ADMM for Sparse Semidefinite Programming with Applications to Optimal Power Flow Problem

Ramtin Madani, Abdulrahman Kalbat and Javad Lavaei
Department of Electrical Engineering, Columbia University

***Abstract*—This paper designs a distributed algorithm for solving sparse semidefinite programming (SDP) problems, based on the alternating direction method of multipliers (ADMM). It is known that exploiting the sparsity of a large-scale SDP problem leads to a decomposed formulation with a lower computational cost. The algorithm proposed in this work solves the decomposed formulation of the SDP problem using an ADMM scheme whose iterations consist of two subproblems. Both subproblems are highly parallelizable and enjoy closed-form solutions, which make the iterations computationally very cheap. The developed numerical algorithm is also applied to the SDP relaxation of the optimal power flow (OPF) problem, and tested on the IEEE benchmark systems.**

I. Introduction

While small- to medium-sized semidefinite programs (SDP) are efficiently solvable by second-order-based interior point methods in polynomial time up to any arbitrary precision [1], these methods are impractical for solving large-scale SDPs due to computation time and memory issues. A promising approach 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 by [2] and [3]. While second-order methods are capable of achieving high accuracy via expensive iterations, a modest accuracy can be achieved through tens of ADMM's low-complex iterations. In order to reach high accuracy in reasonable number of iterations, great effort has been devoted to accelerating ADMM through Nesterov's scheme [4], [5]. Because of the sensitivity of the gradient methods to the condition number of the problem's data, diagonal rescaling is proposed in [6] for a class of problems to improve the performance of ADMM. The $\mathcal{O}(\frac{1}{n})$ worst-case convergence rate of ADMM is proven in [7], [8] under certain assumptions.

In light of the scalability of ADMM, the main objective of this work is to design an ADMM-based parallel algorithm for solving sparse large-scale SDPs, with a guaranteed convergence under very 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 large-scale SDP formulation using an explicit formula.

This work is related to and improves upon some recent papers in this area. [9] applies ADMM to the dual SDP formulation, leading to a centralized algorithm that is not parallelizable and is computationally expensive for large-scale SDPs. [10] 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, [11] solves the decomposed SDP problem created by [10], but the algorithm needs to solve an optimization subproblem at every iteration. In contrast with the above papers, the algorithm proposed in this work is composed of low-complex and parallelizable iterations, which run fast if the treewidth of the representative graph of the SDP problem is small. Since this treewidth is low for real-world power networks, our algorithm is well suited for the SDP relaxation of power optimization problems, and indeed this is the main motivation behind this work. This will be explained below.

A. Optimal Power Flow Problem

The optimal power flow (OPF) problem 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 [12], [13]. Motivated by the importance of this fundamental problem for operation and planning as well as the potential monetary savings involved [14], many optimization techniques have been explored for the OPF problem. Due to the non-convexity and NP-hardness of OPF, the existing algorithms are not robust, lack performance guarantees and may not find a global optimum. With the goal of designing a polynomial-time algorithm that finds a global solution for OPF, [15] derives an SDP relaxation for OPF, which results in a globally optimal solution if the duality gap is zero. The proposed relaxation can find near-global solutions with global optimality guarantees of at least 99% for IEEE and Polish systems [16], and is theoretically proven to be exact under various assumptions [17], [18], [19], [20], [21], [22]. However, this relaxation is a high-dimensional SDP problem, which imposes some limitations on its practicality for real-world networks.

The emerging smart grid paradigm and the integration of intermittent and distributed power generation calls for the development of efficient, scalable, and parallel algorithms for solving large-scale OPF problems to enable real-time network

Email: madani@ee.columbia.edu, ak3369@columbia.edu and lavaei@ee.columbia.edu.

This work was supported by the ONR YIP Award, NSF CAREER Award 1351279, and NSF EECS Award 1406865.

management and improve the system's reliability. In response to this need, we aim to design an algorithm that is able to solve large-scale SDP relaxations. Early efforts to solve OPF in a distributed way (without considering non-convexity) can be traced back to [23], [24]. In [25], a fully decentralized ADMM-based algorithm is developed for a convex approximation of dynamic OPF. The papers [26] and [27] exploit primal-dual decomposition and ADMM methods for the SDP relaxation of OPF, but they need to solve an expensive SDP sub-problem at every iteration. The work [28] designs a distributed algorithm for a second-order cone relaxation of OPF over radial (acyclic) networks. In contrast to the existing methods, the algorithm to be proposed here applies to both distribution and transmission networks, and does not require solving any optimization sub-problem at any iteration.

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 Frobinous 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 $\mathrm{Re}\{\mathbf{W}\}$, $\mathrm{Im}\{\mathbf{W}\}$, $\mathrm{rank}\{\mathbf{W}\}$, and $\mathrm{diag}\{\mathbf{W}\}$ denote the real part, imaginary part, rank, and diagonal of a Hermitian matrix $\mathbf{W}$, respectively. Given a vector $\mathbf{v}$, the notation $\mathrm{diag}\{\mathbf{v}\}$ 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 $(\cdot)_{\mathrm{opt}}$ is used to show the optimal value of an optimization variable. 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. 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 $W_{12}, W_{13}, W_{22}, W_{23}$.

II. Preliminaries

Consider the semidefinite program

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

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 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. *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.*

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)**.** *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 a 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**.

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 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 X are referred to as *important entries.* Once the optimal values of the important entries of X are found using an arbitrary algorithm, the remaining entries can be obtained from an explicit (recursive) 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. Nevertheless, there are many efficient algorithms in the literature that find near-optimal tree decompositions (specially for power networks due to their near planarity) [29], [30].

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 $\mathbf{N}$. Also, define $\mathcal{S}(\mathbf{N})$ as*

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

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 $r \in \{1, \ldots, p\}$.*
- *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 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

$$\begin{aligned} \underset{\mathbf{X} \in \mathcal{S}(\mathbf{C})}{\text{minimize}} \quad & \langle \mathbf{X}, \mathbf{M}_0 \rangle && \text{(2a)} \\ \text{subject to} \quad & l_s \leq \langle \mathbf{X}, \mathbf{M}_s \rangle \leq u_s, \quad s = 1, \ldots, p, && \text{(2b)} \\ & \mathbf{X}\{\mathcal{C}_r, \mathcal{C}_r\} \succeq 0, \quad r = 1, \ldots, q && \text{(2c)} \end{aligned}$$

which is referred to as *decomposed SDP* throughout this paper. Due to the chordal theorem [31], problems (1) and (2) lead to the same optimal objective value. Furthermore, if $\mathbf{X}_{\text{ref}} \in \mathcal{S}(\mathbf{C})$ denotes an arbitrary solution of the decomposed SDP problem (2), then there exists a solution $\mathbf{X}_{\text{opt}}$ to the SDP problem (1) such that $\mathbf{X}_{\text{opt}} \circ \mathbf{C} = \mathbf{X}_{\text{ref}}$.

To understand how $\mathbf{X}_{\text{opt}}$ can be constructed from $\mathbf{X}_{\text{ref}}$, observe that those entries of $\mathbf{X}$ corresponding to the zeros of $\mathbf{C}$ are 0 due to the relation $\mathbf{X}_{\text{ref}} \in \mathcal{S}(\mathbf{C})$. These entries of the matrix variable $\mathbf{X}$ that are needed for SDP but have not been found by decomposed SDP are referred to as *missing entries*. Several completion approaches can be adopted in order to recover these missing entries. An algorithm is proposed in [10], [32] that obtains a completion for $\mathbf{X}_{\text{ref}}$ within the set

$$\{\mathbf{X} \in \mathbb{H}^n \mid \mathbf{X} \circ \mathbf{C} = \mathbf{X}_{\text{ref}},\ \mathbf{X} \succeq 0\}$$

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 [33], [34]. In [35], we extended this result to the complex domain by proposing a recursive algorithm that transforms $\mathbf{X}_{\text{ref}} \in \mathcal{S}(\mathbf{C})$ into a solution $\mathbf{X}_{\text{opt}}$ for the original SDP problem (1) whose rank is upper bounded by the maximum rank among the matrices $\mathbf{X}_{\text{ref}}\{\mathcal{C}_1, \mathcal{C}_1\}, \mathbf{X}_{\text{ref}}\{\mathcal{C}_2, \mathcal{C}_2\}, \ldots, \mathbf{X}_{\text{ref}}\{\mathcal{C}_q, \mathcal{C}_q\}$. This algorithm is stated below for completeness.

Matrix completion algorithm:

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

$$\begin{aligned} \mathbf{K} &\triangleq \mathrm{pinv}\{\mathbf{X}\{\mathcal{C}_x \cap \mathcal{C}_y, \mathcal{C}_x \cap \mathcal{C}_y\}\} && \text{(3a)} \\ \mathbf{G}_x &\triangleq \mathbf{X}\{\mathcal{C}_x \setminus \mathcal{C}_y, \mathcal{C}_x \cap \mathcal{C}_y\} && \text{(3b)} \\ \mathbf{G}_y &\triangleq \mathbf{X}\{\mathcal{C}_y \setminus \mathcal{C}_x, \mathcal{C}_x \cap \mathcal{C}_y\} && \text{(3c)} \\ \mathbf{E}_x &\triangleq \mathbf{X}\{\mathcal{C}_x \setminus \mathcal{C}_y, \mathcal{C}_x \setminus \mathcal{C}_y\} \in \mathbb{C}^{d_x \times d_x} && \text{(3d)} \\ \mathbf{E}_y &\triangleq \mathbf{X}\{\mathcal{C}_y \setminus \mathcal{C}_x, \mathcal{C}_y \setminus \mathcal{C}_x\} \in \mathbb{C}^{d_y \times d_y} && \text{(3e)} \\ \mathbf{S}_x &\triangleq \mathbf{E}_x - \mathbf{G}_x \mathbf{K} \mathbf{G}_x^* = \mathbf{Q}_x \mathbf{D}_x \mathbf{Q}_x^* && \text{(3f)} \\ \mathbf{S}_y &\triangleq \mathbf{E}_y - \mathbf{G}_y \mathbf{K} \mathbf{G}_y^* = \mathbf{Q}_y \mathbf{D}_y \mathbf{Q}_y^* && \text{(3g)} \end{aligned}$$

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:

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

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* $\mathbf{X}_{\text{ref}}$ *of the decomposed SDP problem* (2)*. The output of the matrix completion algorithm, denoted as* $\mathbf{X}_{\text{opt}}$*, is a solution of the original SDP problem* (1)*. Moreover, the rank of* $\mathbf{X}_{\text{opt}}$ *is smaller than or equal to:*

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

Proof. See [35], [36] for the proof. □

IV. Alternating Direction Method of Multipliers

Consider the optimization problem

$$\begin{aligned}&\underset{\substack{\mathbf{x}\in\mathbb{R}^{n_x}\\ \mathbf{y}\in\mathbb{R}^{n_y}}}{\text{minimize}} && f(\mathbf{x}) + g(\mathbf{y}) && \text{(4a)}\\ &\text{subject to} && \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{c}. && \text{(4b)}\end{aligned}$$

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 given matrices. Also $f : \mathbb{R}^{n_x} \to \mathbb{R}\cup\{+\infty\}$ and $g : \mathbb{R}^{n_y} \to \mathbb{R}\cup\{+\infty\}$ are given convex functions. Notice that the variables $\mathbf{x}$ and $\mathbf{y}$ are coupled through the linear constraint (4b) while the objective function is separable.

The augmented Lagrangian function for problem (4) is equal to

$$\begin{aligned}\mathcal{L}_\mu(\mathbf{x},\mathbf{y},\lambda) = {} & f(\mathbf{x}) + g(\mathbf{y}) + && \text{(5a)}\\ & + \lambda^{\mathrm{T}}(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}-\mathbf{c}) && \text{(5b)}\\ & + (\mu/2)\|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}-\mathbf{c}\|_2^2, && \text{(5c)}\end{aligned}$$

where $\lambda \in \mathbb{R}^{n_c}$ is the Lagrange multiplier associated with the constraint (4b), and $\mu \in \mathbb{R}$ is a fixed parameter. ADMM is one approach for solving problem (4), which performs the following procedure at each iteration [37]:

$$\begin{aligned}\mathbf{x}^{k+1} &= \underset{\mathbf{x}\in\mathbb{R}^{n_x}}{\arg\min}\quad \mathcal{L}_\mu(\mathbf{x},\mathbf{y}^k,\lambda^k), && \text{(6a)}\\ \mathbf{y}^{k+1} &= \underset{\mathbf{y}\in\mathbb{R}^{n_y}}{\arg\min}\quad \mathcal{L}_\mu(\mathbf{x}^{k+1},\mathbf{y},\lambda^k), && \text{(6b)}\\ \lambda^{k+1} &= \lambda^k + \mu(\mathbf{A}\mathbf{x}^{k+1}+\mathbf{B}\mathbf{y}^{k+1}-\mathbf{c}). && \text{(6c)}\end{aligned}$$

where $k = 0,1,2,...,$ 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 (6a) and (6b) is an optimization sub-problem with respect to either $\mathbf{x}$ and $\mathbf{y}$, by freezing the other variable at its latest value. We employ the energy sequence $\{\varepsilon^k\}_{k=1}^\infty$ proposed in [4] as measure for convergence:

$$\varepsilon^{k+1} = (1/\mu)\|\lambda^{k+1}-\lambda^k\|_2^2 + \mu\|\mathbf{B}(y^{k+1}-y^k)\|_2^2 \qquad (7)$$

ADMM is particularly interesting for the cases where (6a) and (6b) can be performed 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 (2) in the form of (4) 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 Into 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*

$$\begin{aligned}&\underset{\mathbf{Z}\in\mathbb{H}^m}{\text{minimize}} && \|\mathbf{Z}-\widehat{\mathbf{Z}}\|_F^2 && \text{(8a)}\\ &\text{subject to} && \mathbf{Z}\succeq 0 && \text{(8b)}\end{aligned}$$

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 (8) can be solved through an eigenvalue decomposition of $\widehat{\mathbf{Z}}$.

Lemma 1. *Let*

$$\widehat{\mathbf{Z}} = \mathbf{Q}\times\operatorname{diag}\{(\nu_1\dots,\nu_m)\}\times\mathbf{Q}^*$$

denote the eigenvalue decomposition of $\widehat{\mathbf{Z}}$*. The solution of the projection problem* (8) *is given by*

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

Proof. See [38] for the proof. □

B. ADMM for Decomposed SDP

We apply ADMM to the following reformulation of the decomposed SDP problem (2):

$$\begin{aligned}&\underset{\substack{\mathbf{X}\in\mathcal{S}(\mathbf{C})\\ \{\mathbf{X}_{N;s}\in\mathcal{S}(\mathbf{N}_s)\}_{s=0}^p\\ \{\mathbf{X}_{C;r}\in\mathcal{S}(\mathbf{C}_r)\}_{r=1}^q\\ \{z_s\in\mathbb{R}\}_{s=0}^p}}{\text{minimize}} && z_0 + \sum_{s=1}^p \mathcal{I}_{l_s,u_s}(z_s) + \sum_{r=1}^q \mathcal{J}_r(\mathbf{X}_{C;r}) && \\ &\text{subject to} && \mathbf{X}\circ\mathbf{C}_r = \mathbf{X}_{C;r}, \quad r=1,2,\dots,q, && \text{(9a)}\\ & && \mathbf{X}\circ\mathbf{N}_s = \mathbf{X}_{N;s}, \quad s=0,1,\dots,p, && \text{(9b)}\\ & && z_s = \langle\mathbf{M}_s,\mathbf{X}_{N;s}\rangle, \quad s=0,1,\dots,p. && \text{(9c)}\end{aligned}$$

If $\mathbf{X}$ is a feasible solution of (9) with a finite objective value, then

$$\mathcal{J}_r(\mathbf{X}) = \mathcal{J}_r(\mathbf{X}\circ\mathbf{C}_r) \overset{\text{(9a)}}{=} \mathcal{J}_r(\mathbf{X}_{C;r}) = 0$$

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

$$\begin{aligned}\mathcal{I}_{l_s,u_s}(\langle \mathbf{X},\mathbf{M}_s\rangle) &= \mathcal{I}_{l_s,u_s}(\langle \mathbf{X}\circ\mathbf{N}_s,\mathbf{M}_s\rangle)\\ &\overset{(9b)}{=} \mathcal{I}_{l_s,u_s}(\langle \mathbf{X}_{N;s},\mathbf{M}_s\rangle)\\ &\overset{(9c)}{=} \mathcal{I}_{l_s,u_s}(z_s) = 0\end{aligned}$$

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 (2) 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 (9a) 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 (9b) for $s=0,1,\ldots,p$,
3) $\lambda_{z;s} \in \mathbb{R}$ as the Lagrange multiplier associated with the constraint (9c) for $s=0,1,\ldots,p$.

We regroup the primal and dual variables as

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

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

$$(2/\mu)\mathcal{L}_\mu(\mathcal{P}_1,\mathcal{P}_2,\mathcal{D}) = \mathcal{L}_D(\mathcal{D})/\mu^2 \tag{11a}$$
$$+\ \|z_0 - \langle \mathbf{M}_0,\mathbf{X}_{N;0}\rangle + (1+\lambda_{z;0})/\mu\|_F^2 \tag{11b}$$
$$+\sum_{s=1}^{p}\|z_s - \langle \mathbf{M}_s,\mathbf{X}_{N;s}\rangle + \lambda_{z;s}/\mu\|_F^2 + \mathcal{I}_{l_s,u_s}(z_s) \tag{11c}$$
$$+\sum_{r=1}^{q}\|\mathbf{X}\circ\mathbf{C}_r - \mathbf{X}_{C;r} + (1/\mu)\mathbf{\Lambda}_{C;r}\|_F^2 + \mathcal{J}_r(\mathbf{X}_{C;k}) \tag{11d}$$
$$+\sum_{s=1}^{p}\|\mathbf{X}\circ\mathbf{N}_s - \mathbf{X}_{N;s} + (1/\mu)\mathbf{\Lambda}_{N;s}\|_F^2 \tag{11e}$$

where

$$\begin{aligned}\mathcal{L}_D(\mathcal{D}) =& -(1+\lambda_{z;0})^2\\ &-\sum_{s=1}^{p}\lambda_{z;s}^2 - \sum_{r=1}^{q}\|\mathbf{\Lambda}_{C;r}\|_F^2 - \sum_{s=1}^{p}\|\mathbf{\Lambda}_{N;s}\|_F^2\end{aligned} \tag{12}$$

Using the blocks $\mathcal{P}_1$ and $\mathcal{P}_2$, the ADMM iterations for problem (9) can be expressed as follows:

1) The subproblem (6a) 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 (6b) 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 (8). 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 (6c), consists of three parallel steps:
   (a) *Updating* $\{\mathbf{\Lambda}_{C;r}\}_{r=1}^{q}$: Computational costs for this step involves no multiplications and is negligible.
   (b) *Updating* $\{\mathbf{\Lambda}_{N;s}\}_{s=0}^{p}$: Computational costs for this step involves no multiplications and is 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 $s=0,1,\ldots,p$.

The fact that every step of the above algorithm has an explicit easy-to-compute formula makes the algorithm very appealing for large-scale 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.,*

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

Theorem 2. *Assume that Slater's conditions hold for the decomposable SDP problem* (2) *and consider the iterative algorithm given in* (19). *The limit of* $\mathbf{X}^k$ *at* $k=+\infty$ *is an optimal solution for* (2).

Proof. The convergence of both primal and dual variables is guaranteed for a standard ADMM problem if the matrix $\mathbf{B}$ in (4b) has full column rank [39]. After realizing that (19) 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 (19) 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. □

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

Block 1: The first step of the algorithm that corresponds to (6a) consists of the operation

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

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

$$\mathrm{Re}\{X_{ij}\} \quad \text{for} \quad i=1,\ldots,n; \quad j=i,\ldots,n \tag{14a}$$
$$\mathrm{Im}\{X_{ij}\} \quad \text{for} \quad i=1,\ldots,n; \quad j=i+1,\ldots,n \tag{14b}$$
$$z_s \quad \text{for} \quad s=1,\ldots,p \tag{14c}$$

which leads to the explicit formulas (19a), (19b) and (19c).

Block 2: The second step of the algorithm that corresponds to (6b) consists of the operation

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

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

$$\mathbf{X}_{C;r} \quad \text{for} \quad r = 1, 2, \ldots, q \tag{16a}$$

$$\mathbf{X}_{N;s} \quad \text{for} \quad s = 0, 1, \ldots, p. \tag{16b}$$

Hence, the update of $\mathbf{X}_{C;r}$ reduces to the problem (8) 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

$$\begin{aligned}\mathcal{L}_{N;s}(\mathbf{Z}) = &\| z_s - \langle \mathbf{M}_s, \mathbf{Z}\rangle + \lambda_{z;s}/\mu\|_F^2 + \\ &\|\mathbf{X} \circ \mathbf{N}_s - \mathbf{Z} + (1/\mu)\mathbf{\Lambda}_{N;s}\|_F^2\end{aligned} \tag{17}$$

By taking the derivatives of this function, it is possible to find an explicit formula for $\mathbf{Z}_{\text{opt}}$. 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:

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

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,

$$\mathbf{Z}_{\text{opt}} = \mathbf{X} \circ \mathbf{N}_s + (1/\mu)\mathbf{\Lambda}_{N,s} + y_s \mathbf{M}_s, \tag{18}$$

where $y_s \triangleq z_s - \langle \mathbf{M}_s, \mathbf{Z}^{\text{opt}}\rangle + \lambda_{z;s}/\mu$. Hence, it only remains to derive the scalar y_s, which can be done by inner multiplying $\mathbf{M}_s$ to the both sides of the equation (18). This leads to the equations (19e) and (19f).

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 $\mathcal{E}_\mathcal{G}$ represents a transmission line. Let $\mathbf{Y} \in \mathbb{C}^{n\times n}$ denote the admittance matrix of the network. Define $\mathbf{V} \in \mathbb{C}^n$ as the 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 nodal complex power vector, where $\mathbf{P} \in \mathbb{R}^n$ and $\mathbf{Q} \in \mathbb{R}^n$ are the vectors of active and reactive powers injected at all buses. $\mathbf{P} + \mathbf{Q}\ \mathbf{i}$ can be interpreted as the complex-power supply minus the complex-power demand at node k of the network. The classical OPF problem can be described as follows:

$$\underset{\substack{\mathbf{V}\in\mathbb{C}^n\\ \mathbf{Q}\in\mathbb{R}^n\\ \mathbf{P}\in\mathbb{R}^n}}{\text{minimize}} \quad \sum_{k\,\in\,\mathcal{V}_\mathcal{G}} f_k(P_k) \tag{20a}$$

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

$$Q_k^{\min} \leq Q_k \leq Q_k^{\max}, \qquad k \in \mathcal{N} \tag{20c}$$

$$P_k^{\min} \leq P_k \leq P_k^{\max} \qquad k \in \mathcal{N} \tag{20d}$$

$$\mathbf{P} + \mathbf{i}\mathbf{Q} = \text{diag}\{\mathbf{V}\mathbf{V}^*\mathbf{Y}^*\} \tag{20e}$$

where $V_k^{\min}$, $V_k^{\max}$, $P_k^{\min}$, $P_k^{\max}$, $Q_k^{\min}$ and $Q_k^{\max}$ are given network limitations, and $f_k(P_k)$ is a convex function accounting for the power generation cost at node k. This problem may include additional constraints (such as thermal limits over the lines) that are ignored here only for the sake of simplicity in the presentation. For the same reason, assume that the objective function is the total active power loss $\sum_{k\,\in\,\mathcal{V}_\mathcal{G}} P_k$. More details on a general formulation may be found in [15].

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 $V_i V_j^*$ by 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) $\text{rank}\{\mathbf{W}\} = 1$. If we drop the rank condition as the only non-convex constraint of the reformulated OPF problem, we attain the SDP relaxation of OPF that is convex:

$$\underset{\mathbf{W}\in\mathbb{H}^n}{\text{minimize}} \quad \langle \mathbf{W}, (\mathbf{Y} + \mathbf{Y}^*)/2\rangle \tag{21a}$$

$$\text{subject to} \quad (V_k^{\min})^2 \leq \langle \mathbf{W}, e_k e_k^*\rangle \leq (V_k^{\max})^2, \quad k \in \mathcal{V}_\mathcal{H} \tag{21b}$$

$$Q_k^{\min} \leq \langle \mathbf{W}, \mathbf{Y}_{Q;k}\rangle \leq Q_k^{\max}, \qquad k \in \mathcal{V}_\mathcal{H} \tag{21c}$$

$$P_k^{\min} \leq \langle \mathbf{W}, \mathbf{Y}_{P;k}\rangle \leq P_k^{\max}, \qquad k \in \mathcal{V}_\mathcal{H} \tag{21d}$$

$$\mathbf{W} \succeq 0 \tag{21e}$$

where $e_1, \ldots, e_n$ denote the standard basis vectors in $\mathbb{R}^n$ and

$$\mathbf{Y}_{Q;k} \triangleq \frac{1}{2\mathbf{i}}(\mathbf{Y}_k^* e_k e_k^* - e_k e_k^* \mathbf{Y})$$

$$\mathbf{Y}_{P;k} \triangleq \frac{1}{2}(\mathbf{Y}^* e_k e_k^* + e_k e_k^* \mathbf{Y})$$

for every $k \in \mathcal{V}_\mathcal{H}$.

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. The major drawback of relaxing the OPF problem to an 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 [36] 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 the Polish Grid amounts to simple operations over matrices of size 27 by 27 or smaller.

VI. Simulation Results

In this section, we evaluate the performance of the proposed algorithm for solving the SDP relaxation of OPF over IEEE test cases. All simulations are run in MATLAB using a laptop with an Intel Core i7 quad-core 2.5 GHz CPU and 12 GB RAM. As shown in Figure 1, the energy function ε^k (as defined in (7)) is monotonically decreasing for all

ADMM for Decomposed SDP:

Block 1 :

$$\mathbf{X}^{k+1} := \left[\sum_{r=1}^{q} \mathbf{C}_r \circ (\mathbf{X}_{C;r}^k - \mathbf{\Lambda}_{C;r}^k/\mu) + \sum_{s=1}^{p} \mathbf{N}_s \circ (\mathbf{X}_{N;s}^k - \mathbf{\Lambda}_{N;s}^k/\mu)\right] \oslash_{\mathbf{C}} \left[\sum_{r=1}^{q} \mathbf{C}_r + \sum_{s=1}^{p} \mathbf{N}_s\right] \tag{19a}$$

$$z_0^{k+1} := \langle \mathbf{M}_0, \mathbf{X}_{N;0}^k \rangle - (\lambda_{z;0}^k + 1)/\mu \tag{19b}$$

$$z_s^{k+1} := \max\{\min\{\langle \mathbf{M}_s, \mathbf{X}_{N;s}^k \rangle - \lambda_{z;s}^k/\mu, u_s\}, l_s\} \qquad \text{for} \quad s = 1, 2, \ldots, p \tag{19c}$$

Block 2 :

$$\mathbf{X}_{C;r}^{k+1} := (\mathbf{X}^{k+1} \circ \mathbf{C}_r + \mathbf{\Lambda}_{C;r}^k/\mu)^+ \qquad \text{for} \quad r = 1, 2, \ldots, q \tag{19d}$$

$$y_s^{k+1} := \frac{z_s^{k+1} + \lambda_{z;s}^k/\mu - \langle \mathbf{M}_s, \mathbf{N}_s \circ \mathbf{X}^{k+1} + \mathbf{\Lambda}_{N;s}^k/\mu \rangle}{1 + \|\mathbf{M}_s\|_F^2} \qquad \text{for} \quad s = 0, 1, \ldots, p \tag{19e}$$

$$\mathbf{X}_{N;s}^{k+1} := \mathbf{N}_s \circ \mathbf{X}^{k+1} + \mathbf{\Lambda}_{N,s}^k/\mu + y_s^{k+1}\mathbf{M}_s \qquad \text{for} \quad s = 0, 1, \ldots, p \tag{19f}$$

Dual :

$$\mathbf{\Lambda}_{C;r}^{k+1} := \mathbf{\Lambda}_{C;r}^k + \mu(\mathbf{X}^{k+1} \circ \mathbf{C}_r - \mathbf{X}_{C;r}^{k+1}) \qquad \text{for} \quad r = 1, 2, \ldots, q \tag{19g}$$

$$\mathbf{\Lambda}_{N;s}^{k+1} := \mathbf{\Lambda}_{N;s}^k + \mu(\mathbf{X}^{k+1} \circ \mathbf{N}_s - \mathbf{X}_{N;s}^{k+1}) \qquad \text{for} \quad s = 0, 1, \ldots, p \tag{19h}$$

$$\lambda_{z;s}^{k+1} := \lambda_{z;s}^k + \mu(z_s^{k+1} - \langle \mathbf{M}_s, \mathbf{X}_{N;s}^{k+1} \rangle) \qquad \text{for} \quad s = 0, 1, \ldots, p \tag{19i}$$

Test cases	p	q	Maximum size of bags	Running time of 1000 iterations (sec)
Chow's 9 bus	27	7	3	6.18
IEEE 14 bus	42	12	3	9.96
IEEE 30 bus	90	18	4	14.66
IEEE 57 bus	171	26	6	21.25
IEEE 118 bus	354	66	5	53.13
IEEE 300 bus	900	111	7	98.95

TABLE I: Running time of the proposed algorithm for solving the SDP relaxation of OPF problem on IEEE test cases.

simulated cases. In addition, the utmost accuracy of 10^{-25} is ultimately achievable for all these systems. The time per 1000 iteration is between 6.18 and 100 seconds in a MATLAB implementation, which can be reduced significantly in C++ and parallel computing. We have verified that these numbers diminish by at least a factor of 3 if certain small-sized bags are combined to obtain a modest number of bags. This shows a trade-off between the chosen granularity for the algorithm and its computation time for a serial implementation (as opposed to a parallel implementation). To elaborate on the algorithm, note that every iteration amounts to a basic matrix operation or an eigendecomposition over matrices of size at most 7×7 for the IEEE 300-bus system. Efficient preconditioning methods could dramatically reduce the number of iterations (as OPF is often very ill-conditioned due to high inductance-to-resistance ratios), and this is left for future work.

VII. Conclusions

Motivated by the application of sparse semidefinite programming (SDP) to power networks, the objective of this work is to design a fast and parallelizable algorithm for solving sparse 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 IEEE benchmark systems.

References

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

[2] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Comput. Math. Appl.*, vol. 2, no. 1, pp. 17 – 40, 1976.

[3] i. 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-Math. Model. Num.*, vol. 9, no. R2, pp. 41–76, 1975.

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

[5] 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.

[6] P. Giselsson and S. Boyd, "Diagonal scaling in douglas-rachford splitting and admm," in *53rd Annual Conference on Decision and Control (CDC),*, Dec 2014, pp. 5033–5039.

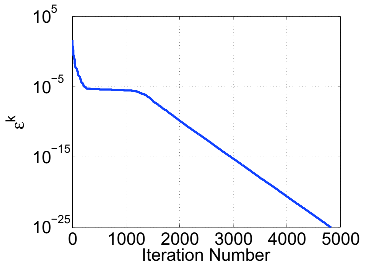
[7] B. He and X. Yuan, "On the $O(1/n)$ convergence rate of the Douglas–Rachford alternating direction method," *SIAM J. Numer. Anal.*, vol. 50, no. 2, pp. 700–709, 2012.

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

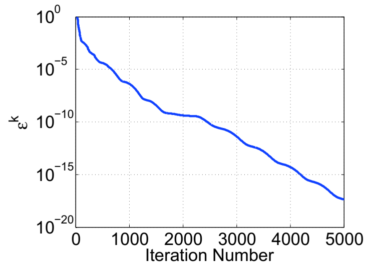
[9] Z. Wen, D. Goldfarb, and W. Yin, "Alternating direction augmented lagrangian methods for semidefinite programming," *Math. Program.*, vol. 2, no. 3-4, pp. 203–230, 2010.

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

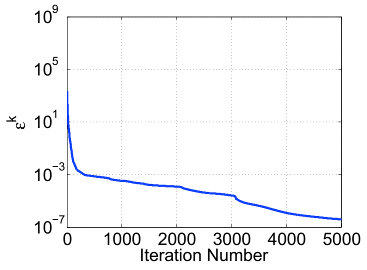
[11] Y. Sun, M. S. Andersen, and L. Vandenberghe, "Decomposition in conic optimization with partially separable structure," *SIAM J. Optimiz.*, vol. 24, no. 2, pp. 873–897, 2014.



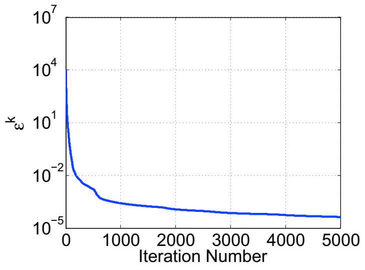
(a)



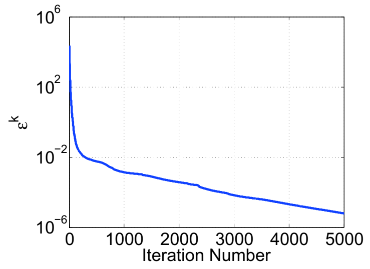
(b)



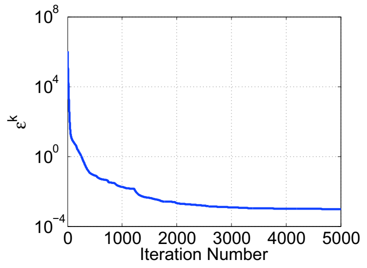
(c)



(d)



(e)



(f)

Fig. 1: These plots show the convergence behavior of the energy function ε^k for IEEE test cases. (a): Chow's 9 bus, (b): IEEE 14 bus, (c): IEEE 30 bus, (d): IEEE 57 bus, (e): IEEE 118 bus, (f): IEEE 300 bus.

[12] 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 Trans. Power Syst.,*, vol. 14, no. 1, pp. 96–104, Feb 1999.

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

[14] M. B. Cain, R. P. O'Neill, and A. Castillo, "History of optimal power flow and formulations," Federal Energy Regulatory Commission FERC, Tech. Rep., December 2012.

[15] J. Lavaei and S. Low, "Zero duality gap in optimal power flow problem," *IEEE Trans. Power Syst.*, vol. 27, no. 1, pp. 92–107, Feb. 2012.

[16] R. Madani, M. Ashraphijuo, and J. Lavaei, "Promises of conic relaxation for contingency-constrained optimal power flow problem," in *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton),*, Sept 2014, pp. 1064–1071.

[17] 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.

[18] S. H. Low, "Convex relaxation of optimal power flow (part I, II)," *IEEE Trans. Control of Network Systems*, vol. 1, no. Part I: 1; Part II: 2, pp. Part–I, 2014.

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

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

[21] L. Gan, N. Li, U. Topcu, and S. H. Low, "Optimal power flow in distribution networks," *Proc. 52nd IEEE Conference on Decision and Control*, 2013.

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

[23] B. Kim and R. Baldick, "Coarse-grained distributed optimal power flow," *IEEE Trans. Power Syst.,*, vol. 12, no. 2, pp. 932–939, May 1997.

[24] R. Baldick, B. Kim, C. Chase, and Y. Luo, "A fast distributed implementation of optimal power flow," *IEEE T. Power Syst.,*, vol. 14, no. 3, pp. 858–864, Aug 1999.

[25] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, "Dynamic network energy management via proximal message passing," *Foundations and Trends in Optimization*, vol. 1, no. 2, pp. 73–126, 2014.

[26] A. Lam, B. Zhang, and D. Tse, "Distributed algorithms for optimal power flow problem," in *51st Annual Conference on Decision and Control (CDC)*, Dec 2012, pp. 430–437.

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

[28] Q. Peng and S. Low, "Distributed algorithm for optimal power flow on a radial network," in *53rd Annual Conference on Decision and Control (CDC)*, Dec 2014, pp. 167–172.

[29] H. L. Bodlaender and A. M. Koster, "Treewidth computations I. upper bounds," *Inform. Comput.*, vol. 208, no. 3, pp. 259–275, 2010.

[30] H. L. Bodlaender and A. M. Koster, "Treewidth computations II. lower bounds," *Inform. Comput.*, vol. 209, no. 7, pp. 1103–1119, 2011.

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

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

[33] M. Laurent, "Polynomial instances of the positive semidefinite and Euclidean distance matrix completion problems," *SIAM J. Matrix Aanl. A.*, vol. 22, no. 3, pp. 874–894, 2001.

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

[35] R. Madani, G. Fazelnia, S. Sojoudi, and J. Lavaei, "Low-rank solutions of matrix inequalities with applications to polynomial optimization and matrix completion problems," in *53rd Annual Conference on Decision and Control (CDC)*, Dec 2014, pp. 4328–4335.

[36] R. Madani, M. Ashraphijuo, and J. Lavaei, "Promises of conic relaxation for contingency-constrained optimal power flow problem," to appear in *IEEE T. Power Syst.*, 2015, http://www.ee.columbia.edu/~lavaei/SCOPF_2014.pdf.

[37] 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.

[38] N. J. Higham, "Computing a nearest symmetric positive semidefinite matrix," *Linear Algebra Appl.*, vol. 103, pp. 103–118, May. 1988.

[39] B. He and X. Yuan, "On non-ergodic convergence rate of douglasrachford alternating direction method of multipliers," *Numer. Math.*, pp. 1–11, 2014. [Online]. Available: http://dx.doi.org/10.1007/s00211-014-0673-6