A Low-Complexity Parallelizable Numerical Algorithm for Sparse Semidefinite Programming

Ramtin Madani, Abdulrahman Kalbat and Javad Lavaei

***Abstract*—In the past two decades, the semidefinite programming technique has been proven to be extremely successful in the convexificiation of hard optimization problems appearing in graph theory, control theory, polynomial optimization theory, and many areas in engineering. In particular, major power optimization problems, such as optimal power flow, state estimation and unit commitment, can be formulated or well approximated as semidefinite programs (SDPs). However, the inability to efficiently solve large-scale SDPs is an impediment to the deployment of such formulations in practice. Motivated by the significant role of SDPs in revolutionizing the decision-making process for real-world systems, this paper designs a low-complexity numerical algorithm for solving sparse SDPs, using the alternating direction method of multipliers and the notion of tree decomposition in graph theory. The iterations of the designed algorithm are highly parallelizable and enjoy closed-form solutions, whose most expensive computation amounts to eigenvalue decompositions over certain submatrices of the SDP matrix. The proposed algorithm is a general-purpose parallelizable SDP solver for sparse SDPs, and its performance is demonstrated on the SDP relaxation of the optimal power flow problem for real-world benchmark systems with more than 13,600 nodes.**

I. INTRODUCTION

Inspired by the seminal papers [1]–[3], there has been a growing interest in semidefinite programming (SDP), due in part to its applications in combinatorial optimization and a large set of real-world problems across engineering [4]–[7]. Semidefinite programming offers a convex formulation or relaxation framework that is applicable to a wide range of non-convex optimization problems, and has been proven to achieve nontrivial bounds and approximation ratios that are beyond the reach of conventional methods [3], [8]–[10]. While small- to medium-sized SDPs are efficiently solvable by second-order-based interior point methods in polynomial time up to any arbitrary precision [11], these methods are mostly impractical for large-scale SDPs due to computation time and memory issues. The primary obstacle is the requirement of calculating Schur complement matrices and their Cholesky factorizations. Several attempts have been made in order to parallelize this procedure, which have led to software packages such as SDPA and SMCP [12]–[14]. In presence of sparsity, a graph-theoretic analysis of SDP problems is proven to be effective in reducing the number of variables and the order of conic constraints [15], [16]. This approach is remarkably helpful for structured SDP problems, particularly those arising in power system optimization [17].

A promising numerical technique for solving large-scale SDP problems is the alternating direction method of multipliers (ADMM), which is a first-order optimization algorithm proposed in the mid-1970s [18] and [19]. While second-order methods are capable of achieving a high accuracy via expensive iterations, a modest accuracy can be attained through tens of ADMM's low-complexity iterations. In order to obtain a highly accurate solution in a reasonable number of iterations, great effort has been devoted to accelerating ADMM [20], [21]. Because of the sensitivity of the gradient methods to the condition number of the problem's data, diagonal rescaling is proposed in [22] to improve the performance of ADMM. Moreover, several accelerated variants of ADMM as well as parameter tuning methods have been proposed in the literature to significantly improve the speed of convergence for specific application domains [20]. The $\mathcal{O}(\frac{1}{n})$ worst-case convergence rate of ADMM is proven in [23] and [24] under certain assumptions, and a systematic framework is introduced in [25] for the convergence analysis of ADMM by means of control-theoretic methods.

The main objective of this work is to design a general-purpose SDP solver for sparse large-scale SDPs, with a guaranteed convergence and parallelization capabilities under mild assumptions. We start by defining a representative graph for the large-scale SDP problem, from which a decomposed SDP formulation is obtained using a tree/chordal/clique decomposition technique. This decomposition replaces the large-scale SDP matrix variable with certain submatrices of this matrix. In order to solve the decomposed SDP problem iteratively, a distributed ADMM-based algorithm is derived, whose iterations comprise entry-wise matrix multiplication/division and eigendecomposition on certain submatrices of the SDP matrix. By finding the optimal solution for the distributed SDP, one could recover the solution to the original SDP formulation using an explicit formula.

This work is related to and improves upon some recent papers in this area. The paper [26] applies ADMM to the dual SDP formulation, leading to a centralized algorithm that is not parallelizable and is computationally expensive for large-scale SDPs. The work [16] decomposes a sparse SDP into smaller-sized SDPs through a tree decomposition, which are then solved by interior point methods. However, this approach is limited by the large number of consistency constraints. Using a first-order splitting method, [27] solves the decomposed SDP problem created by [16], but the algorithm needs to solve an optimization subproblem at every iteration. Similar

Ramtin Madani is with the Electrical Engineering Department at the University of Texas at Arlington (ramtin.madani@uta.edu). Abdulrahman Kalbat is with the Electrical Engineering Department, United Arab Emirates University (akalbat@uaeu.ac.ae). Javad Lavaei is with the Department of Industrial Engineering and Operations Research, University of California, Berkeley, and also with the Tsinghua-Berkeley Shenzhen Institute (lavaei@berkeley.edu).
This work was supported by ONR YIP Award, DARPA Young Faculty Award, AFOSR YIP Award, NSF CAREER Award 1351279, and NSF EECS Award 1552096. Parts of this work have appeared in the conference paper *"Ramtin Madani, Abdulrahman Kalbat and Javad Lavaei, ADMM for sparse semidefinite programming with applications to optimal power flow problem, IEEE Conference on Decision and Control, 2015"*.

frameworks with the requirement of solving smaller SDPs have been applied to power optimization problems [28]–[32]. In contrast with the above papers, the algorithm proposed in this work is composed of low-complexity and parallelizable iterations, which run fast if the treewidth of the sparsity graph of the SDP problem is small. Since a wide range of real-world optimization problems, including those appearing in power systems, are sparse and benefit from a low treewidth, the proposed algorithm enables solving such problems at scale. This algorithm offers the following advantages compared to the method developed in our related work [33]: i) it can handle arbitrary constraints that are not necessarily local, ii) it does not rely on the inversion of large matrices as an initial step. Both of these improvements are essential for solving general large-scale sparse SDP problems, including power system optimization problems to be discussed later. This paper is also related to [34], which designs a distributed algorithm for second-order conic programs. In contrast to the existing methods, the algorithm to be proposed in this paper applies to higher-order conic problems, and does not require solving any optimization sub-problem at any iteration.

The paper [35], as a conference version of this work, studies the potential of ADMM for solving semidefinite programming problems. However, it fails to solve large-scale SDPs and its examples are limited to matrices with at most 300 rows. The current paper improves upon [35] by means of an accelerated version of ADMM combined with a preconditioner, which enables solving real-world problems with over 13,600 rows in the SDP matrix.

A. Motivation: Power System Optimization

Real-world power optimization problems are concerned with the efficiency, robustness, reliability, security and resiliency of power systems, whose decision variables consist of various real-time parameters across a time horizon and under several failure scenarios. These parameters include voltages, currents, phase angles, power productions, line flows, transformer settings, and the on/off statuses of generators and lines. Several factors contribute to a high computational complexity of power optimization problems, such as the nonlinearities induced by laws of physics and discrete variables, the scale of modern grids, the wide range of failure scenarios, and the level of uncertainty for demand and renewable energy sources. The above-mentioned factors give rise to non-convex mixed-integer optimization problems with tens of thousands of decision parameters. While the expected level of efficiency and reliability in recent years necessitates the use of accurate models of power systems that are inevitably highly nonconvex, current state-of-the-art solvers such as CPLEX, Gurobi and MOSEK are incapable of handling continuous non-convexity and a large number of discrete parameters arising in real-world power system optimization problems.

Recent approaches to tackle computationally-hard power optimization problems rely on convex algebraic and/or geometry methods, such as conic relaxation and Sherali-Adams hierarchies [17], [36]–[38]. These advanced techniques are based on solving SDPs with a considerably large number of variables and constraints. Hence, it is imperative to design efficient and fast algorithms for large-scale SDPs, which are applicable to fundamental power optimization problems such as optimal power flow (OPF). The OPF problem is at the heart of the operation of power systems, which finds an optimal operating point of a power system by minimizing a certain objective function (e.g., transmission loss or generation cost) subject to power flow equations and operational constraints [39], [40].

Several optimization techniques have been studied for the OPF problem in recent years [41]. Due to the non-convexity and NP-hardness of OPF, these algorithms are not robust, lack performance guarantees and may not find a global optimum. The paper [7] evaluates the potentials of SDP relaxations for OPF and shows that a global minimum of the problem can be found using an appropriate SDP formulation if the duality gap is zero. SDP relaxation is shown to find global or near globally optimal solutions (with global optimality guarantees of at least 99%) for IEEE and Polish systems, and theoretically proven to work under different conditions [17], [36], [42]–[46]. Moreover, more advanced SDP relaxations based on the sum-of-squares hierarchy have been proven to be effective for solving hard instances of the OPF problem [37]. Note that SDP relaxation is not unique and therefore if one relaxation does not work for a particular power problem, it is always possible to find a more complex SDP formulation to solve the problem (to obtain different hierarchies of convex relaxation, please refer to the tutorial paper [47] and the references therein).

Due to the great success of SDP for the OPF problem, conic relaxations have been designed for other power optimization problems, including state estimation, unit commitment and charging of electric vehicles [38], [48]–[50]. However, the high dimension of these conic formulations for real-world systems is an impediment to their implementation. The main objective of this work is to design a low-complexity numerical algorithm or a general-purpose SDP solver for large-scale conic problems that can be used for a variety of problems, including those appearing in the operation of power systems.

This paper is organized as follows. Some preliminaries and definitions are provided in Section II. An arbitrary sparse SDP is converted into a decomposed SDP in Section III, for which a numerical algorithm is developed in Section IV. The application of this algorithm for OPF is investigated in Section V. Numerical examples are given in Section VI, followed by concluding remarks in Section VII.

Notations: $\mathbb{R}$, $\mathbb{C}$, and $\mathbb{H}^n$ denote the sets of real numbers, complex numbers, and $n\times n$ Hermitian matrices, respectively. The notation $\mathbf{X}_1 \circ \mathbf{X}_2$ refers to the Hadamard (entrywise) multiplication of matrices $\mathbf{X}_1$ and $\mathbf{X}_2$. The symbols $\langle \cdot, \cdot \rangle$ and $\|\cdot\|_F$ denote the Frobenius inner product and norm of matrices, respectively. The notation $\|\mathbf{v}\|_2$ denotes the ℓ_2-norm of a vector $\mathbf{v}$. The $m \times n$ rectangular identity matrix, whose (i,j) entry is equal to the Kronecker delta δ_{ij}, is denoted by $\mathbf{I}_{m\times n}$. The notations Re$\{\mathbf{W}\}$, Im$\{\mathbf{W}\}$, rank$\{\mathbf{W}\}$, and diag$\{\mathbf{W}\}$ denote the real part, imaginary part, rank, and diagonal of a Hermitian matrix $\mathbf{W}$, respectively. Given a vector $\mathbf{v}$, the notation diag$\{\mathbf{v}\}$ denotes a diagonal square matrix whose entries are given by $\mathbf{v}$. The notation $\mathbf{W} \succeq 0$

means that $\mathbf{W}$ is Hermitian and positive semidefinite. The notation "$\mathbf{i}$" is reserved for the imaginary unit. The superscripts $(\cdot)^*$ and $(\cdot)^{\mathrm{T}}$ represent the conjugate transpose and transpose operators, respectively. Given a matrix $\mathbf{W}$, its (l,m) entry is denoted as W_{lm}. The subscript $(\cdot)_{\text{opt}}$ is used to refer to an optimal solution to an optimization problem. Given a matrix $\mathbf{W}$, its Moore-Penrose pseudoinverse is denoted as $\mathrm{pinv}\{\mathbf{W}\}$. Given a simple graph $\mathcal{H}$, its vertex and edge sets are denoted by $\mathcal{V}_{\mathcal{H}}$ and $\mathcal{E}_{\mathcal{H}}$, respectively, and the graph $\mathcal{H}$ is shown as $\mathcal{H} = (\mathcal{V}_{\mathcal{H}}, \mathcal{E}_{\mathcal{H}})$. Given two sets $\mathcal{S}_1$ and $\mathcal{S}_2$, the notation $\mathcal{S}_1 \backslash \mathcal{S}_2$ denotes the set of all elements of $\mathcal{S}_1$ that do not exist in $\mathcal{S}_2$. Given a Hermitian matrix $\mathbf{W}$ and two sets of positive integer numbers $\mathcal{S}_1$ and $\mathcal{S}_2$, define $\mathbf{W}\{\mathcal{S}_1, \mathcal{S}_2\}$ as a submatrix of $\mathbf{W}$ obtained through two operations: (i) removing all rows of $\mathbf{W}$ whose indices do not belong to $\mathcal{S}_1$, and (ii) removing all columns of $\mathbf{W}$ whose indices do not belong to $\mathcal{S}_2$. For instance, $\mathbf{W}\{\{\mathbf{1},\mathbf{2}\},\{\mathbf{2},\mathbf{3}\}\}$ is a 2×2 matrix with the entries $W_{12}, W_{13}, W_{22}, W_{23}$. The notation $|\mathcal{D}|$ shows the cardinality of a discrete set $\mathcal{D}$ or the number of vertices of a graph $\mathcal{D}$.

II. Preliminaries

Consider the semidefinite program

$$\begin{aligned} &\underset{\mathbf{X}\in\mathbb{H}^n}{\text{minimize}} && \langle \mathbf{X}, \mathbf{M}_0 \rangle && \text{(1a)}\\ &\text{subject to} && l_s \leq \langle \mathbf{X}, \mathbf{M}_s \rangle \leq u_s, \quad s = 1,\ldots,p, && \text{(1b)}\\ &&& \mathbf{X} \succeq 0. && \text{(1c)} \end{aligned}$$

where $\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_p \in \mathbb{H}^n$, and

$$(l_s, u_s) \in (\{-\infty\} \cup \mathbb{R}) \times (\mathbb{R} \cup \{+\infty\})$$

for every $s = 1,\ldots,p$. Notice that the constraint (1b) reduces to an equality constraint if $l_s = u_s$. Problem (1) is computationally expensive for a large number n due to the presence of the positive semidefinite constraint (1c). However, if $\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_p$ are sparse, this expensive constraint can be decomposed and expressed in terms of some principal submatrices of $\mathbf{X}$ with smaller dimensions. This will be explained next.

A. Representative Graph and Tree Decomposition

In order to leverage any possible sparsity of problem (1), a simple graph shall be defined to capture the zero-nonzero patterns of $\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_p$.

Definition 1 (Representative graph [51])**.** *Define $\mathcal{G} = (\mathcal{V}_{\mathcal{G}}, \mathcal{E}_{\mathcal{G}})$ as the representative graph of the SDP problem* (1)*, which is a simple graph with n vertices whose edges are specified by the nonzero off-diagonal entries of* $\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_p$. *In other words, two arbitrary vertices i and j are connected if the (i,j) entry of at least one of the matrices $\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_p$ is nonzero.*

To illustrate Definition 1, consider the problem (1) with $n = 4$ and $p = 1$ such that

$$\mathbf{M}_0 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 5 & 3 & 0 \\ 1 & 3 & 10 & 2 \\ 0 & 0 & 2 & 1 \end{bmatrix}, \quad \mathbf{M}_1 = \begin{bmatrix} 0 & 3 & 0 & 0 \\ 3 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 2 & 1 & 1 \end{bmatrix} \qquad (2)$$

The representative graph of the SDP problem (1) in this case is a graph with the vertex set $\{1,2,3,4\}$ such that every two vertices are connected to one another except for the vertices 1 and 4. The reason is that the $(1,4)$ entries of $\mathbf{M}_0$ and $\mathbf{M}_1$ are both equal to 0.

Using a tree decomposition algorithm (also known as chordal or clique decomposition), we can obtain a *decomposed* formulation for problem (1), in which the positive semidefinite requirement is imposed on certain principal submatrices of $\mathbf{X}$ as opposed to $\mathbf{X}$ itself.

Definition 2 (Tree decomposition [52])**.** *A tree graph $\mathcal{T}$ is called a tree decomposition of $\mathcal{G}$ if it satisfies the following properties:*

1) *Every node of $\mathcal{T}$ corresponds to and is identified by a subset of $\mathcal{V}_{\mathcal{G}}$.*
2) *Every vertex of $\mathcal{G}$ is a member of at least one node of $\mathcal{T}$.*
3) *$\mathcal{T}_k$ is a connected graph for every $k \in \mathcal{V}_{\mathcal{G}}$, where $\mathcal{T}_k$ denotes the subgraph of $\mathcal{T}$ induced by all nodes of $\mathcal{T}$ containing the vertex k of $\mathcal{G}$.*
4) *The subgraphs $\mathcal{T}_i$ and $\mathcal{T}_j$ have at least one node in common, for every $(i,j) \in \mathcal{E}_{\mathcal{G}}$.*

Each node of $\mathcal{T}$ is a bag (collection) of vertices of $\mathcal{G}$ and hence it is referred to as a **bag**.

As an example, Figure 1 borrowed from [51] shows a tree decomposition of the graph corresponding to the physical structure of the IEEE 14-bus power network. Notice that four properties are satisfied: 1) every bag of the tree decomposition is a collection of the vertices of the graph, 2) every vertex of the graph appears in at least one bag of the tree decomposition, 3) all bags containing each particular vertex of the graph form a connected subgraph in the tree decomposition (for example, there are three bags containing node 2 and they form a connected path), and 4) every two connected vertices of the graph appear in at least one common bag of the tree decomposition.

Let $\mathcal{T} = (\mathcal{V}_{\mathcal{T}}, \mathcal{E}_{\mathcal{T}})$ be an arbitrary tree decomposition of $\mathcal{G}$, with the set of bags $\mathcal{V}_{\mathcal{T}} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_q\}$. As will be discussed in the next section, it is possible to cast problem (1) in terms of those entries of $\mathbf{X}$ that appear in at least one of the submatrices $\mathbf{X}\{\mathcal{C}_1,\mathcal{C}_1\}, \mathbf{X}\{\mathcal{C}_2,\mathcal{C}_2\}, \ldots, \mathbf{X}\{\mathcal{C}_q,\mathcal{C}_q\}$.These entries of $\mathbf{X}$ are referred to as *important entries*. Once the optimal values of the important entries of $\mathbf{X}$ are found via an iterative algorithm, the remaining entries of $\mathbf{X}$ can be obtained through an explicit formula to be stated later.

Among the factors that may contribute to the computational complexity of the decomposed problem are: the size of the largest bag, the number of bags, and the total number of important entries. Finding a tree decomposition that leads to the minimum number of important entries (minimum fill-in problem) or possesses the minimum size for its largest bag (treewidth problem) is known to be NP-hard. The algorithm proposed in this paper reaches its maximum efficiency when a tree decomposition of the sparsity graph with $\max\{|\mathcal{C}_1|, \ldots, |\mathcal{C}_q|\} = O(1)$ is already available. This algorithm only requires a suboptimal tree decomposition with a low width. Good decompositions can be easily found using the nested dissection method. The work [53] proves that

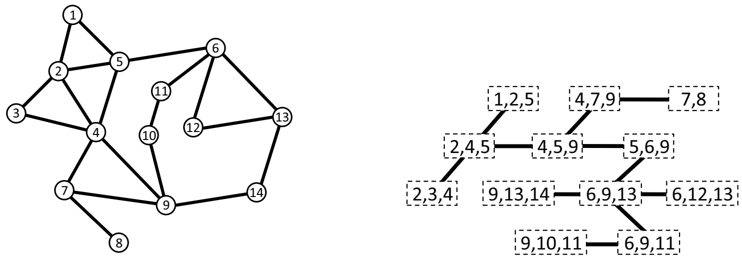


Fig. 1: The graph of the IEEE 14-bus test case (left figure) and a tree decomposition of this graph (right figure)

nested dissection is $O(\log(|\mathcal{G}|))$ suboptimal for bounded-degree graphs, and notes that: "we do not know a class of graphs for which nested dissection is suboptimal by more than a constant factor". There are many other efficient algorithms in the literature that find near-optimal tree decompositions (specially for power networks due to their near planarity) [54], [55]. In all of the experiments of this paper, a tree decomposition of the sparsity graph is obtained in less than 90 seconds, using the algorithm described in [54]. Moreover, optimization problems defined over physical infrastructures, such as power systems, often benefit from the fact that the topology of the system changes slowly, and therefore the tree decomposition may be performed offline and used for a class of SDP problems as opposed to a single instance of the problem.

B. Sparsity Pattern of Matrices

Let $\mathbb{F}^n$ denote the set of symmetric $n \times n$ matrices with entries belonging to the set $\{0, 1\}$. The distributed optimization scheme to be proposed in this work uses a group of sparse slack matrices. We identify the locations of nonzero entries of such matrix variables using descriptive matrices in $\mathbb{F}^n$.

Definition 3. *Given an arbitrary matrix $\mathbf{X} \in \mathbb{H}^n$, define its sparsity pattern as a matrix $\mathbf{N} \in \mathbb{F}^n$ such that $N_{ij} = 1$ if and only if $X_{ij} \neq 0$ for every $i, j \in \{1, ..., n\}$. Let $|\mathbf{N}|$ denote the number of nonzero entries of $\mathbf{N}$. Define the set*

$$\mathcal{S}(\mathbf{N}) \triangleq \{\mathbf{X} \in \mathbb{H}^n \mid \mathbf{X} \circ \mathbf{N} = \mathbf{X}\}.$$

Due to the Hermitian property of $\mathbf{X}$, if d denotes the number of nonzero diagonal entries of $\mathbf{N}$, then every $\mathbf{X} \in \mathcal{S}(\mathbf{N})$ can be specified by $(|\mathbf{N}| + d)/2$ real-valued scalars corresponding to $\mathrm{Re}\{\mathbf{X}\}$ and $(|\mathbf{N}| - d)/2$ real scalars corresponding to $\mathrm{Im}\{\mathbf{X}\}$. Therefore, $\mathcal{S}(\mathbf{N})$ is $|\mathbf{N}|$-dimensional over $\mathbb{R}$.

Definition 4. *Suppose that $\mathcal{T} = (\mathcal{V}_{\mathcal{T}}, \mathcal{E}_{\mathcal{T}})$ is a tree decomposition of the representative graph $\mathcal{G}$ with the bags $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_q$.*

- *For $r = 1, \ldots, q$, define $\mathbf{C}_r \in \mathbb{F}^n$ as a sparsity pattern whose (i, j) entry is equal to 1 if $\{i, j\} \subseteq \mathcal{C}_r$ and is 0 otherwise for every $i, j \in \{1, ..., n\}$.*
- *Define $\mathbf{C} \in \mathbb{F}^n$ as an aggregate sparsity pattern whose (i, j) entry is equal to 1 if and only if $\{i, j\} \subseteq \mathcal{C}_r$ for at least one index $r \in \{1, \ldots, q\}$.*
- *For $s = 0, 1, \ldots, p$, define $\mathbf{N}_s \in \mathbb{F}^n$ as the sparsity pattern of $\mathbf{M}_s$.*

The sparsity pattern $\mathbf{C}$, which can also be interpreted as the adjacency matrix of a chordal extension of $\mathcal{G}$ induced by $\mathcal{T}$, captures the locations of the important entries of $\mathbf{X}$. The matrix $\mathbf{C}$ will later be used to describe the domain of definition for the variable of the decomposed SDP problem.

C. Indicator Functions

To streamline the formulation, we will replace any positivity or positive semidefiniteness constraints in the decomposed SDP problem by the indicator functions introduced below.

Definition 5. *For every $l \in \{-\infty\} \cup \mathbb{R}$ and $u \in \mathbb{R} \cup \{+\infty\}$, define the convex indicator function $\mathcal{I}_{l,u} : \mathbb{R} \to \{0, +\infty\}$ as*

$$\mathcal{I}_{l,u}(x) \triangleq \begin{cases} 0 & \text{if } \ l \leq x \leq u \\ +\infty & \text{otherwise} \end{cases}$$

Definition 6. *For every $r \in \{1, 2, \ldots, q\}$, define the convex indicator function $\mathcal{J}_r : \mathbb{H}^n \to \{0, +\infty\}$ as*

$$\mathcal{J}_r(\mathbf{X}) \triangleq \begin{cases} 0 & \text{if } \ \mathbf{X}\{\mathcal{C}_r, \mathcal{C}_r\} \succeq 0 \\ +\infty & \text{otherwise} \end{cases}$$

III. Decomposed SDP

Consider the problem

$$\begin{aligned} &\underset{\mathbf{X} \in \mathcal{S}(\mathbf{C})}{\text{minimize}} && \langle \mathbf{X}, \mathbf{M}_0 \rangle && && \text{(3a)} \\ &\text{subject to} && l_s \leq \langle \mathbf{X}, \mathbf{M}_s \rangle \leq u_s, && s = 1, \ldots, p, && \text{(3b)} \\ & && \mathbf{X}\{\mathcal{C}_r, \mathcal{C}_r\} \succeq 0, && r = 1, \ldots, q && \text{(3c)} \end{aligned}$$

which is referred to as *decomposed SDP* throughout this paper. Due to the chordal theorem [56], problems (1) and (3) lead to the same optimal objective value. Furthermore, if $\mathbf{X}_{\mathrm{ref}} \in \mathcal{S}(\mathbf{C})$ denotes an arbitrary solution of the decomposed SDP problem (3), then there exists a solution $\mathbf{X}_{\mathrm{opt}}$ to the SDP problem (1) such that $\mathbf{X}_{\mathrm{opt}} \circ \mathbf{C} = \mathbf{X}_{\mathrm{ref}}$.

The matrix $\mathbf{X}_{\mathrm{opt}}$ can be constructed by mapping certain entries of $\mathbf{X}_{\mathrm{ref}}$ to 0 according to the sparsity patten. These entries of the matrix variable $\mathbf{X}$ are referred to as *missing entries* and cannot be found by solving the decomposed problem (3). Several matrix completion approaches can be adopted to find the missing entries, which enable the construction of a feasible point for the original SDP (1). An algorithm is proposed in [16] and [15] that obtains a completion for $\mathbf{X}_{\mathrm{ref}}$ within the set $\{\mathbf{X} \in \mathbb{H}^n \,|\, \mathbf{X} \circ \mathbf{C} = \mathbf{X}_{\mathrm{ref}}, \ \mathbf{X} \succeq 0\}$ whose determinant is maximum. However, such a solution may not be favorable for applications that require a low-rank solution such as an SDP relaxation. It is also known that there exists a polynomial-time algorithm to fill a partially-known real-valued matrix in such a way that the rank of the resulting matrix becomes equal to the highest rank among all bags [57], [58]. In [51], we extended this result to the complex domain by proposing a recursive algorithm that transforms $\mathbf{X}_{\rm ref} \in \mathcal{S}(\mathbf{C})$ into a solution $\mathbf{X}_{\rm opt}$ for the original SDP problem (1) whose rank is upper bounded by the maximum rank among the matrices $\mathbf{X}_{\rm ref}\{\mathcal{C}_1, \mathcal{C}_1\}, \mathbf{X}_{\rm ref}\{\mathcal{C}_2, \mathcal{C}_2\}, \dots, \mathbf{X}_{\rm ref}\{\mathcal{C}_q, \mathcal{C}_q\}$. This algorithm is stated below for completeness.

Matrix completion algorithm:

1) Set $\mathcal{T}' := \mathcal{T}$ and $\mathbf{X} := \mathbf{X}_{\rm ref}$.
2) If $\mathcal{T}'$ has a single node, then consider $\mathbf{X}_{\rm opt}$ as $\mathbf{X}$ and terminate; otherwise continue to the next step.
3) Choose two bags $\mathcal{C}_x$ and $\mathcal{C}_y$ of $\mathcal{T}'$ such that $\mathcal{C}_x$ is a leaf of $\mathcal{T}'$ and $\mathcal{C}_y$ is its unique neighbor.
4) Define

$$\mathbf{K} \triangleq \operatorname{pinv}\{\mathbf{X}\{\mathcal{C}_x \cap \mathcal{C}_y, \mathcal{C}_x \cap \mathcal{C}_y\}\} \tag{4a}$$

$$\mathbf{G}_x \triangleq \mathbf{X}\{\mathcal{C}_x \setminus \mathcal{C}_y, \mathcal{C}_x \cap \mathcal{C}_y\} \tag{4b}$$

$$\mathbf{G}_y \triangleq \mathbf{X}\{\mathcal{C}_y \setminus \mathcal{C}_x, \mathcal{C}_x \cap \mathcal{C}_y\} \tag{4c}$$

$$\mathbf{E}_x \triangleq \mathbf{X}\{\mathcal{C}_x \setminus \mathcal{C}_y, \mathcal{C}_x \setminus \mathcal{C}_y\} \tag{4d}$$

$$\mathbf{E}_y \triangleq \mathbf{X}\{\mathcal{C}_y \setminus \mathcal{C}_x, \mathcal{C}_y \setminus \mathcal{C}_x\} \tag{4e}$$

$$\mathbf{S}_x \triangleq \mathbf{E}_x - \mathbf{G}_x \mathbf{K} \mathbf{G}_x^* = \mathbf{Q}_x \mathbf{D}_x \mathbf{Q}_x^* \tag{4f}$$

$$\mathbf{S}_y \triangleq \mathbf{E}_y - \mathbf{G}_y \mathbf{K} \mathbf{G}_y^* = \mathbf{Q}_y \mathbf{D}_y \mathbf{Q}_y^* \tag{4g}$$

where $\mathbf{Q}_x \mathbf{D}_x \mathbf{Q}_x^*$ and $\mathbf{Q}_y \mathbf{D}_y \mathbf{Q}_y^*$ denote the eigenvalue decompositions of $\mathbf{S}_x$ and $\mathbf{S}_y$ with the diagonals of $\mathbf{D}_x$ and $\mathbf{D}_y$ arranged in descending order. Then, update a part of $\mathbf{X}$ as follows:

$$\begin{aligned}\mathbf{X}\{\mathcal{C}_y \setminus \mathcal{C}_x, \mathcal{C}_x \setminus \mathcal{C}_y\} := \mathbf{G}_y \mathbf{K} \mathbf{G}_x^* \\ + \mathbf{Q}_y \sqrt{\mathbf{D}_y}\ \mathbf{I}_{|\mathcal{C}_y \setminus \mathcal{C}_x| \times |\mathcal{C}_x \setminus \mathcal{C}_y|} \sqrt{\mathbf{D}_x}\ \mathbf{Q}_x^*\end{aligned}$$

and update $\mathbf{X}\{\mathcal{C}_x \setminus \mathcal{C}_y, \mathcal{C}_y \setminus \mathcal{C}_x\}$ accordingly to preserve the Hermitian property of $\mathbf{X}$.
5) Update $\mathcal{T}'$ by merging $\mathcal{C}_x$ into $\mathcal{C}_y$, i.e., replace $\mathcal{C}_y$ with $\mathcal{C}_x \cup \mathcal{C}_y$ and then remove $\mathcal{C}_x$ from $\mathcal{T}'$.
6) Go back to step 2.

Theorem 1. *Consider an arbitrary solution $\mathbf{X}_{\rm ref}$ of the decomposed SDP problem* (3)*. The output of the matrix completion algorithm, denoted as $\mathbf{X}_{\rm opt}$, is a solution of the original SDP problem* (1)*. Moreover, the rank of $\mathbf{X}_{\rm opt}$ is smaller than or equal to:*

$$\max\left\{\operatorname{rank}\{\mathbf{X}_{\rm ref}\{\mathcal{C}_r, \mathcal{C}_r\}\} \;\middle|\; r = 1, \dots, q\right\}.$$

Proof. Please refer to [51] or [17] for the proof. □

IV. Alternating Direction Method of Multipliers

Consider the optimization problem

$$\underset{\substack{\mathbf{x}\in\mathbb{R}^{n_x}\\ \mathbf{y}\in\mathbb{R}^{n_y}}}{\text{minimize}} \quad f(\mathbf{x}) + g(\mathbf{y}) \tag{5a}$$

$$\text{subject to} \quad \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{c}. \tag{5b}$$

where $\mathbf{c} \in \mathbb{R}^{n_c}$, $\mathbf{A} \in \mathbb{R}^{n_c \times n_x}$ and $\mathbf{B} \in \mathbb{R}^{n_c \times n_y}$ are constant matrices, and $f : \mathbb{R}^{n_x} \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^{n_y} \to \mathbb{R} \cup \{+\infty\}$ are convex functions. Notice that the variables $\mathbf{x}$ and $\mathbf{y}$ are coupled through the linear constraint (5b) while the objective function is separable. The augmented Lagrangian function for problem (5) is equal to

$$\begin{aligned}\mathcal{L}_\mu(\mathbf{x}, \mathbf{y}, \lambda) = f(\mathbf{x}) + g(\mathbf{y}) \\ + \lambda^{\rm T}(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{c}) + (\mu/2)\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{c}\|_2^2,\end{aligned} \tag{6}$$

where $\lambda \in \mathbb{R}^{n_c}$ is the Lagrange multiplier associated with the constraint (5b), and $\mu \in \mathbb{R}$ is a fixed parameter. ADMM is one approach for solving problem (5), which performs the following procedure at each iteration [59]:

$$\mathbf{x}^{k+1} = \underset{\mathbf{x}\in\mathbb{R}^{n_x}}{\arg\min} \quad \mathcal{L}_\mu(\mathbf{x}, \mathbf{y}^k, \lambda^k), \tag{7a}$$

$$\mathbf{y}^{k+1} = \underset{\mathbf{y}\in\mathbb{R}^{n_y}}{\arg\min} \quad \mathcal{L}_\mu(\mathbf{x}^{k+1}, \mathbf{y}, \lambda^k), \tag{7b}$$

$$\lambda^{k+1} = \lambda^k + \mu(\mathbf{A}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{y}^{k+1} - \mathbf{c}). \tag{7c}$$

where $k = 0, 1, 2, \dots$, for an arbitrary initialization $(\mathbf{x}^0, \mathbf{y}^0, \lambda^0)$. In these equations, "argmin" means an arbitrary minimizer of a convex function and does not need any uniqueness assumption. Notice that each of the updates (7a) and (7b) is an optimization sub-problem with respect to either $\mathbf{x}$ and $\mathbf{y}$, by freezing the other variable at its latest value. Several acceleration techniques have been proposed in the literature, aiming to improve the convergence behavior of ADMM [20], [25]. One such approach, regarded as over-relaxed ADMM, involves adopting a sequence of intermediate Lagrange multipliers $\{\hat{\lambda}^k\}_{k=1}^\infty$ as follows:

$$\mathbf{x}^{k+1} = \underset{\mathbf{x}\in\mathbb{R}^{n_x}}{\arg\min} \quad \mathcal{L}_\mu(\mathbf{x}, \mathbf{y}^k, \lambda^k), \tag{8a}$$

$$\hat{\lambda}^{k+1} = \lambda^k + \mu(\alpha - 1)(\mathbf{A}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{y}^k - \mathbf{c}), \tag{8b}$$

$$\mathbf{y}^{k+1} = \underset{\mathbf{y}\in\mathbb{R}^{n_y}}{\arg\min} \quad \mathcal{L}_\mu(\mathbf{x}^{k+1}, \mathbf{y}, \hat{\lambda}^{k+1}), \tag{8c}$$

$$\lambda^{k+1} = \hat{\lambda}^{k+1} + \mu(\mathbf{A}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{y}^{k+1} - \mathbf{c}), \tag{8d}$$

where $\alpha \in [1, 2]$ is a fixed parameter. We employ the residue sequence $\{\varepsilon^k\}_{k=1}^\infty$ proposed in [20] as measure for convergence:

$$\varepsilon^{k+1} = (1/\mu)\|\lambda^{k+1} - \lambda^k\|_2^2 + \mu\|\mathbf{B}(y^{k+1} - y^k)\|_2^2 \tag{9}$$

ADMM is particularly interesting for the cases where sub-problems can be solved efficiently through an explicit formula. Under such circumstances, it would be possible to execute a large number of iterations in a short amount of time. In this section, we first cast the decomposed SDP problem (3) in the form (5) and then regroup the variables into two blocks $\mathcal{P}_1$ and $\mathcal{P}_2$ playing the roles of $\mathbf{x}$ and $\mathbf{y}$ in the ADMM algorithm.

A. Projection Onto Positive Semidefinite Cone

The algorithm to be proposed in this work requires the projection of q matrices belonging to $\mathbb{H}^{|\mathcal{C}_1|}, \mathbb{H}^{|\mathcal{C}_2|}, \dots, \mathbb{H}^{|\mathcal{C}_q|}$ onto the positive semidefinite cone. This is probably the most computationally expensive part of each iteration.

Definition 7. *For a given Hermitian matrix $\widehat{\mathbf{Z}}$, define the unique solution to the optimization problem*

$$\underset{\mathbf{Z}\in\mathbb{H}^m}{\text{minimize}} \quad \|\mathbf{Z} - \widehat{\mathbf{Z}}\|_F^2 \tag{10a}$$

$$\text{subject to} \quad \mathbf{Z} \succeq 0 \tag{10b}$$

Algorithm 1 Over-relaxed ADMM for decomposed SDP

Initialize $\mathbf{X}^0, \{z_s^0\}_{s=0}^p, \{\mathbf{X}_{C;r}^0\}_{r=1}^q, \{\mathbf{X}_{N;s}^0\}_{s=0}^p, \{\mathbf{\Lambda}_{C;r}^0\}_{r=1}^q, \{\mathbf{\Lambda}_{N;s}^0\}_{s=0}^p, \{\lambda_{z;s}^0\}_{s=0}^p$

repeat

$$\mathbf{X}^{k+1} := \left[\textstyle\sum_{r=1}^q \mathbf{C}_r \circ (\mathbf{X}_{C;r}^k - \mathbf{\Lambda}_{C;r}^k/\mu) + \sum_{s=1}^p \mathbf{N}_s \circ (\mathbf{X}_{N;s}^k - \mathbf{\Lambda}_{N;s}^k/\mu)\right] \oslash_{\mathbf{C}} \left[\textstyle\sum_{r=1}^q \mathbf{C}_r + \sum_{s=1}^p \mathbf{N}_s\right]$$

$$z_0^{k+1} := \langle \mathbf{M}_0, \mathbf{X}_{N;0}^k \rangle - (\lambda_{z;0}^k + 1)/\mu$$

$$z_s^{k+1} := \max\{\min\{\langle \mathbf{M}_s, \mathbf{X}_{N;s}^k \rangle - \lambda_{z;s}^k/\mu, u_s\}, l_s\} \quad \text{for} \quad s = 1, 2, \ldots, p$$

$$\hat{\mathbf{\Lambda}}_{C;r}^{k+1} := \mathbf{\Lambda}_{C;r}^k + \mu(\alpha - 1)(\mathbf{X}^{k+1} \circ \mathbf{C}_r - \mathbf{X}_{C;r}^k) \quad \text{for} \quad r = 1, 2, \ldots, q$$

$$\hat{\mathbf{\Lambda}}_{N;s}^{k+1} := \mathbf{\Lambda}_{N;s}^k + \mu(\alpha - 1)(\mathbf{X}^{k+1} \circ \mathbf{N}_s - \mathbf{X}_{N;s}^k) \quad \text{for} \quad s = 0, 1, \ldots, p$$

$$\hat{\lambda}_{z;s}^{k+1} := \lambda_{z;s}^k + \mu(\alpha - 1)(z_s^{k+1} - \langle \mathbf{M}_s, \mathbf{X}_{N;s}^k \rangle) \quad \text{for} \quad s = 0, 1, \ldots, p$$

$$\mathbf{X}_{C;r}^{k+1} := (\mathbf{X}^{k+1} \circ \mathbf{C}_r + \hat{\mathbf{\Lambda}}_{C;r}^k/\mu)^+ \quad \text{for} \quad r = 1, 2, \ldots, q$$

$$y_s^{k+1} := \frac{z_s^{k+1} + \hat{\lambda}_{z;s}^k/\mu - \langle \mathbf{M}_s, \mathbf{N}_s \circ \mathbf{X}^{k+1} + \hat{\mathbf{\Lambda}}_{N;s}^k/\mu \rangle}{1 + \|\mathbf{M}_s\|_F^2} \quad \text{for} \quad s = 0, 1, \ldots, p$$

$$\mathbf{X}_{N;s}^{k+1} := \mathbf{N}_s \circ \mathbf{X}^{k+1} + \hat{\mathbf{\Lambda}}_{N,s}^k/\mu + y_s^{k+1}\mathbf{M}_s \quad \text{for} \quad s = 0, 1, \ldots, p$$

$$\mathbf{\Lambda}_{C;r}^{k+1} := \hat{\mathbf{\Lambda}}_{C;r}^{k+1} + \mu(\mathbf{X}^{k+1} \circ \mathbf{C}_r - \mathbf{X}_{C;r}^{k+1}) \quad \text{for} \quad r = 1, 2, \ldots, q$$

$$\mathbf{\Lambda}_{N;s}^{k+1} := \hat{\mathbf{\Lambda}}_{N;s}^{k+1} + \mu(\mathbf{X}^{k+1} \circ \mathbf{N}_s - \mathbf{X}_{N;s}^{k+1}) \quad \text{for} \quad s = 0, 1, \ldots, p$$

$$\lambda_{z;s}^{k+1} := \hat{\lambda}_{z;s}^{k+1} + \mu(z_s^{k+1} - \langle \mathbf{M}_s, \mathbf{X}_{N;s}^{k+1} \rangle) \quad \text{for} \quad s = 0, 1, \ldots, p$$

until meet stopping criterion

as the projection of $\widehat{\mathbf{Z}}$ onto the cone of positive semidefinite matrices, and denote it as $\widehat{\mathbf{Z}}^+$.

The next Lemma reveals the interesting fact that problem (10) can be solved through an eigenvalue decomposition of $\widehat{\mathbf{Z}}$.

Lemma 1. *Let $\widehat{\mathbf{Z}} = \mathbf{Q} \times \operatorname{diag}\{(\nu_1 \ldots, \nu_m)\} \times \mathbf{Q}^*$ denote the eigenvalue decomposition of $\widehat{\mathbf{Z}}$. The solution of the projection problem* (10) *is given by*

$$\widehat{\mathbf{Z}}^+ = \mathbf{Q} \times \operatorname{diag}\{(\max\{\nu_1, 0\}, \ldots, \max\{\nu_m, 0\})\} \times \mathbf{Q}^*$$

Proof. Please refer to [60] for the proof. □

B. ADMM for Decomposed SDP

We apply ADMM to the following reformulation of the decomposed SDP problem (3):

$$\begin{aligned} \underset{\substack{\mathbf{X}\in\mathcal{S}(\mathbf{C}) \\ \{\mathbf{X}_{N;s}\in\mathcal{S}(\mathbf{N}_s)\}_{s=0}^p \\ \{\mathbf{X}_{C;r}\in\mathcal{S}(\mathbf{C}_r)\}_{r=1}^q \\ \{z_s\in\mathbb{R}\}_{s=0}^p}}{\text{minimize}} \quad & z_0 + \sum_{s=1}^p \mathcal{I}_{l_s,u_s}(z_s) + \sum_{r=1}^q \mathcal{J}_r(\mathbf{X}_{C;r}) \\ \text{subject to} \quad & \mathbf{X} \circ \mathbf{C}_r = \mathbf{X}_{C;r}, \quad r = 1, 2, \ldots, q, \quad \text{(11a)} \\ & \mathbf{X} \circ \mathbf{N}_s = \mathbf{X}_{N;s}, \quad s = 0, 1, \ldots, p, \quad \text{(11b)} \\ & z_s = \langle \mathbf{M}_s, \mathbf{X}_{N;s} \rangle, \quad s = 0, 1, \ldots, p. \quad \text{(11c)} \end{aligned}$$

If $\mathbf{X}$ is a feasible solution of (11) with a finite objective value, then

$$\mathcal{J}_r(\mathbf{X}) = \mathcal{J}_r(\mathbf{X} \circ \mathbf{C}_r) \overset{\text{(11a)}}{=} \mathcal{J}_r(\mathbf{X}_{C;r}) = 0$$

which concludes that $\mathbf{X}\{\mathcal{C}_r, \mathcal{C}_r\} \succeq 0$. Morever,

$$\begin{aligned} \mathcal{I}_{l_s,u_s}(\langle \mathbf{X}, \mathbf{M}_s \rangle) &= \mathcal{I}_{l_s,u_s}(\langle \mathbf{X} \circ \mathbf{N}_s, \mathbf{M}_s \rangle) \\ &\overset{\text{(11b)}}{=} \mathcal{I}_{l_s,u_s}(\langle \mathbf{X}_{N;s}, \mathbf{M}_s \rangle) \\ &\overset{\text{(11c)}}{=} \mathcal{I}_{l_s,u_s}(z_s) = 0 \end{aligned}$$

which yields that $l_s \le \langle \mathbf{X}, \mathbf{M}_s \rangle \le u_s$. Therefore, $\mathbf{X}$ is a feasible point for problem (3) as well, with the same objective value. Define

1) $\mathbf{\Lambda}_{C;r} \in \mathcal{S}(\mathbf{C}_r)$ as the Lagrange multiplier associated with the constraint (11a) for $r = 1, 2, \ldots, q$,
2) $\mathbf{\Lambda}_{N;s} \in \mathcal{S}(\mathbf{N}_s)$ as the Lagrange multiplier associated with the constraint (11b) for $s = 0, 1, \ldots, p$,
3) $\lambda_{z;s} \in \mathbb{R}$ as the Lagrange multiplier associated with the constraint (11c) for $s = 0, 1, \ldots, p$.

We regroup the primal and dual variables as

$$\begin{aligned} &\text{(Block 1)} \quad \mathcal{P}_1 = (\mathbf{X}, \{z_s\}_{s=0}^p) \\ &\text{(Block 2)} \quad \mathcal{P}_2 = (\{\mathbf{X}_{C;r}\}_{r=1}^q, \{\mathbf{X}_{N;s}\}_{s=0}^p) \\ &\text{(Dual)} \quad \mathcal{D} = (\{\mathbf{\Lambda}_{C;r}\}_{r=1}^q, \{\mathbf{\Lambda}_{N;s}\}_{s=0}^p, \{\lambda_{z;s}\}_{s=0}^p). \end{aligned}$$

Note that "block 1", "block 2" and "$\mathcal{D}$" play the roles of $\mathbf{x}$, $\mathbf{y}$ and λ in the standard formulation of ADMM, respectively.

The augmented Lagrangian can be calculated as

$$
\begin{aligned}
&(2/\mu)\mathcal{L}_\mu(\mathcal{P}_1,\mathcal{P}_2,\mathcal{D}) = \mathcal{L}_D(\mathcal{D})/\mu^2\\
&+\|z_0-\langle\mathbf{M}_0,\mathbf{X}_{N;0}\rangle+(1+\lambda_{z;0})/\mu\|_F^2\\
&+\sum_{s=1}^{p}\left(\|z_s-\langle\mathbf{M}_s,\mathbf{X}_{N;s}\rangle+\lambda_{z;s}/\mu\|_F^2+\mathcal{I}_{l_s,u_s}(z_s)\right)\\
&+\sum_{r=1}^{q}\left(\|\mathbf{X}\circ\mathbf{C}_r-\mathbf{X}_{C;r}+(1/\mu)\mathbf{\Lambda}_{C;r}\|_F^2+\mathcal{J}_r(\mathbf{X}_{C;k})\right)\\
&+\sum_{s=1}^{p}\|\mathbf{X}\circ\mathbf{N}_s-\mathbf{X}_{N;s}+(1/\mu)\mathbf{\Lambda}_{N;s}\|_F^2 \qquad (13)
\end{aligned}
$$

where

$$
\begin{aligned}
\mathcal{L}_D(\mathcal{D}) =& -(1+\lambda_{z;0})^2\\
&-\sum_{s=1}^{p}\lambda_{z;s}^2-\sum_{r=1}^{q}\|\mathbf{\Lambda}_{C;r}\|_F^2-\sum_{s=1}^{p}\|\mathbf{\Lambda}_{N;s}\|_F^2 \qquad (14)
\end{aligned}
$$

Using the blocks $\mathcal{P}_1$ and $\mathcal{P}_2$, the ADMM iterations for problem (11) can be expressed as follows:

1) The subproblem (7a) in terms of $\mathcal{P}_1$ consists of two parallel steps:
   (a) *Minimization in terms of* $\mathbf{X}$: This step consists of $|\mathbf{C}|$ scalar quadratic and unconstrained programs. It possesses an explicit formula that involves $|\mathbf{C}|$ parallel multiplication operations.
   (b) *Minimization in terms of* $\{z_s\}_{s=0}^p$: This step consists of $p+1$ scalar quadratic programs each with a box constraint. It possesses an explicit formula that involves $p+1$ parallel multiplication operations.
2) The subproblem (7b) in terms of $\mathcal{P}_2$ also consists of two parallel steps:
   (a) *Minimization in terms of* $\{\mathbf{X}_{C;r}\}_{r=1}^q$: This step consists of q projection problems of the form (10). According to Lemma 1, this reduces to q parallel eigenvalue decomposition operations on matrices of sizes $|\mathcal{C}_r|\times|\mathcal{C}_r|$ for $r=1,\ldots,q$.
   (b) *Minimization in terms of* $\{\mathbf{X}_{N;s}\}_{s=0}^p$: This step consists of p unconstrained quadratic programs of sizes $|\mathbf{N}_s|$ for $s=0,1,\ldots,p$. The quadratic programs are parallel and each of them possesses an explicit formula that involves $2|\mathbf{N}_s|$ multiplications.
3) Computation of the dual variables at each iteration, in equation (7c), consists of three parallel steps:
   (a) *Updating* $\{\mathbf{\Lambda}_{C;r}\}_{r=1}^q$: Computational costs for this step involve no multiplications and are negligible.
   (b) *Updating* $\{\mathbf{\Lambda}_{N;s}\}_{s=0}^p$: Computational costs for this step involve no multiplications and are negligible.
   (c) *Updating* $\{\lambda_{z;s}\}_{s=0}^p$: This step is composed of $p+1$ parallel inner product computations, each involving $|\mathbf{N}_s|$ multiplications for $s=0,1,\ldots,p$.

Like any other first-order method, ADMM can be applied to the problem (3) or its various reformulations in many different ways. However, the two-block partitioning into $(\mathcal{P}_1,\mathcal{P}_2)$ is meticulously performed in this work to achieve two goals:

- Making the optimization problem for each block naturally decomposable into disjoint subproblems.
- Making all of the subproblems have explicit closed-form solutions.

These two features distinguish the developed algorithm from a generic ADMM-based algorithm for SDPs, and enable solving large-scale sparse SDPs.

Notation 1. *For every* $\mathbf{D},\mathbf{E}\in\mathbb{H}^n$*, the notation* $\mathbf{D}\oslash_{\mathbf{C}}\mathbf{E}$ *refers to the entrywise division of those entries of* $\mathbf{D}$ *and* $\mathbf{E}$ *that correspond to the ones of* $\mathbf{C}$ *i.e.,*

$$
(\mathbf{D}\oslash_{\mathbf{C}}\mathbf{E})_{ij}\triangleq\begin{cases} D_{ij}/E_{ij} & \text{if } C_{ij}=1\\ 0 & \text{if } C_{ij}=0.\end{cases}
$$

In what follows, we will elaborate on every step of the ADMM iterations:

Block 1: The first step of the algorithm that corresponds to (7a) consists of the operation

$$
\mathcal{P}_1^{k+1} := \underset{\mathcal{P}_1}{\arg\min}\quad \mathcal{L}_\mu(\mathcal{P}_1,\mathcal{P}_2^k,\mathcal{D}^k).
$$

Notice that the minimization of $\mathcal{L}_\mu(\mathcal{P}_1,\mathcal{P}_2^k,\mathcal{D}^k)$ with respect to $\mathcal{P}_1$ is decomposable in terms of the real scalars

$$
\begin{aligned}
\mathrm{Re}\{X_{ij}\} &\quad \text{for}\quad i=1,\ldots,n;\quad j=i,\ldots,n && (16a)\\
\mathrm{Im}\{X_{ij}\} &\quad \text{for}\quad i=1,\ldots,n;\quad j=i+1,\ldots,n && (16b)\\
z_s &\quad \text{for}\quad s=1,\ldots,p && (16c)
\end{aligned}
$$

which leads to an explicit formula.

Block 2: The second step of the algorithm that corresponds to (7b) consists of the operation

$$
\mathcal{P}_2^{k+1} = \underset{\mathcal{P}_2}{\arg\min}\quad \mathcal{L}_\mu(\mathcal{P}_1^{k+1},\mathcal{P}_2,\mathcal{D}^k)
$$

Notice that the minimization of $\mathcal{L}_\mu(\mathcal{P}_1,\mathcal{P}_2^k,\mathcal{D}^k)$ with respect to $\mathcal{P}_1$ is decomposable in terms of the matrix variables $\{\mathbf{X}_{C;r}\}_{r=1}^q$ and $\{\mathbf{X}_{N;s}\}_{s=0}^p$. Hence, the update of $\mathbf{X}_{C;r}$ reduces to the problem (10) for $\widehat{\mathbf{Z}}=\mathbf{X}_{C;r}\{\mathcal{C}_r,\mathcal{C}_r\}$. As shown in Lemma 1, this can be performed via the eigenvalue decomposition of a $|\mathcal{C}_r|\times|\mathcal{C}_r|$ matrix. In addition, the updated value of $\mathbf{X}_{N;s}$ is a minimizer of the function

$$
\begin{aligned}
\mathcal{L}_{N;s}(\mathbf{Z}) =& \|z_s-\langle\mathbf{M}_s,\mathbf{Z}\rangle+\lambda_{z;s}/\mu\|_F^2+\\
&\|\mathbf{X}\circ\mathbf{N}_s-\mathbf{Z}+(1/\mu)\mathbf{\Lambda}_{N;s}\|_F^2 \qquad (18)
\end{aligned}
$$

By taking the derivatives of this function, it is possible to find an explicit formula for $\mathbf{Z}_{\mathrm{opt}}$. Define $\mathcal{L}'_{N;s}(\mathbf{Z})\in\mathcal{S}(\mathbf{N}_s)$ as the gradient of $\mathcal{L}_{N;s}(\mathbf{Z})$ with the following structure:

$$
\mathcal{L}'_{N;s}(\mathbf{Z})\triangleq\left[\frac{\partial\mathcal{L}_{N;s}}{\partial\mathrm{Re}\{Z_{ij}\}}+\mathbf{i}\frac{\partial\mathcal{L}_{N;s}}{\partial\mathrm{Im}\{Z_{ij}\}}\right]_{i,j=1,\ldots,n}
$$

Then, we have

$$
\begin{aligned}
\mathcal{L}'_{N;s}(\mathbf{Z})/2 =& \mathbf{Z}-\mathbf{X}\circ\mathbf{N}_s-(1/\mu)\mathbf{\Lambda}_{N,s}\\
&+(-z_s+\langle\mathbf{M}_s,\mathbf{Z}\rangle-\lambda_{z;s}/\mu)\mathbf{M}_s.
\end{aligned}
$$

Therefore,

$$
\mathbf{Z}_{\mathrm{opt}}=\mathbf{X}\circ\mathbf{N}_s+(1/\mu)\mathbf{\Lambda}_{N,s}+y_s\mathbf{M}_s, \qquad (19)
$$

where $y_s\triangleq z_s-\langle\mathbf{M}_s,\mathbf{Z}^{\mathrm{opt}}\rangle+\lambda_{z;s}/\mu$. Hence, it only remains to derive the scalar y_s, which can be done by inner multiplying

Algorithm 2 Preconditioning of problem (3)

Require: $\{\mathbf{M}_s\}_{s=0}^p, \{l_s\}_{s=1}^p, \{u_s\}_{s=1}^p, \gamma$

$\mathbf{M}_0 \leftarrow \gamma \times \frac{\mathbf{M}_0}{\|\mathbf{M}_0\|_F}$

for $s = 1, \ldots, p$ **do**

  $\mathbf{l}_s \leftarrow \frac{\mathbf{l}_s}{\|\mathbf{M}_s\|_F}$

  $\mathbf{u}_s \leftarrow \frac{\mathbf{u}_s}{\|\mathbf{M}_s\|_F}$

  $\mathbf{M}_s \leftarrow \frac{\mathbf{M}_s}{\|\mathbf{M}_s\|_F}$

end for

return $\{\mathbf{M}_s\}_{s=0}^p, \{l_s\}_{s=1}^p, \{u_s\}_{s=1}^p$

$\mathbf{M}_s$ to both sides of the equation (19).

Closed-form solutions for each step of the over-relaxed ADMM can be derived similarly, which leads to Algorithm 1.

Theorem 2. *Assume that Slater's conditions hold for the decomposable SDP problem* (3). *For* $\alpha = 1$, *the sequence* $\{\mathbf{X}^k\}_{k=0}^\infty$ *generated by Algorithm 1 converges to an optimal solution for* (3).

Proof. The convergence of both primal and dual variables is guaranteed for a standard ADMM problem if the matrix $\mathbf{B}$ in (5b) has full column rank [61]. After realizing that (1) is obtained from a two-block ADMM procedure, the theorem can be concluded form the fact that the equivalent of $\mathbf{B}$ for the algorithm (1) is a mapping from the variables $\{\mathbf{X}_{C;r}\}_{r=1}^q$ and $\{\mathbf{X}_{N;s}\}_{s=0}^p$ to

$$\{\mathbf{X}_{C;r}\}_{r=1}^q, \{\mathbf{X}_{N;s}\}_{s=0}^p \quad \text{and} \quad \{\langle \mathbf{M}_s, \mathbf{X}_{N;s}\rangle\}_{s=0}^p$$

which is not singular, i.e., it has full column rank. The details are omitted for brevity. □

C. Parameter Selection and Preconditioning

The performance of the proposed algorithm somewhat depends on the parameter μ. Since this algorithm is based on a two-block ADMM technique with the blocks $\mathcal{P}_1$ and $\mathcal{P}_2$, one may directly use the existing results for parameter selection in two-block ADMM, such as the techniques developed in [62] and [25]. However, given the complexity of finding an optimal value of μ, it may be more efficient to instead take more iterations with a sub-optimal value μ to reduce the overall runtime by avoiding the preprocessing time needed for designing μ. Another issue is that first-order methods, including ADMM, are sensitive to the condition number of the problem data. One may optimally precondition the derived two-block ADMM by adopting the existing methods, such as [22], which is a daunting task in general. In this paper, we resort to a simple preconditioning technique, described in Algorithm 2, to normalize different parts of the data. More precisely, the constraints in (3b) are normalized and the objective function is rescaled according to a tuning factor γ.

V. Optimal Power Flow

Consider an n-bus electrical power network with the topology described by a simple graph $\mathcal{H} = (\mathcal{V}_\mathcal{H}, \mathcal{E}_\mathcal{H})$, meaning that each vertex belonging to $\mathcal{V}_\mathcal{H} = \{1, \ldots, n\}$ represents a node of the network and each edge belonging to $\mathcal{E}_\mathcal{H} = \{1, \ldots, m\}$ represents a transmission line. Define $\mathbf{C}_f \in \{0,1\}^{m\times n}$ and $\mathbf{C}_t \in \{0,1\}^{m\times n}$ to be the "from" and "to" incidence matrices of the network graph, respectively, i.e., the (l,k) entry of $\mathbf{C}_f$ is equal to one if and only if k is the starting node of the branch l, whereas the (l,k) entry of $\mathbf{C}_t$ is equal to one if and only if k is the ending node of l. Define $\mathcal{G} = \{1, \ldots, u\}$ to be the set of generating units, and let $\mathbf{C}_g \in \{0,1\}^{u\times n}$ denote the unit incidence matrix, i.e., the (g,k) entry of $\mathbf{C}_g$ is 1 if and only if unit g is located at bus k. Additionally, let $\mathbf{Y} \in \mathbb{C}^{n\times n}$ denote the admittance matrix of the network, and define $\mathbf{Y}_f \in \mathbb{C}^{m\times n}$ and $\mathbf{Y}_t \in \mathbb{C}^{m\times n}$ to be the "from" and "to" admittance matrices, respectively. Define $\mathbf{V} \in \mathbb{C}^n$ as the unknown voltage phasor vector, i.e., V_k is the voltage phasor for node $k \in \mathcal{V}_\mathcal{H}$. Let $\mathbf{P} + \mathbf{Q}\mathbf{i}$ represent the vector of complex power supply by generating units, where $\mathbf{P} \in \mathbb{R}^u$ and $\mathbf{Q} \in \mathbb{R}^u$ are the vectors of active and reactive powers, respectively. Define $\mathbf{S}_\mathrm{d} \in \mathbb{C}^n$ to be the vector of nodal complex power demand. Finally, define $\mathbf{s}_f = [s_{f;1}, s_{f;2}, \ldots, s_{f;m}]^\top$ and $\mathbf{s}_t = [s_{t;1}, s_{t;2}, \ldots, s_{t;m}]^\top$ to be the vectors of complex power entering the starting and ending nodes of branches, respectively.

The classical OPF problem can be described as follows:

$$\underset{\substack{\mathbf{V}\in\mathbb{C}^n \\ \mathbf{P},\mathbf{Q}\in\mathbb{R}^u \\ \mathbf{s}_f,\mathbf{s}_f\in\mathbb{C}^m}}{\text{minimize}} \quad \sum_{g\in\mathcal{G}} c_{2;g}P_g^2 + c_{1;g}P_g + c_{0;g} \tag{20a}$$

$$\text{subject to} \quad V_k^{\min} \leq |V_k| \leq V_k^{\max}, \qquad k \in \mathcal{V}_\mathcal{H}, \tag{20b}$$

$$Q_g^{\min} \leq Q_g \leq Q_g^{\max}, \qquad g \in \mathcal{G}, \tag{20c}$$

$$P_g^{\min} \leq P_g \leq P_g^{\max}, \qquad g \in \mathcal{G}, \tag{20d}$$

$$\mathbf{C}_g^\mathrm{T}(\mathbf{P} + \mathbf{Q}\mathbf{i}) = \mathrm{diag}\{\mathbf{V}\mathbf{V}^*\mathbf{Y}^*\} + \mathbf{S}_\mathrm{d}, \tag{20e}$$

$$|s_{f;l}| \leq s_l^{\max}, \qquad l \in \mathcal{E}_\mathcal{H}, \tag{20f}$$

$$|s_{t;l}| \leq s_l^{\max}, \qquad l \in \mathcal{E}_\mathcal{H}, \tag{20g}$$

$$\mathbf{s}_f = \mathrm{diag}\{\mathbf{C}_f\mathbf{V}\mathbf{V}^*\mathbf{Y}_f^*\}, \tag{20h}$$

$$\mathbf{s}_t = \mathrm{diag}\{\mathbf{C}_t\mathbf{V}\mathbf{V}^*\mathbf{Y}_t^*\}, \tag{20i}$$

where $V_k^{\min}$, $V_k^{\max}$, $P_g^{\min}$, $P_g^{\max}$, $Q_g^{\min}$, $Q_g^{\max}$ and $s_l^{\max}$ are constant limits, and $c_{2;g} \geq 0$, $c_{1;g}$ and $c_{0;g}$ are coefficients accounting for the cost of producing power by unit g. More details on a general formulation may be found in [7].

OPF is a highly non-convex problem, which is known to be difficult to solve in general. However, the constraints of problem (20) can all be expressed as linear functions of the entries of the quadratic matrix $\mathbf{V}\mathbf{V}^*$. This implies that the constraints of OPF are linear in terms of a matrix variable $\mathbf{W} \triangleq \mathbf{V}\mathbf{V}^*$. One can reformulate OPF by replacing each monomial $V_iV_j^*$ with the new variable W_{ij} and represent the constraints in the form of problem (1) with a representative graph that is isomorphic to the network topology graph $\mathcal{H}$. In order to preserve the equivalence of the two formulations, two additional constraints must be added to the problem: (i) $\mathbf{W} \succeq 0$, (ii) $\mathrm{rank}\{\mathbf{W}\} = 1$. If we drop the rank condition as the only non-convex constraint of the reformulated OPF problem, we attain the SDP relaxation of OPF as follows:

$$\begin{aligned}
\underset{\substack{\mathbf{W}\in\mathbb{H}^n\\ \mathbf{P}',\mathbf{P},\mathbf{Q}\in\mathbb{R}^u\\ \mathbf{s}_f,\mathbf{s}_f\in\mathbb{C}^m}}{\text{minimize}} \quad & \sum_{g\in\mathcal{G}} c_{2;g}P'_g + c_{1;g}P_g + c_{0;g} && && \text{(21a)}\\
\text{subject to} \quad & (V_k^{\min})^2 \le W_{kk} \le (V_k^{\max})^2, && k\in\mathcal{V}_{\mathcal{H}}, && \text{(21b)}\\
& Q_g^{\min} \le Q_g \le Q_g^{\max}, && g\in\mathcal{G}, && \text{(21c)}\\
& P_g^{\min} \le P_g \le P_g^{\max}, && g\in\mathcal{G}, && \text{(21d)}\\
& \mathbf{P}+\mathbf{Q}\mathbf{i} = \operatorname{diag}\{\mathbf{W}\mathbf{Y}^*\} + \mathbf{S}_{\mathrm{d}}, && && \text{(21e)}\\
& \mathbf{s}_f = \operatorname{diag}\{\mathbf{C}_f\mathbf{W}\mathbf{Y}_f^*\}, && && \text{(21f)}\\
& \mathbf{s}_t\ = \operatorname{diag}\{\mathbf{C}_t\,\mathbf{W}\mathbf{Y}_t^*\}, && && \text{(21g)}\\
& \begin{bmatrix} P'_g & P_g\\ P_g & 1\end{bmatrix} \succeq 0, && g\in\mathcal{G}, && \text{(21h)}\\
& \begin{bmatrix} s_l^{\max} & s_{f;l}\\ s_{f;l}^* & s_l^{\max}\end{bmatrix} \succeq 0, && l\in\mathcal{E}_{\mathcal{H}}, && \text{(21i)}\\
& \begin{bmatrix} s_l^{\max} & s_{t;l}\\ s_{t;l}^* & s_l^{\max}\end{bmatrix} \succeq 0, && l\in\mathcal{E}_{\mathcal{H}}, && \text{(21j)}\\
& \mathbf{W} \succeq 0, && && \text{(21k)}
\end{aligned}$$

where the auxiliary variable P'_g accounts for the square of P_g for each $g\in\mathcal{G}$. Note that the above problem is an SDP because its objective function is linear and its constraints are linear scalar/matrix equalities and inequalities in the variables of the problem [11]. Therefore, it can be put into the canonical form (1) (please refer to [7] for details on such a reformulation). As stated in the introduction, several papers in the literature have shown great promises for finding global or near-global solutions of OPF using the above relaxation or a penalized version of the SDP relaxation. The major drawback of relaxing the OPF problem to SDP is the requirement of defining a matrix variable, which makes the number of scalar variables of the problem quadratic with respect to the number of network buses. However, we have shown in [17] that real-world grids would have a low treewidth, e.g., at most 26 for the Polish test system with over 3000 buses. This makes our proposed numerical algorithm scalable and highly parallelizable for the above SDP relaxation. As an example, the SDP relaxation of OPF for a large-scale European grid with 9241 buses amounts to simple operations over 857 matrices of size 31 by 31, as well as 14035 matrices of size 2 by 2.

VI. Simulation Results

In this section, we evaluate the performance of the proposed algorithm for solving the SDP relaxation of OPF over Pan European Grid Advanced Simulation and State Estimation (PEGASE) test systems [63], [64]. All simulations are run in MATLAB using a system with an Intel 3.0 GHz, 12-core CPU and 256 GB RAM. The overall running time of 5000 iteration is between 8 and 50 minutes in a MATLAB implementation without parallelization. For larger cases, running time diminishes using MATLAB parallel computing toolbox and could be further reduced in C++.

In all of the experiments, a tree decomposition of the sparsity graph is obtained in less than 90 seconds, using the algorithm described in [54]. The experiment results are summarized in Table I. The second column indicates the total number of upper and lower bounds of the form (3b) (i.e., $2\times p$). The number of positive-semidefinite submatrices of the form (3c) and their maximum size are shown in the third and forth columns, respectively. Notice that since no thermal limits are imposed for the case 13659-bus system, the number of constraints and bags for this case are smaller, compared to its preceding system. The over-relaxation parameter $\alpha = 1.8$ is used for all cases. The quality of solutions obtained via 5000 ADMM iterations is given in columns 7 to 10. As shown in Figure 2, the residue function ε^k (as defined in (9)) is monotonically decreasing for all simulated cases. The convergence behavior of the ADMM coupling constraint (i.e., $\|\mathbf{A}\mathbf{x}^{5000}+\mathbf{B}\mathbf{y}^{5000}-\mathbf{c}\|_2$) and the cost value for different cases are depicted in Figure 2, as well. In order to further assess the quality of solutions after 5000 ADMM iterations, the maximum violation of inequality constraints in (3b) and the largest absolute value among the negative eigenvalues of all submatrices $\mathbf{X}^{5000}\{\mathcal{C}_r,\mathcal{C}_r\}$ are given in the eighth and ninth columns. Small-sized bags, corresponding to submatrices of $\mathbf{W}$ in (21), are combined to obtain a modest number of bags. To elaborate on the algorithm, note that every iteration amounts to a basic matrix operation or an eigendecomposition over matrices of size at most 31×31 for the PEGASE 13659-bus system.

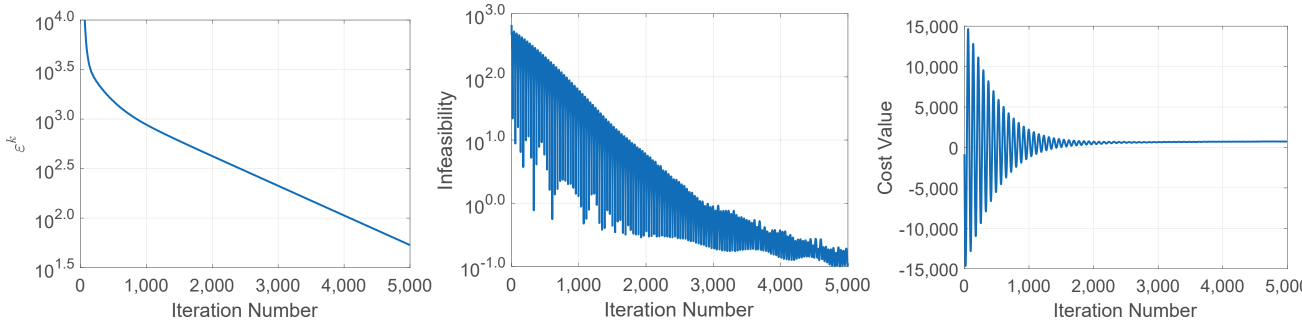
The effect of different choices for tuning parameter μ on the convergence of the objective value for the 13659-bus system is illustrated in Figure 3a. Additionally, Figure 3b compares different choices of the over-relaxation parameter for the 2869-bus system. Further preconditioning efforts, in addition to the one suggested in Algorithm 2, may reduce the number of iterations (note that OPF is often very ill-conditioned due to high inductance-to-resistance ratios), and this is left for future work.

VII. Conclusion

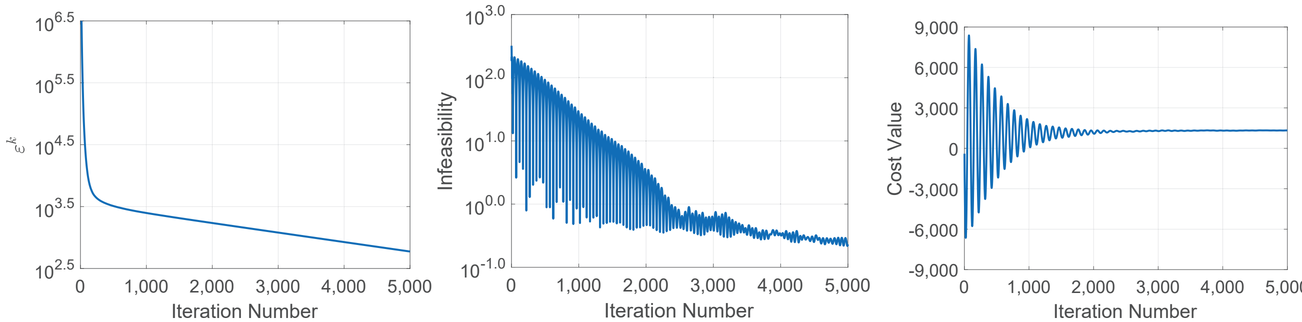
Motivated by the application of sparse semidefinite programming in many hard optimization problems across engineering, the objective of this work is to design a fast and parallelizable numerical algorithm for solving sparse semidefinite programs (SDPs). To this end, the underling sparsity structure of a given SDP problem is captured using a tree decomposition technique, leading to a decomposed SDP problem. A highly distributed/parallelizable numerical algorithm is developed for solving the decomposed SDP, based on the alternating direction method of multipliers (ADMM). Each iteration of the designed algorithm has a closed-form solution, which involves multiplications and eigenvalue decompositions over certain submatrices induced by the tree decomposition of the sparsity graph. The proposed algorithm is applied to the classical optimal power flow problem, and also evaluated on large-scale PEGASE benchmark systems. The numerical technique developed in this paper enables solving complex models of various power optimization problems, such as optimal power flow, unit commitment and state estimation, through conic optimization.

Test cases	Number of inequality constraints	Number of bags	Maximum size of bags	μ	γ	Linear coupling violation	Maximum inequality violation	Maximum PSD violation	ADMM cost value	Running time without parallelization	Running time with parallelization
Case 1354-bus	17238	3398	13	1200	1e-6	1.0e-1	1.4e-2	6.2e-4	738.10	8 min	7 min
Case 2869-bus	34694	6637	13	2500	1e-6	2.3e-1	7.9e-2	7.6e-4	1328.52	14 min	10 min
Case 9241-bus	96108	14892	31	3200	1e-7	4.2e-1	2.1e-1	4.5e-4	3140.25	39 min	18 min
Case 13659-bus	90140	5185	31	3000	1e-6	5.8e-1	1.6e-1	1.5e-4	3850.71	50 min	24 min

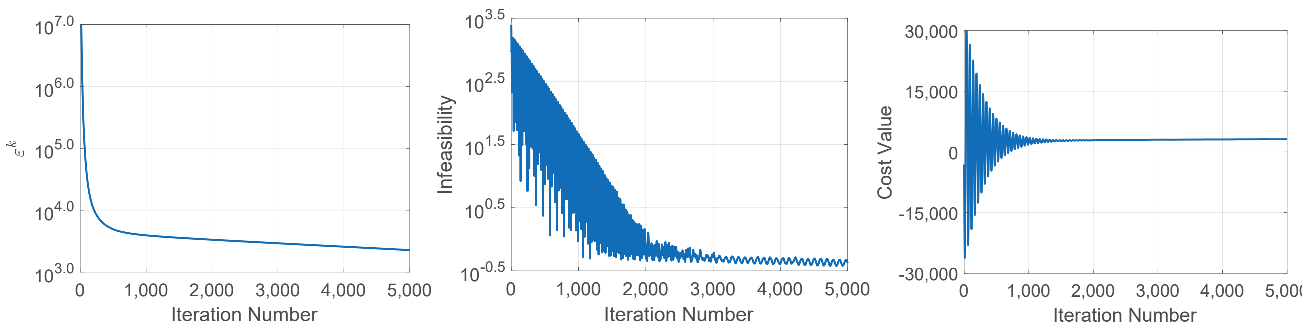
TABLE I: Performance of the proposed algorithm for solving the SDP relaxation of the OPF problem on PEGASE test cases via serial and parallel implementations.



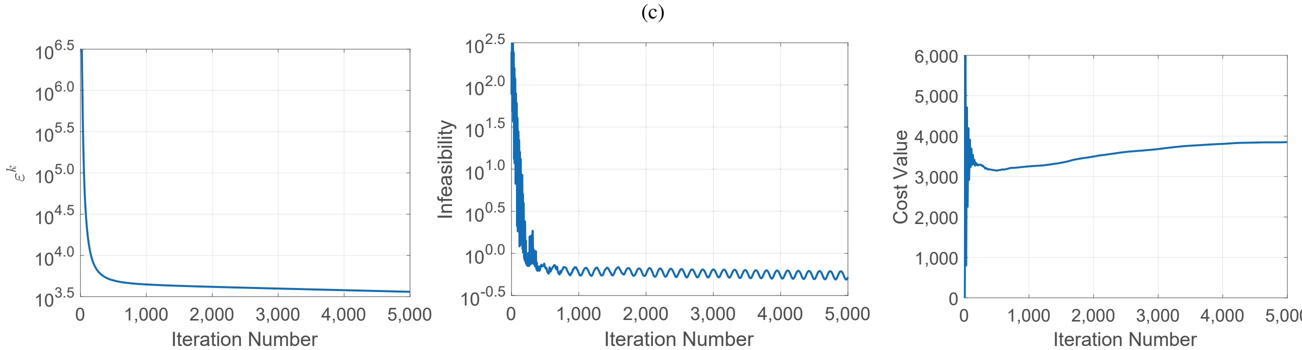
(a)



(b)

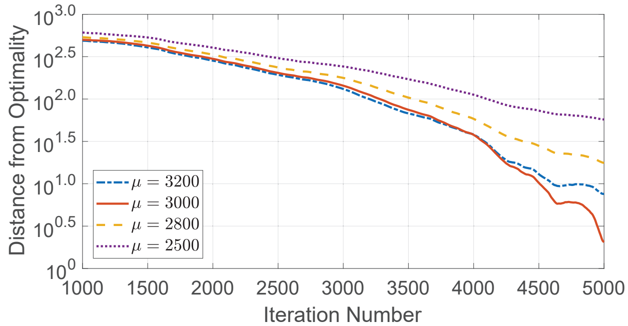


(c)

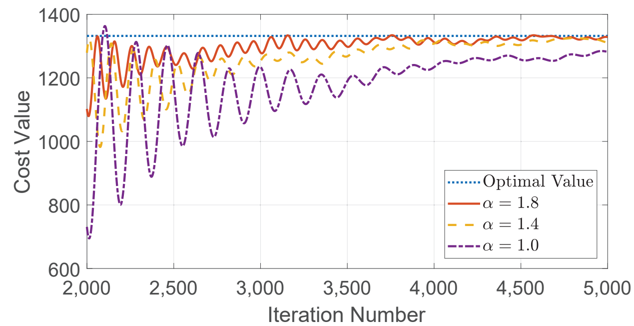


(d)

Fig. 2: These plots show the convergence behavior of the residue functions, indefeasibly sequences and cost values for PEGASE test systems. (a): 1354-bus system, (b): 2869-bus system, (c): 9241-bus system, (d): 13659-bus system.



(a)



(b)

Fig. 3: The effect of different choices for tuning parameters on the convergence of cost value, (a): 13659-bus system, (b): 2869-bus system.

References

[1] F. Alizadeh-Dehkharghani, "Combinatorial optimization with interior point methods and semi-definite matrices," Ph.D. dissertation, University of Minnesota, 1992.

[2] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*. SIAM, 1994, vol. 13.

[3] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM (JACM)*, vol. 42, no. 6, pp. 1115–1145, 1995.

[4] S. P. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory*. SIAM, 1994, vol. 15.

[5] Y. Nesterov, "Semidefinite relaxation and nonconvex quadratic optimization," *Optimization methods and software*, vol. 9, no. 1-3, pp. 141–160, 1998.

[6] Z.-Q. Luo, "Applications of convex optimization in signal processing and digital communication," *Mathematical Programming*, vol. 97, no. 1-2, pp. 177–207, 2003.

[7] J. Lavaei and S. Low, "Zero duality gap in optimal power flow problem," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 92–107, 2012.

[8] T.-H. Chang, Z.-Q. Luo, and C.-Y. Chi, "Approximation bounds for semidefinite relaxation of max-min-fair multicast transmit beamforming problem," *IEEE Transactions on Signal Processing*, vol. 56, no. 8, pp. 3932–3943, 2008.

[9] H. D. Mittelmann and F. Vallentin, "High-accuracy semidefinite programming bounds for kissing numbers," *Experimental Mathematics*, vol. 19, no. 2, pp. 175–179, 2010.

[10] E. de Klerk and R. Sotirov, "Improved semidefinite programming bounds for quadratic assignment problems with suitable symmetry," *Mathematical Programming*, vol. 133, no. 1-2, pp. 75–91, 2012.

[11] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, 1996.

[12] K. Fujisawa, K. Nakata, M. Yamashita, and M. Fukuda, "SDPA project: Solving large-scale semidefinite programs," *Journal of the Operations Research Society of Japan*, vol. 50, no. 4, pp. 278–298, 2007.

[13] K. Fujisawa, H. Sato, S. Matsuoka, T. Endo, M. Yamashita, and M. Nakata, "High-performance general solver for extremely large-scale semidefinite programming problems," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2012.

[14] M. S. Andersen, J. Dahl, and L. Vandenberghe, "Implementation of nonsymmetric interior-point methods for linear optimization over sparse matrix cones," *Mathematical Programming Computation*, vol. 2, no. 3-4, pp. 167–201, 2010.

[15] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota, "Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results," *Mathematical Programming*, vol. 95, no. 2, pp. 303–327, 2003.

[16] M. Fukuda, M. Kojima, K. Murota, and K. Nakata, "Exploiting sparsity in semidefinite programming via matrix completion I: General framework," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 647–674, 2001.

[17] R. Madani, M. Ashraphijuo, and J. Lavaei, "Promises of conic relaxation for contingency-constrained optimal power flow problem," *IEEE Transactions on Power Systems*, vol. 31, no. 2, pp. 1297–1307, 2016.

[18] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17 – 40, 1976.

[19] R. Glowinsk and A. Marroco, "Sur l'approximation, par lments finis d'ordre un, et la rsolution, par pnalisation-dualit d'une classe de problmes de dirichlet non linaires," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 9, no. R2, pp. 41–76, 1975.

[20] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, "Fast alternating direction optimization methods," *SIAM Journal on Imaging Sciences*, vol. 7, no. 3, pp. 1588–1623, 2014.

[21] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," *Soviet Mathematics Doklady*, vol. 27, no. 2, pp. 372–376, 1983.

[22] P. Giselsson and S. Boyd, "Diagonal scaling in Douglas-Rachford splitting and ADMM," in *IEEE Conference on Decision and Control*, 2014.

[23] B. He and X. Yuan, "On the $O(1/n)$ convergence rate of the Douglas–Rachford alternating direction method," *SIAM Journal on Numerical Analysis*, vol. 50, no. 2, pp. 700–709, 2012.

[24] R. D. C. Monteiro and B. F. Svaiter, "Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers," *SIAM Journal on Optimization*, vol. 23, no. 1, pp. 475–507, 2013.

[25] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. I. Jordan, "A general analysis of the convergence of ADMM," in *International Conference on Machine Learning*, 2015.

[26] Z. Wen, D. Goldfarb, and W. Yin, "Alternating direction augmented Lagrangian methods for semidefinite programming," *Mathematical Programming*, vol. 2, no. 3-4, pp. 203–230, 2010.

[27] Y. Sun, M. S. Andersen, and L. Vandenberghe, "Decomposition in conic optimization with partially separable structure," *SIAM Journal on Optimization*, vol. 24, no. 2, pp. 873–897, 2014.

[28] R. A. Jabr, "Exploiting sparsity in SDP relaxations of the OPF problem," *IEEE Transactions on Power Systems*, vol. 27, no. 2, pp. 1138–1139, 2012.

[29] Y. Weng, Q. Li, R. Negi, and M. Ilić, "Distributed algorithm for SDP state estimation," in *Innovative Smart Grid Technologies*, 2013.

[30] A. Hauswirth, T. Summers, J. Warrington, J. Lygeros, A. Kettner, and A. Brenzikofer, "A modular AC optimal power flow implementation for distribution grid planning," in *IEEE PowerTech*, 2015.

[31] H. Zhu and G. B. Giannakis, "Power system nonlinear state estimation using distributed semidefinite programming," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 6, pp. 1039–1050, 2014.

[32] E. Dall'Anese, H. Zhu, and G. B. Giannakis, "Distributed optimal power flow for smart microgrids," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1464–1475, 2013.

[33] A. Kalbat and J. Lavaei, "A fast distributed algorithm for decomposable semidefinite programs," in *IEEE Conference on Decision and Control*, 2015.

[34] Q. Peng and S. H. Low, "Distributed optimal power flow algorithm for radial networks, I: Balanced single phase case," to appear in *IEEE Transactions on Smart Grid*, 2016.

[35] R. Madani, A. Kalbat, and J. Lavaei, "ADMM for sparse semidefinite programming with applications to optimal power flow problem," in *IEEE Conference on Decision and Control*, 2015.

[36] R. Madani, S. Sojoudi, and J. Lavaei, "Convex relaxation for optimal power flow problem: Mesh networks," *IEEE Transactions on Power Systems*, vol. 30, no. 1, pp. 199–211, 2015.

[37] C. Josz and D. K. Molzahn, "Moment/sum-of-squares hierarchy for complex polynomial optimization," *arXiv preprint arXiv:1508.02068*, 2015.

[38] S. Fattahi, M. Ashraphijuo, J. Lavaei, and A. Atamtürk, "Conic relaxations of the unit commitment problem," *Energy*, vol. 134, pp. 1079–1095, 2017.

[39] J. Momoh, R. Adapa, and M. El-Hawary, "A review of selected optimal power flow literature to 1993. I. nonlinear and quadratic programming approaches," *IEEE Transactions on Power Systems,*, vol. 14, no. 1, pp. 96–104, 1999.

[40] J. Carpentier, "Contribution a 1 etude du dispatching economique," *Bulletin Society Francaise Electricians*, vol. 3, no. 8, pp. 431–447, 1962.

[41] M. B. Cain, R. P. O'Neill, and A. Castillo, "History of optimal power flow and formulations," *Technical Report*, 2012.

[42] S. Sojoudi and J. Lavaei, "Physics of power networks makes hard optimization problems easy to solve," in *IEEE Power and Energy Society General Meeting*, 2012.

[43] S. H. Low, "Convex relaxation of optimal power flow–Part I: Formulations and equivalence," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 15–27, 2014.

[44] J. Lavaei, D. Tse, and B. Zhang, "Geometry of power flows and optimization in distribution networks," *IEEE Transactions on Power Systems*, vol. 29, no. 2, pp. 572–583, 2014.

[45] S. Sojoudi and J. Lavaei, "Exactness of semidefinite relaxations for nonlinear optimization problems with underlying graph structure," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 1746–1778, 2014.

[46] L. Gan, N. Li, U. Topcu, and S. H. Low, "Optimal power flow in distribution networks," *IEEE Conference on Decision and Control*, 2013.

[47] R. Y. Zhang, C. Josz, and S. Sojoudi, "Conic optimization theory: Convexification techniques and numerical algorithms," *arXiv preprint arXiv:1709.08841*, 2017.

[48] R. Madani, J. Lavaei, and R. Baldick, "Convexification of power flow problem over arbitrary networks," in *IEEE Conference on Decision and Control*, 2015.

[49] R. Madani, J. Lavaei, R. Baldick, and A. Atamturk, "Power system state estimation and bad data detection by means of conic relaxation," in *Hawaii International Conference on System Sciences*, 2017.

[50] S. Sojoudi and S. H. Low, "Optimal charging of plug-in hybrid electric vehicles in smart grids," in *IEEE Power and Energy Society General Meeting*, 2011.

[51] R. Madani, S. Sojoudi, G. Fazelnia, and J. Lavaei, "Finding low-rank solutions of sparse linear matrix inequalities using convex optimization," *SIAM Journal on Optimization*, vol. 27, no. 2, pp. 725–758, 2017.

[52] R. Halin, "S-functions for graphs," *Journal of geometry*, vol. 8, no. 1-2, pp. 171–186, 1976.

[53] J. R. Gilbert, "Some nested dissection order is nearly optimal," *Information Processing Letters*, vol. 26, no. 6, pp. 325–328, 1988.

[54] H. L. Bodlaender and A. M. Koster, "Treewidth computations I. Upper bounds," *Information and Computation*, vol. 208, no. 3, pp. 259–275, 2010.

[55] ——, "Treewidth computations II. Lower bounds," *Information and Computation*, vol. 209, no. 7, pp. 1103–1119, 2011.

[56] R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz, "Positive definite completions of partial Hermitian matrices," *Linear Algebra and its Applications*, vol. 58, pp. 109–124, 1984.

[57] M. Laurent, "Polynomial instances of the positive semidefinite and Euclidean distance matrix completion problems," *SIAM Journal on Matrix Analysis and Applications*, vol. 22, no. 3, pp. 874–894, 2001.

[58] M. Laurent and A. Varvitsiotis, "A new graph parameter related to bounded rank positive semidefinite matrix completions," *Mathematical Programming*, vol. 145, no. 1-2, pp. 291–325, 2014.

[59] 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*, vol. 3, no. 1, pp. 1–122, 2011.

[60] N. J. Higham, "Computing a nearest symmetric positive semidefinite matrix," *Linear Algebra and its Applications*, vol. 103, pp. 103–118, 1988.

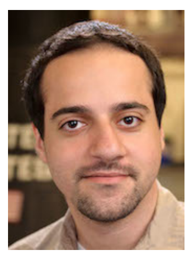
[61] B. He and X. Yuan, "On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers," *Numerische Mathematik*, vol. 130, no. 3, pp. 567–577, 2014.

[62] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (ADMM): quadratic problems," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 644–658, 2015.

[63] S. Fliscounakis, P. Panciatici, F. Capitanescu, and L. Wehenkel, "Contingency ranking with respect to overloads in very large power systems taking into account uncertainty, preventive, and corrective actions," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4909–4917, 2013.

[64] C. Josz, S. Fliscounakis, J. Maeght, and P. Panciatici, "AC Power Flow Data in Matpower and QCQP Format: iTesla, RTE Snapshots, and PEGASE," *arXiv preprint arXiv:1603.01533*, 2016.

Ramtin Madani is an Assistant Professor at the Electrical Engineering Department of the University of Texas at Arlington. He received the Ph.D. degree in Electrical Engineering from Columbia University in 2015, and was a postdoctoral scholar in the Department of Industrial Engineering and Operations Research at University of California, Berkeley in 2016.



Abdulrahman Kalbat is an Assistant Professor in the Electrical Engineering Department at United Arab Emirates University. He received the Ph.D. degree in Electrical Engineering from Columbia University in 2015.

Javad Lavaei (SM'10) is an Assistant Professor in the Department of Industrial Engineering and Operations Research at University of California, Berkeley. He obtained his Ph.D. degree in Control and Dynamical Systems from California Institute of Technology in 2011. He has won multiple awards, including NSF CAREER Award, Office of Naval Research Young Investigator Award, AFOSR Young Faculty Award, DARPA Young Faculty Award, Google Faculty Research Award, Donald P. Eckman Award, Resonate Award, INFORMS Optimization Society Prize for Young Researchers, INFORMS ENRE Energy Best Publication Award, and SIAM Control and Systems Theory Prize.