1} = 0$, we see that

$$0 = \mathbf{1}^T Y \mathbf{1} = \mathbf{1}^T \begin{bmatrix} P & \mathbf{1} \end{bmatrix} \begin{bmatrix} X & u \\ u^T & v \end{bmatrix} \begin{bmatrix} P & \mathbf{1} \end{bmatrix}^T \mathbf{1} = n^2 v,$$

and therefore $v = 0$. Since the matrix (56) is positive semidefinite, we also have $u = 0$. Hence every feasible Y can be expressed as $Y = PXP^T$, with $X \succeq 0$. If we make this substitution in (55) we obtain

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(P^T L P X) \\ \text{subject to} & \mathbf{diag}(PXP^T) = \mathbf{1} \\ & X \succeq 0. \end{array}$$

The $(n-1) \times (n-1)$ matrix $P^T L P$ has elements

$$(P^T L P)_{ij} = \begin{cases} L_{ii} - 2L_{i,i+1} + L_{i+1,i+1} & i = j \\ L_{ij} - L_{i+1,j} - L_{i,j+1} + L_{i+1,j+1} & i \neq j. \end{cases}$$

Thus the sparsity pattern E' of the matrix $P^T L P$ is denser than E, *i.e.*, $E \subseteq E'$. The n constraints $\mathbf{diag}(PXP^T) = \mathbf{1}$ reduce to

$$X_{11} = 1, \qquad X_{i-1,i-1} + X_{ii} - 2X_{i,i-1} = 1, \quad i = 2, \ldots, n-1, \qquad X_{n-1,n-1} = 1.$$

To apply algorithm (33), we first rewrite the graph partitioning problem as

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(P^T L P X) \\ \text{subject to} & \mathbf{diag}(PXP^T) = \mathbf{1} \\ & X \in \Pi_{E''}(\mathbf{S}_+^{n-1}) \end{array} \tag{57}$$

where E'' is a chordal extension of the aggregate sparsity pattern E'. Note that $\mathbf{tr}(P^T P X) = n-1$ for all feasible X. The centering problem for this sparse SDP is of the form (5) with

$$C = \frac{1}{4} P^T L P, \qquad N = \frac{1}{n-1} P^T P, \qquad \mathcal{A}(X) = \mathbf{diag}(PXP^T), \qquad \mathcal{A}^*(y) = P^T\,\mathbf{diag}(y) P.$$

	n	$p^\star_{\rm sdp}$	$p^\star_{\rm sdp} - \frac{1}{4}\,\mathbf{tr}(P^T L P X)$	primal residual	dual residual
gpp100	100	-44.943551	3.78×10^{-4}	3.24×10^{-7}	8.34×10^{-7}
gpp124-1	124	-7.3430761	4.02×10^{-4}	3.86×10^{-8}	7.45×10^{-8}
gpp250-1	250	-45.444917	8.23×10^{-4}	1.28×10^{-7}	8.39×10^{-7}
gpp500-1	500	-25.320544	5.17×10^{-4}	7.42×10^{-8}	7.12×10^{-7}

Table 3: Results for four graph partitioning problems from SDPLIB. Column 3 is the optimal value computed by MOSEK. Column 4 is the difference with the optimal value of the centering problem computed by algorithm (33). The last two columns give the primal and dual residuals in the computed solution.

	n	time per Cholesky factorization	Newton steps per iteration	time per PDHG iteration	PDHG iterations
gpp100	100	0.01	2.43	0.02	305
gpp124-1	124	0.01	2.00	0.02	392
gpp250-1	250	0.01	2.65	0.03	365
gpp500-1	500	0.02	3.01	0.07	394
delaunay_n10	1024	0.37	4.36	1.76	403
delaunay_n11	2048	0.48	4.70	2.54	420
delaunay_n12	4096	0.60	4.43	3.05	367
delaunay_n13	8192	1.02	4.42	4.98	375

Table 4: The four graph partitioning problems from SDPLIB plus four larger graphs from the SuiteSparse collection. The last column gives the number of iterations in the primal–dual algorithm. Columns 3–5 describe the complexity of one iteration of the algorithm. The CPU time is measured in seconds.

Numerical results Table 3 shows the numerical results for four problems from SDPLIB [15]. The SDP relaxation (55) is solved by MOSEK and its optimal value is denoted by $p^\star_{\rm sdp}$. In solving (57), we set $\mu = 0.001/n$, and report in Table 3 the value $(1/4)\,\mathbf{tr}(P^T L P X)$, where X is the solution returned by the algorithm (33). As in the first experiment, the numerical results show that the algorithm is able to solve the centering SDP (57) with desired accuracy.

In addition, we test the algorithm for four additional graphs from the SuiteSparse collection [36]. Table 4 reports the time per Cholesky factorization, the number of Newton steps per iteration, the time per PDHG iteration, and the number of iterations in the primal–dual algorithm. The same observations as in Section 4.1 apply: the number of Newton steps remains moderate as the size of the problem increases, and the cost per iteration is roughly linear in the cost of a Cholesky factorization.

5 Conclusions

We presented a Bregman proximal algorithm for the centering problem in sparse semidefinite programming. The Bregman distance used in the proximal operator is generated by the logarithmic

barrier function for the cone of sparse matrices with a positive semidefinite completion. With this choice of Bregman distance, the per-iteration complexity of the algorithm is dominated by the cost of a Cholesky factorization with the aggregate sparsity pattern of the SDP, plus the cost of evaluating the linear mapping in the constraints and its adjoint.

The proximal algorithm we used is based on the primal–dual method proposed by Chambolle and Pock [22]. An important addition to the algorithm is a new procedure for selecting the primal and dual step sizes, without knowledge of the norm of the linear mapping or the strong convexity of the Bregman kernel. In the current implementation the ratio of the primal and dual step sizes is kept fixed throughout the iteration. An interesting further improvement would be to relax this condition, choosing $\beta = \sigma_k/\tau_k$ adaptively [5, 39].

The standard primal–dual hybrid gradient algorithm is known to include several important algorithms as special cases. The Bregman extension of the algorithm is equally versatile. We mention one interesting example. Suppose the matrix A in (25) is a product of two matrices $A = CB$. Then (25) is equivalent to

$$\begin{array}{ll} \text{minimize} & f(x) + g(y) \\ \text{subject to} & Bx = y \end{array} \tag{58}$$

where $g(y) = \delta_{\{b\}}(Cy)$. The standard (Euclidean) proximal operator of g is the mapping

$$\text{prox}_g(u) = \operatorname*{argmin}_{Cy=b} \|y - u\|_2^2. \tag{59}$$

The PDHG algorithm applied to the reformulated problem requires in each iteration an evaluation of the Bregman proximal operator of f, matrix–vector products with B and B^T, and the solution of the least norm problem in the definition of prox_g. For $C = A$, $B = I$, this can be interpreted as a Bregman extension of the Douglas–Rachford algorithm, or of Spingarn's method for convex optimization with equality constraints.

Acknowledgements We thank Martin S. Andersen for suggestions that greatly improved the implementation used in Section 4. We also thank the editor and the reviewers for their insightful feedback and valuable suggestions.

References

[1] P. Amestoy, T. Davis, and I. Duff. An approximate minimum degree ordering. *SIAM Journal on Matrix Analysis and Applications*, 17(4):886–905, 1996.

[2] M. Andersen, J. Dahl, and L. Vandenberghe. *CVXOPT: A Python Package for Convex Optimization, Version 1.2.4*. `www.cvxopt.org`, 2020.

[3] M. S. Andersen, J. Dahl, and L. Vandenberghe. Logarithmic barriers for sparse matrix cones. *Optimization Methods and Software*, 28(3):396–423, 2013.

[4] M. S. Andersen and L. Vandenberghe. *CHOMPACK: A Python Package for Chordal Matrix Computations, Version 2.2.1*, 2015. `cvxopt.github.io/chompack`.

[5] D. Applegate, M. Dóaz, O. Hinder, H. Lu, M. Lubin, B. O'Donoghue, and W. Schudy. Practical large-scale linear programming using primal-dual hybrid gradient. *arXiv*, 2021.

[6] A. Auslender and M. Teboulle. Interior gradient and proximal methods for convex and conic optimization. *SIAM Journal on Optimization*, 16(3):697–725, 2006.

[7] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[8] A. Beck and M. Teboulle. Gradient-based algorithms with applications to signal recovery. In Y. Eldar and D. Palomar, editors, *Convex Optimization in Signal Processing and Communications*. Cambridge University Press, 2009.

[9] S. Bellavia, J. Gondzio, and B. Morini. A matrix-free preconditioner for sparse symmetric positive definite systems and least-squares problems. *SIAM Journal on Scientific Computing*, 35(1):A192–A211, 2013.

[10] S. Bellavia, J. Gondzio, and M. Porcelli. An inexact dual logarithmic barrier method for solving sparse semidefinite programs. *Mathematical Programming*, 178(1-2):109–143, 2019.

[11] S. Bellavia, J. Gondzio, and M. Porcelli. A relaxed interior point method for low-rank semidefinite programming problems with applications to matrix completion. *arXiv*, 2019.

[12] S. J. Benson and Y. Ye. Algorithm 875: DSDP5—software for semidefinite programming. *ACM Transactions on Mathematical Software (TOMS)*, 34(3):16, 2008.

[13] S. J. Benson, Y. Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM Journal on Optimization*, 10:443–461, 2000.

[14] J. R. S. Blair and B. Peyton. An introduction to chordal graphs and clique trees. In A. George, J. R. Gilbert, and J. W. H. Liu, editors, *Graph Theory and Sparse Matrix Computation*. Springer-Verlag, 1993.

[15] B. Borchers. SDPLIB 1.2, a library of semidefinite programming test problems. *Optimization Methods and Software*, 11(1-4):683–690, 1999.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[17] H. Brézis. *Opérateurs maximaux monotones et semi-groupes de contractions dans les espaces de Hilbert*, volume 5 of *North-Holland Mathematical Studies*. North-Holland, 1973.

[18] S. Burer. Semidefinite programming in the space of partial positive semidefinite matrices. *SIAM Journal on Optimization*, 14(1):139–172, 2003.

[19] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 1997.

[20] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40:120–145, 2011.

[21] A. Chambolle and T. Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, pages 161–319, 2016.

[22] A. Chambolle and T. Pock. On the ergodic convergence rates of a first-order primal-dual algorithm. *Mathematical Programming, Series A*, 159:253–287, 2016.

[23] G. Chen and M. Teboulle. Convergence analysis of a proximal-like minimization algorithm using Bregman functions. *SIAM Journal on Optimization*, 3:538–543, 1993.

[24] L. Condat. A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms. *Journal of Optimization Theory and Applications*, 158(2):460–479, 2013.

[25] D. Davis and W. Yin. A three-operator splitting scheme and its optimization applications. 2015. `arxiv.org/abs/1504.01032`.

[26] J. Eckstein. Nonlinear proximal point algorithms using Bregman functions, with applications to convex programming. *Mathematics of Operations Research*, 18(1):202–226, 1993.

[27] A. Eltved, J. Dahl, and M. S. Andersen. On the robustness and scalability of semidefinite relaxation for optimal power flow problems. *Optimization and Engineering*, pages 1–18, 2020.

[28] E. Esser, X. Zhang, and T. Chan. A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science. *SIAM Journal on Imaging Sciences*, 3(4):1015–1046, 2010.

[29] K. Fujisawa, M. Kojima, and K. Nakata. Exploiting sparsity in primal-dual interior-point methods for semidefinite programming. *Mathematical Programming*, 79(1-3):235–253, October 1997.

[30] M. Fukuda, M. Kojima, K. Murota, and K. Nakata. Exploiting sparsity in semidefinite programming via matrix completion I: general framework. *SIAM Journal on Optimization*, 11:647–674, 2000.

[31] R. Grone, C. R. Johnson, E. M Sá, and H. Wolkowicz. Positive definite completions of partial Hermitian matrices. *Linear Algebra and Appl.*, 58:109–124, 1984.

[32] O. Güler. Ergodic convergence in proximal point algorithms with Bregman functions. In D.-Z. Du and J. Sun, editors, *Advances in Optimization and Approximation*, pages 155–165. Springer, 1994.

[33] B. He and X. Yuan. Convergence analysis of primal-dual algorithms for a saddle-point problem: from contraction perspective. *SIAM Journal on Imaging Sciences*, 5(1):119–149, 2012.

[34] S. Kim, M. Kojima, M. Mevissen, and M. Yamashita. Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion. *Mathematical Programming*, 129:33–68, 2011.

[35] K. Kobayashi, S. Kim, and M. Kojima. Correlative sparsity in primal-dual interior-point methods for LP, SDP, and SOCP. *Applied Mathematics and Optimization*, 58(1):69–88, 2008.

[36] S. Kolodziej, M. Aznaveh, M. Bullock, J. David, T. Davis, M. Henderson, Y. Hu, and R. Sandstrom. The suitesparse matrix collection website interface. *Journal of Open Source Software*, 4(35):1244, 2019.

[37] T. Lin, S. Ma, Y. Ye, and S. Zhang. An ADMM-based interior-point method for large-scale linear programming. *Optimization Methods and Software*, 36(2-3):389–424, 2021.

[38] R. Madani, A. Kalbat, and J. Lavaei. ADMM for sparse semidefinite programming with applications to optimal power flow problem. In *Proceedings of the 54th IEEE Converence on Decision and Control*, pages 5932–5939, 2015.

[39] Y. Malitsky and T. Pock. A first-order primal-dual algorithm with linesearch. *SIAM Journal on Optimization*, 28(1):411–432, 2018.

[40] J. J. Moreau. Proximité et dualité dans un espace hilbertien. *Bull. Math. Soc. France*, 93:273–299, 1965.

[41] MOSEK ApS. *The MOSEK Optimization Tools Manual. Version 8.1.*, 2019. Available from `www.mosek.com`.

[42] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota. Exploiting sparsity in semidefinite programming via matrix completion II: implementation and numerical details. *Mathematical Programming Series B*, 95:303–327, 2003.

[43] Yu. Nesterov. *Lectures on Convex Optimization*. Springer Publishing Company, Incorporated, 2018.

[44] Yu. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Methods in Convex Programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, 1994.

[45] D. O'Connor and L. Vandenberghe. On the equivalence of the primal-dual hybrid gradient method and Douglas—Rachford splitting. *Mathematical Programming*, 179(1-2):85–108, 2020.

[46] S. K. Pakazad, A. Hansson, M. S. Andersen, and A. Rantzer. Distributed semidefinite programming with application to large-scale system analysis. *IEEE Transactions on Automatic Control*, 63(4):1045–1058, 2018.

[47] T. Pock, D. Cremers, H. Bischof, and A. Chambolle. An algorithm for minimizing the Mumford-Shah functional. In *Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV)*, pages 1133–1140, 2009.

[48] S. Pougkakiotis and J. Gondzio. An interior point-proximal method of multipliers for convex quadratic programming. *Computational Optimization and Applications*, 78, March 2021.

[49] R. Shefi and M. Teboulle. Rate of convergence analysis of decomposition methods based on the proximal method of multipliers for convex minimization. *SIAM Journal on Optimization*, 24(1):269–297, 2014.

[50] G. Srijuntongsiri and S. Vavasis. A fully sparse implementation of a primal-dual interior-point potential reduction method for semidefinite programming. 2004. `arXiv:cs/0412009`.

[51] Y. Sun, M. S. Andersen, and L. Vandenberghe. Decomposition in conic optimization with partially separable structure. *SIAM Journal on Optimization*, 24:873–897, 2014.

[52] Y. Sun and L. Vandenberghe. Decomposition methods for sparse matrix nearness problems. *SIAM Journal on Matrix Analysis and Applications*, 36(4):1691–1717, 2015.

[53] P. Tseng. A modified forward-backward splitting method for maximal monotone mappings. *SIAM Journal on Control and Optimization*, 38(2):431–446, 2000.

[54] L. Vandenberghe and M. S. Andersen. Chordal graphs and semidefinite optimization. *Foundations and Trends in Optimization*, 1(4):241–433, 2014.

[55] L. Vandenberghe and S. Boyd. A primal-dual potential reduction method for problems involving matrix inequalities. *Mathematical Programming*, 69(1):205–236, July 1995.

[56] B. C. Vũ. A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Advances in Computational Mathematics*, 38:667–681, 2013.

[57] M. Yan. A new primal–dual algorithm for minimizing the sum of three functions with a linear operator. *Journal of Scientific Computing*, 76(3):1698–1717, September 2018.

[58] R. Y. Zhang and J. Lavaei. Sparse semidefinite programs with guaranteed near-linear time complexity via dualized clique tree conversion. *Mathematical Programming*, 188(1):351–393, 2021.

[59] Y. Zheng, G. Fantuzzi, A. Papachristodolou, P. Goulart, and A. Wynn. Fast ADMM for semidefinite programs with chordal sparsity. In *2017 American Control Conference (ACC)*, pages 3335–3340, 2017.

[60] Y. Zheng, G. Fantuzzi, and A. Papachristodoulou. Chordal and factor-width decompositions for scalable semidefinite and polynomial optimization. *arXiv*, 2021.

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