A Distributed Algorithm for Nonconvex Quadratically Constrained Programs ⋆

M.E. Ashour and C.M. Lagoa *

* *Department of Electrical Engineering and Computer Science, The Pennsylvania State University.*

Abstract: This paper considers nonconvex quadratically constrained programs over undirected connected graphs. We focus on problems whose quadratic constraint forms have sparse hessians, i.e., each constraint only involves a small subset of variables. We present both centralized and distributed treatment of the problem, where the centralized method is used as a benchmark with which we compare the performance of the proposed distributed algorithm. Towards this objective, we derive a lower bound on the optimal value of the problem using a semidefinite relaxation. Then, we develop a decentralized algorithm based on proximal gradient ADMM that exploits the structure of the constraints to distribute the computations among the graph nodes. Indeed, the proposed algorithm does away with a central node. Each node updates its local variables via solving a simple subproblem, then communicate with its immediate neighbors. An application of this work is the AC optimal power flow problem in power networks for which we provide preliminary numerical results to validate our findings.

Keywords: Distributed optimization, convex relaxation.

1. INTRODUCTION

In this paper, we consider optimization problems where the objective function is convex but the constraints are not. More precisely, we allow for the existence of quadratic equality constraints. The problem setup considered encompasses the so-called AC optimal power flow problem, where, apart from a set of convex constraints, quadratic equality constraints arise when modeling the power balance at each bus. Such problems are known to be NP-hard; see for example Bienstock and Verma (2015); Cain et al. (2012); Lehmann et al. (2016) and references therein. Hence, several numerical approaches propose approximations based e.g. on Newton methods Dommel and Tinney (1968), interior point based methods Yan and Quintana (1999) or global optimization heuristics Abido (2002); Lai et al. (1997). More recently, approximations based on semidefinite programming have been proposed; see for instance the recent two-part tutorial Low (2014a,b) and references therein.

However, all of the approaches mentioned above have a common limitation: lack of scalability. Their complexity rises quickly with the size of the network with quite a few of these becoming unusable for medium to large size problems. Guo et al. (2017) use ADMM with variable augmented Lagrangian penalty parameter to tackle the nonconvex optimization problem required for power systems in a distributed fashion. However, the technique proposed requires solving nonconvex ADMM subproblems which the reference does not elaborate on how to find local optima for these subproblems. An interesting study appears in Sun et al. (2013), which introduces local copies of the coupled variables at nodes, and demand that these local copies match the true ones, i.e., a similar idea to a consensus formulation. Their approach relies also on solving subproblems with nonconvex quadratic constraints, and do not explicitly mention ways of tackling these subproblems apart from using numerical solvers. Furthermore, there seems to be no hard evidence for a general convergence theory for ADMM on nonconvex problems, although this a hot research area now, see for instance Sun (2019). Kraning et al. (2014) present a proximal message passing algorithm for network energy management and highlight that convergence is not guaranteed with nonconvex subproblems.

To address the previously mentioned concerns, we follow a relax then distribute approach to the problem. In particular, we present a semidefinite relaxation of the problem inspired by d'Aspremont and Boyd (2003). Then, we propose a consensus based decentralized algorithm that exploits sparsity of the constraints to distribute the computations among the nodes on the network inspired by the proximal gradient ADMM (PG-ADMM) algorithm proposed by Aybat et al. (2018). The main contributions of this work are:

- We present a convex relaxation to the problem for which there exist concrete solution algorithms, rather than using a numerical solver.
- We use a distributed algorithm in which each node solves a convex subproblem and communicates with its neighbors. We show that the computational complexity per node scales with the node's degree, not with the network size. Hence, the solution is scalable.
- The proposed distributed algorithm is theoretically guaranteed to converge to a solution of the relaxed convex problem Aybat et al. (2018).

⋆ This work was partially supported by National Institutes of Health (NIH) Grant R01 HL142732, and National Science Foundation (NSF) Grant 1808266.

1.1 Notation

We represent scalars by lowercase light letters, e.g., x, vectors by boldface lowercase letters, e.g., $\mathbf{x}$, matrices with boldface uppercase letters, e.g., $\mathbf{X}$, and sets with caligraphic letters, e.g., $\mathcal{X}$. Vectors are viewed as column vectors and the transpose of $\mathbf{x}$ is $\mathbf{x}^\top$. The i-th entry of $\mathbf{x}$ is x_i, and the (i,j)-th entry of the matrix $\mathbf{X}$ is X_{ij}. The trace of $\mathbf{X}$ is $\mathrm{tr}(\mathbf{X})$. The cardinality of $\mathcal{X}$ is $|\mathcal{X}|$, and a vector whose elements are the diagonal entries of $\mathbf{X}$ is denoted by $\mathrm{diag}(\mathbf{X})$. Let $\Pi_{\mathcal{X}}(\mathbf{y}) = \mathrm{argmin}\{\|\mathbf{x}-\mathbf{y}\| : \mathbf{x} \in \mathcal{X}\}$ be the projection of $\mathbf{y}$ onto $\mathcal{X}$. The indicator function to the convex set $\mathcal{X}$ is denoted by $1_{\mathcal{X}}$, i.e., $1_{\mathcal{X}}(\mathbf{x}) = 0$ if $\mathbf{x} \in \mathcal{X}$, and is $+\infty$ otherwise. Vectors of all ones and all zeros are $\mathbf{1}$ and $\mathbf{0}$, respectively. Inequalities of vectors are with respect to the nonnegative orthant, and inequalities of matrices are with respect to the positive semidefinite cone. For $n \in \mathbb{Z}_+$, denote the index set $\{1,\dots,n\}$ by $[n]$.

2. PROBLEM FORMULATION

Let $\mathbb{G} = (\mathcal{N}, \mathcal{E})$ be a connected undirected graph with a set of nodes $\mathcal{N} = [N]$ for some $N \in \mathbb{Z}_+$, and a set of edges $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$. Assume without loss of generality that $(i,j) \in \mathcal{E}$ implies that $i < j$. Moreover, let $\mathcal{N}_i = \{j \in \mathcal{N} : (i,j) \in \mathcal{E} \text{ or } (j,i) \in \mathcal{E}\}$ be the set of neighbors of node i, and define $\mathcal{O}_i = \mathcal{N}_i \cup \{i\}$. Let $\mathbf{x} = [x_i]_{i\in\mathcal{N}}$ and consider the following nonconvex quadratically constrained program:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \triangleq \sum_{i\in\mathcal{N}} f_i(x_i) \\ \text{s.t.} \quad & \mathbf{x}^\top \mathbf{W}^{i\ell}\mathbf{x} + (\mathbf{w}^{i\ell})^\top \mathbf{x} + r^{i\ell} = 0, \quad i \in \mathcal{N},\ \ell \in [m_i] \\ & [x_j]_{j\in\mathcal{O}_i} \in \mathcal{X}_i, \quad i \in \mathcal{N}, \end{aligned} \tag{1}$$

where for each $i \in \mathcal{N}$, $f_i : \mathbb{R} \to \mathbb{R}$ is a differentiable convex function with a Lipschitz continuous gradient whose Lipschitz constant is L_i, and $\mathcal{X}_i$ is a convex set. Furthermore, $m_i \in \mathbb{Z}_+$, and for all $\ell \in [m_i]$, $r^{i\ell} \in \mathbb{R}$, $\mathbf{W}^{i\ell}_{kj} = 0$ for $(k,j) \notin \mathcal{O}_i \times \mathcal{O}_i$, and $w^{i\ell}_j = 0$ for $j \notin \mathcal{O}_i$. We impose no definiteness assumptions on the matrices $\mathbf{W}^{i\ell}$.

In the following lines, we state the motivation behind considering problem (1). Although f and $\cap_{i\in\mathcal{N}}\mathcal{X}_i$ are convex, the quadratic equality constraints render problem (1) nonconvex; hence, hard to solve. Furthermore, the complexity of finding a solution for (1) rapidly grows with N. Thus, we seek to develop a scalable decentralized algorithm that exploits the structure of (1) to approximate its solution. Indeed, problems of practical interest usually have lots of structure to be exploited, and some can be cast as in (1). A well known problem that fits the setup considered in this paper is the so-called AC optimal power flow problem. Let $\mathbb{G}$ represent a power network of $\mathcal{N}$ buses, and let $\mathbf{z} = [z_i]_{i\in\mathcal{N}}$, $\mathbf{y} = [y_i]_{i\in\mathcal{N}}$. Consider the following problem:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{y}} \quad & f(\mathbf{z}) \triangleq \sum_{i\in\mathcal{N}} f_i(z_i) \\ \text{s.t.} \quad & z_i = \sum_{j\in\mathcal{N}_i} y_i(y_i - y_j)w_{ij} + r_i, \quad i \in \mathcal{N} \\ & z_{\min_i} \le z_i \le z_{\max_i}, \quad i \in \mathcal{N} \\ & |y_i| \le y_{\max}, \quad i \in \mathcal{N}. \end{aligned} \tag{2}$$

Let z_i be the power generated at node $i \in \mathcal{N}$, y_i be the voltage of node i, w_{ij} be the admittance of the transmission line connecting nodes i and j, r_i be the power consumed by the load at node i, and $f(\mathbf{z})$ be the cost of power generation. Then, problem (2) is the optimal power flow problem that minimizes the cost of power generation subject to balancing the supply and demand. In particular, the quadratic constraint in (2) represents the power balance equations that need to be satisfied at each node. Problem (2) lends itself to the formulation in (1).

A centralized solution to problem (1) is not desirable. Indeed, a centralized solution requires the presence of a central node that possesses the knowledge of the global topology of the graph $\mathbb{G}$. Moreover, the complexity of a centralized solution rapidly increases with the size of the network; thus hinders its scalability. That said, we aim in this work to provide a solution that resolves the two aforementioned issues, namely, the nonconvexity of the problem and the pressing need for a decentralized scalable algorithm that exploits the structure of the quadratic constraint to distribute the computations among the graph nodes while maintaining a low overhead of communication in the network.

Remark 1. For a generic program with quadratic equality constraints, one could construct a graph of dependencies for coupled decision variables to cast the problem as in (1).

Indeed, if one picks random quadratic constraints, it might end up being the case that the generated graph is complete. However, practical problems as the one shown in (2) usually have structure that might lead to a graph with sparse connectivity. Nevertheless, the method described here works for complete graphs, but with a relatively high overhead generated by control signaling among nodes.

3. CONVEX RELAXATION

In this section, we establish the basis of the proposed distributed algorithm by relaxing problem (1) into a convex semidefinite program, d'Aspremont and Boyd (2003). Note that $\mathbf{x}^\top\mathbf{W}^{i\ell}\mathbf{x} = \mathrm{tr}(\mathbf{W}^{i\ell}\mathbf{x}\mathbf{x}^\top)$; thus, problem (1) is equivalent to:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{X}} \quad & \sum_{i\in\mathcal{N}} f_i(x_i) \\ \text{s.t.} \quad & \mathrm{tr}(\mathbf{W}^{i\ell}\mathbf{X}) + (\mathbf{w}^{i\ell})^\top\mathbf{x} + r^{i\ell} = 0,\ i \in \mathcal{N},\ \ell \in [m_i] \\ & \mathbf{X} = \mathbf{x}\mathbf{x}^\top \\ & [x_j]_{j\in\mathcal{O}_i} \in \mathcal{X}_i, \quad i \in \mathcal{N}. \end{aligned} \tag{3}$$

Introducing the matrix $\mathbf{X} \in \mathbb{R}^{N\times N}$ transforms the quadratic equality constraint into an affine one; hence, convex. However, $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$ is a nonconvex constraint. This constraint is replaced with the semidefinite constraint $\mathbf{X} \succeq \mathbf{x}\mathbf{x}^\top$. Using Schur complement d'Aspremont and Boyd (2003), one can formulate the convex relaxation of (3) as:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{X}} \quad & \sum_{i\in\mathcal{N}} f_i(x_i) + \theta\,\mathrm{tr}(\mathbf{X}) \\ \text{s.t.} \quad & \mathrm{tr}(\mathbf{W}^{i\ell}\mathbf{X}) + (\mathbf{w}^{i\ell})^\top\mathbf{x} + r^{i\ell} = 0,\ i \in \mathcal{N},\ \ell \in [m_i] \\ & [x_j]_{j\in\mathcal{O}_i} \in \mathcal{X}_i, \quad i \in \mathcal{N} \\ & \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix} \succeq 0, \end{aligned} \tag{4}$$

where $\theta \in \mathbb{R}_+$ is a regularization parameter. The regularization $\mathrm{tr}(\mathbf{X})$ induces a sparsification effect on the eigenvalues of $\mathbf{X}$; hence, favors low rank solutions. In fact, a rank 1 matrix $\mathbf{X}$ produced by solving (4) with $\theta = 0$ implies that (4) and (3) are equivalent; consequently, (4) and (1) are equivalent in this case. However, solving (4) provides a lower bound on the optimal value of (1).

4. DISTRIBUTED ALGORITHM

A centralized solution to problem (4) necessitates that each node $i \in \mathcal{N}$ reveal its private cost function f_i, and requires the knowledge of the global topology of the graph $\mathbb{G}$. Herein, we reformulate (1) to exploit the structure of the quadratic constraints. This opens room for the development of a distributed algorithm that makes use only of the local topology known at each node, i.e., each node only needs to know who its immediate neighbors are.

A first step towards our goal is to note that the equality constraints in (1) can be written in a simpler form. Let $\bar{\mathbf{x}}_i = [x_j]_{j\in\mathcal{O}_i}$, then problem (1) is equivalent to:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \sum_{i\in\mathcal{N}} f_i(x_i) \\ \text{s.t.} \quad & \bar{\mathbf{x}}_i^\top \bar{\mathbf{W}}^{i\ell} \bar{\mathbf{x}}_i + (\bar{\mathbf{w}}^{i\ell})^\top \bar{\mathbf{x}}_i + r^{i\ell} = 0, \quad i \in \mathcal{N},\ \ell \in [m_i] \\ & \bar{\mathbf{x}}_i \in \mathcal{X}_i, \quad i \in \mathcal{N}, \end{aligned} \tag{5}$$

with the matrices $\bar{\mathbf{W}}^{i\ell} \in \mathbb{R}^{|\mathcal{O}_i|\times|\mathcal{O}_i|}$ constructed appropriately as submatrices of $\mathbf{W}^{i\ell}$. Clearly, $\bar{\mathbf{W}}^{i\ell}$ is considered local data known at node i.

The formulation in (5) is amenable to decentralization. We now let $\mathbf{x}_i = [x_{ij}]_{j\in\mathcal{O}_i}$ be the variables kept locally at node i. For all $i \in \mathcal{N}$, x_{ij} represents the belief node i has about the decision of node $j \in \mathcal{N}_i$, and x_{ii} is node's i true local decision. Thus, problem (5) can be cast in consensus form, i.e., demanding that local beliefs at each node about the decision of its neighbors match their true ones. Evidently, problem (5) is equivalent to:

$$\begin{aligned} \min_{[\mathbf{x}_i]_{i\in\mathcal{N}}} \quad & \sum_{i\in\mathcal{N}} f_i(x_{ii}) \\ \text{s.t.} \quad & \mathbf{x}_i^\top \bar{\mathbf{W}}^{i\ell} \mathbf{x}_i + (\bar{\mathbf{w}}^{i\ell})^\top \mathbf{x}_i + r^{i\ell}, \quad i \in \mathcal{N},\ \ell \in [m_i] \\ & \mathbf{x}_i \in \mathcal{X}_i, \quad i \in \mathcal{N} \\ & x_{ii} = x_{ji}, \quad x_{ij} = x_{jj}, \quad (i,j) \in \mathcal{E}. \end{aligned} \tag{6}$$

The consensus constraints $x_{ii} = x_{ji}$ and $x_{ij} = x_{jj}$ for all $(i,j) \in \mathcal{E}$ imply that the copies of a node's decision are the same at all neighboring nodes. This establishes the equivalence of (5) and (6).

We make use of the semidefinite relaxation introduced in Section 3 to formulate a convex relaxation of (6). In particular, consider the following problem:

$$\begin{aligned} \min_{\mathbf{x}_i,\mathbf{X}_i} \quad & \sum_{i\in\mathcal{N}} f_i(x_{ii}) \\ \text{s.t.} \quad & \mathrm{tr}(\bar{\mathbf{W}}^{i\ell}\mathbf{X}_i) + (\bar{\mathbf{w}}^{i\ell})^\top \mathbf{x}_i + r^{i\ell}, \quad i \in \mathcal{N},\ \ell \in [m_i] \\ & \mathbf{x}_i \in \mathcal{X}_i, \quad i \in \mathcal{N} \\ & x_{ii} = x_{ji}, \quad x_{ij} = x_{jj} \quad (i,j) \in \mathcal{E} \\ & \begin{bmatrix} \mathbf{X}_i & \mathbf{x}_i \\ \mathbf{x}_i^\top & 1 \end{bmatrix} \succeq 0. \end{aligned} \tag{7}$$

Algorithm 1 $\{\mathbf{x}_i^0,\ \mathbf{X}_i^0,\ \gamma_i,\ c_i\}_{i\in\mathcal{N}}$

1: Set $\forall i \in \mathcal{N}$: $s_{ii}^0 = \sum_{j\in\mathcal{O}_i} \Gamma_{ij} x_{ji}^0$, $s_{ij}^0 = \Gamma_{ij}(x_{jj}^0 - x_{ij}^0)$, $j \in \mathcal{N}_i$, $\mathbf{s}_i^0 = [s_{ij}^0]_{j\in\mathcal{O}_i}$, $\mathbf{p}_i^0 = [p_{ij}^0]_{j\in\mathcal{O}_i} = \mathbf{0}$.
2: **for** $k = 0, 1, \ldots$ and each $i \in \mathcal{N}$ **do**
3: $\quad \bar{x}_{ii}^k \leftarrow x_{ii}^k - c_i[\nabla f_i(x_{ii}^k) + s_{ii}^k + p_{ii}^k]$
4: $\quad \bar{x}_{ij}^k \leftarrow x_{ij}^k - c_i[s_{ij}^k + p_{ij}^k]$, $j \in \mathcal{N}_i$
5: $\quad \bar{\mathbf{x}}_i^k \leftarrow [\bar{x}_{ij}^k]_{j\in\mathcal{O}_i}$
6: $\quad (\mathbf{x}_i^{k+1}, \mathbf{X}_i^{k+1}) \leftarrow \Pi_{\mathcal{I}_i}(\bar{\mathbf{x}}_i^k, \mathbf{X}_i^k)$
7: $\quad s_{ii}^{k+1} \leftarrow \sum_{j\in\mathcal{O}_i} \Gamma_{ij} x_{ji}^{k+1}$
8: $\quad s_{ij}^{k+1} \leftarrow \Gamma_{ij}(x_{jj}^{k+1} - x_{ij}^{k+1})$, $j \in \mathcal{N}_i$
9: $\quad \mathbf{p}_i^{k+1} \leftarrow \mathbf{p}_i^k + \mathbf{s}_i^{k+1}$

Aybat et al. (2018) propose a distributed PG-ADMM algorithm suitable for solving consensus optimization problems. We propose a distributed algorithm based on Aybat et al. (2018) customized to problem (7). To simplify notation, we group all the constraints in (7) that are local to node i in a set $\mathcal{I}_i$. In particular, let

$$\begin{aligned} \mathcal{I}_i = \Big\{ & (\mathbf{x}, \mathbf{X}) \in \mathbb{R}^{|\mathcal{O}_i|} \times \mathbb{R}^{|\mathcal{O}_i|\times|\mathcal{O}_i|} : \\ & \mathrm{tr}(\bar{\mathbf{W}}^{i\ell}\mathbf{X}) + (\bar{\mathbf{w}}^{i\ell})^\top \mathbf{x} + r^{i\ell} = 0,\ \ell \in [m_i],\ \mathbf{x} \in \mathcal{X}_i, \\ & \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix} \succeq 0 \Big\}. \end{aligned} \tag{8}$$

Moreover, define $\Gamma \in \mathbb{R}^{N\times N}$ such that for each $i \in \mathcal{N}$, $\Gamma_{ij} = 0$ for $j \notin \mathcal{N}_i$, $\Gamma_{ij} = -\frac{\gamma_i\gamma_j}{\gamma_i+\gamma_j}$ for $j \in \mathcal{N}_i$, and $\Gamma_{ii} = \sum_{j\in\mathcal{N}_i} \frac{\gamma_i\gamma_j}{\gamma_i+\gamma_j}$, with $\gamma_i \in \mathbb{R}_+$ for all $i \in \mathcal{N}$. Algorithm 1 summarizes the proposed distributed algorithm for approximating the solution of (1), where the superscript k denotes the iteration index.

We highlight the key steps of Algorithm 1 and defer its derivation to Appendix A. Iteration k of Algorithm 1 consists of the following steps:

- In step 6, each $i \in \mathcal{N}$ updates its local variables $(\mathbf{x}_i, \mathbf{X}_i)$ via solving the semidefinite program (SDP):
$$\begin{aligned} \min \quad & \|\mathbf{x}_i - \bar{\mathbf{x}}_i^k\|^2 + \|\mathbf{X}_i - \mathbf{X}_i^k\|_{\mathrm{F}}^2 + \theta_i \mathrm{tr}(\mathbf{X}_i) \\ \text{s.t.} \quad & (\mathbf{x}_i, \mathbf{X}_i) \in \mathcal{I}_i \end{aligned} \tag{9}$$
based solely on available local information, where $\|.\|_{\mathrm{F}}$ denotes the Frobenius norm. The dimension of the SDP depends only on $|\mathcal{N}_i|$ which is negligible compared to $|\mathcal{N}|$ in large scale networks. The penalty on the trace of $\mathbf{X}_i$ is used to induce low rank solutions, and $\theta_i \in \mathbb{R}_+$.
- In steps 7 and 8, each node exchanges information with its immediate neighbors. More precisely, each $i \in \mathcal{N}$ broadcasts x_{ii}^{k+1} to all $j \in \mathcal{N}_i$, sends x_{ij}^{k+1} to node $j \in \mathcal{N}_i$, and receives x_{ji}^{k+1} from node $j \in \mathcal{N}_i$.

We emphasize on the scalability of Algorithm 1. In fact, the computational load at each node originates from solving the local SDP whose complexity is almost independent of the size of the network. Instead, it depends on the degree of the node which is expected to grow extremely slowly as $|\mathcal{N}|$ increases. In contrast, the complexity of a centralized solution adversely grows as $|\mathcal{N}|$ increases.

5. NUMERICAL RESULTS

This section presents an example application of Algorithm 1 to approximate the solution of an instance of problem

(2). In particular, we consider an optimal power flow allocation problem on a power network. We randomly generate a graph $\mathbb{G}$ of $|\mathcal{N}| = 20$ nodes, where the graph is generated such that an edge exists between any pair of nodes with an initial probability 0.01, and that probability is gradually increased until a connected graph is obtained. The weight of any edge, w_{ij}, is drawn from a folded normal distribution, precisely, as the absolute value of a standard Gaussian random variable, independently and identically distributed across all edges $(i,j) \in \mathcal{E}$. Three generator nodes out of $\mathcal{N}$ are picked at random, denoted by $\mathcal{G} \subset \mathcal{N}$, while the rest of nodes generate no power.

The objective is to minimize the power generation cost subject to power balance constraints at each node represented by the quadratic equality constraints in (2). More precisely, let z_i denote the power generated at node $i \in \mathcal{N}$, which is obviously forced to be zero if $i \notin \mathcal{G}$, and the cost at node $i \in \mathcal{G}$ is $f_i(z_i) = z_i^2$. Furthermore, let y_i denote the voltage of node $i \in \mathcal{N}$, r_i denote the power consumed by the load connected to node i, and w_{ij} to be the admittance of the transmission line represented by the edge $(i,j) \in \mathcal{E}$. For each node $i \in \mathcal{N}$, r_i is independently drawn from a uniform distribution on the interval $[0,1]$. We set $z_{\min_i} = 0$, $z_{\max_i} = 5$ for all $i \in \mathcal{G}$, $z_{\min_i} = z_{\max_i} = 0$ for $i \notin \mathcal{G}$, and $y_{\max} = 5$. We then rewrite problem (2) to fit our formulation as follows:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{y}} \quad & \sum_{i\in\mathcal{N}} f_i(z_i) \\ \text{s.t.} \quad & z_i = \mathbf{y}^\top \mathbf{W}^i \mathbf{y} + r_i, \quad i \in \mathcal{N} \\ & z_{\min_i} \le z_i \le z_{\max_i}, \quad i \in \mathcal{N} \\ & |y_i| \le y_{\max}, \quad i \in \mathcal{N}. \end{aligned} \tag{10}$$

The symmetric matrix $\mathbf{W}^i$ is constructed such that $\mathbf{W}^i_{ii} = \sum_{j\in\mathcal{N}_i} w_{ij}$, $\mathbf{W}^i_{ij} = \frac{-w_{ij}}{2}$ for $j \in \mathcal{N}_i$, and zero elsewhere. The centralized convex relaxation of (10) is then given by:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{Y}} \quad & \sum_{i\in\mathcal{N}} f_i(z_i) \\ \text{s.t.} \quad & z_i = \operatorname{tr}(\mathbf{W}^i \mathbf{Y}) + r_i, \quad i \in \mathcal{N} \\ & z_{\min_i} \le z_i \le z_{\max_i}, \quad i \in \mathcal{N} \\ & \mathbf{0} \preceq \operatorname{diag}(\mathbf{Y}) \preceq y_{\max}^2 \mathbf{1} \\ & \mathbf{Y} \succeq 0, \end{aligned} \tag{11}$$

It is worth noting that $\mathbf{y}$ no longer exists in (11) to simplify the implementation. Nevertheless, an approximate retrieval of $\mathbf{y}^*$ can be done using an eigen decomposition of the matrix $\mathbf{Y}^*$ produced by solving (11). For instance, pick $\mathbf{y}^* = \sqrt{\lambda_{\max}}\mathbf{u}_{\lambda_{\max}}$, where $\lambda_{\max}$ is the maximum eigenvalue of $\mathbf{Y}^*$ and $\mathbf{u}_{\lambda_{\max}}$ is its corresponding eigenvector. Thus, $\mathbf{y}^*\mathbf{y}^{*\top}$ is the rank 1 matrix closest to $\mathbf{Y}^*$ in Frobenius norm by the matrix approximation lemma Eckart and Young (1936). We solve problem (11) in a centralized fashion with the purpose of using its solution as a benchmark with which we compare the performance of Algorithm 1. To solve problem (11) we use CVX, a package for specifying and solving convex programs Grant and Boyd (2014).

Next, we use the ideas presented in Section 4 to cast problem (11) in consensus form as follows:

$$\begin{aligned} \min_{\mathbf{z},\{\mathbf{d}_i,\mathbf{Y}_i\}_{i\in\mathcal{N}}} \quad & \sum_{i\in\mathcal{N}} f_i(z_i) \\ \text{s.t.} \quad & z_i = \operatorname{tr}(\bar{\mathbf{W}}^i \mathbf{Y}_i) + r_i, \quad i \in \mathcal{N} \\ & \mathbf{Y}_i \succeq 0, \quad i \in \mathcal{N} \\ & \mathbf{d}_i = \operatorname{diag}(\mathbf{Y}_i), \quad i \in \mathcal{N} \\ & z_{\min_i} \le z_i \le z_{\max_i} \\ & 0 \le d_{ij} \le y_{\max}^2, \quad i \in \mathcal{N},\ j \in \mathcal{N}_i \cup \{i\} \\ & d_{ii} = d_{ji}, \quad d_{ij} = d_{jj}, \quad (i,j) \in \mathcal{E}. \end{aligned} \tag{12}$$

Ideally, if one is able to find rank 1 matrices $\mathbf{Y}_i$ via solving (12), then their diagonal entries d_{ij}, for $j \in \mathcal{N}_i \cup \{i\}$, are precisely equivalent to the squared variables y_{ij} in problem (11). This motivates the choice of the consensus constraints to be on the diagonal entries of the matrices $\mathbf{Y}_i$ in (12), since they act as a proxy for y_{ij}^2.

After performing the appropriate mapping between problems (12) and (7), we use Algorithm 1 to find a solution for problem (12). We evaluate the performance of Algorithm 1 according to the following metrics.

- Violation of the consensus constraints in (6). Indeed, in each iteration of the algorithm, we reconstruct $\mathbf{y}_i^k = [y_{ij}^k]_{j\in\mathcal{O}_i}$ by finding the nearest rank 1 matrix to $\mathbf{Y}_i^k$ Eckart and Young (1936). We then use two measures of consensus violation, the first of which termed max-to-min distance, computed at iteration k as:
$$\max_{i\in\mathcal{N}} \left\{ \max_{j\in\mathcal{O}_i} y_{ji}^k - \min_{j\in\mathcal{O}_i} y_{ji}^k \right\}, \tag{13}$$
and the other termed distance to mean computed at iteration k as:
$$\frac{1}{|\mathcal{N}|} \sum_{i\in\mathcal{N}} \frac{1}{|\mathcal{O}_i|} \sum_{j\in\mathcal{O}_i} (y_{ji}^k - \frac{1}{|\mathcal{O}_i|} \sum_{\ell\in\mathcal{O}_i} y_{\ell i}^k). \tag{14}$$
- Infeasibility of the iterates generated by Algorithm 1 with respect to the quadratic equality constraints in the original nonconvex problem (10). In each iteration k, $\mathbf{y}_i^k = [y_{ij}^k]_{j\in\mathcal{O}_i}$ is reconstructed from $\mathbf{Y}_i^k$ for each $i \in \mathcal{N}$. Then, we let $\mathbf{y} = [y_{ii}^k]_{i\in\mathcal{N}}$, i.e., the vector of true decisions at all nodes, not their copies kept locally at their neighbors. Max infeasibility at iteration k is then measured as:
$$\max_{i\in\mathcal{N}} \quad |z_i^k - (\mathbf{y}^k)^\top \mathbf{W}^i \mathbf{y}^k - r_i|, \tag{15}$$
and average infeasibility at iteration k is computed as:
$$\frac{1}{|\mathcal{N}|} \sum_{i\in\mathcal{N}} |z_i^k - (\mathbf{y}^k)^\top \mathbf{W}^i \mathbf{y}^k - r_i|. \tag{16}$$
- Suboptimality of the iterates generated by Algorithm 1 relative to the centralized solution of problem (11). More precisely, let f^* be the optimal objective value of (11), then the relative suboptimality is computed at iteration k as:
$$\frac{\|\mathbf{z}^k\| - \sqrt{f^*}}{\sqrt{f^*}}. \tag{17}$$

Fig. 1, 2, and 3 show the performance of Algorithm 1 with respect to the three aforementioned metrics. We choose $\gamma_i = 0.3$ for all $i \in \mathcal{N}$, and set $c_i = (\mathrm{L}_i + \gamma_i|\mathcal{N}_i|)^{-1}$, where $\mathrm{L}_i = 1$ for all $i \in \mathcal{N}$ according to the chosen cost function f_i. We initialize z_i^0 and each entry of $\mathbf{d}_i^0$ as realizations

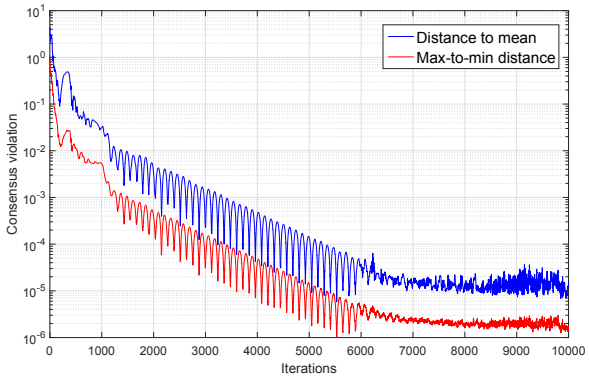

Fig. 1. Consensus violation of Algorithm 1's iterates.

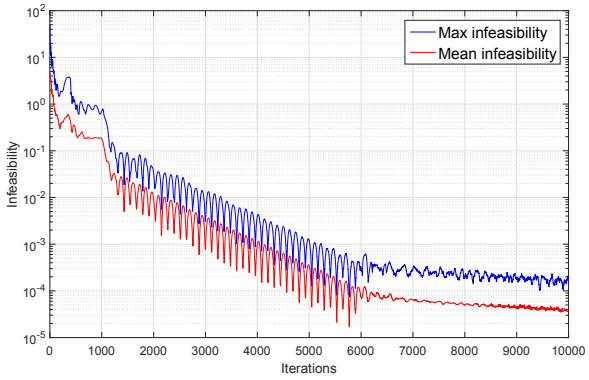

Fig. 2. Infeasibility of Algorithm 1's iterates.

of uniform distributions on the intervals $[z_{\min_i}, z_{\max_i}]$ and $[0, y^2_{\max}]$, respectively. Furthermore, we set $\mathbf{Y}_i^0$ to a rank 1 matrix with the entries of $\mathbf{d}_i^0$ being its diagonal elements.

Fig. 1 serves as an evidence that Algorithm 1 performs quite well in terms of consensus among neighboring nodes. In Fig. 2, we show that Algorithm 1 iterates approach the feasible set of (10) as time progresses. Finally, Fig. 3 shows that the iterates generated by Algorithm 1 lead to a cost that approaches its optimal value obtained by solving (11), which provides a lower bound on the optimal value of (10).

6. CONCLUSION

In this paper we propose an algorithm for solving optimization problems with quadratic equality constraints, a problem known to be NP-hard. By exploiting sparsity of the constraints, we are able to decentralize the computations and develop an approach that scales well with the size of the network. Effectiveness of the proposed algorithm is demonstrated using randomly generated problems. Effort is now being put in extending the proposed approach to problems where the constraints are uncertain, and in performing large-scale simulations.

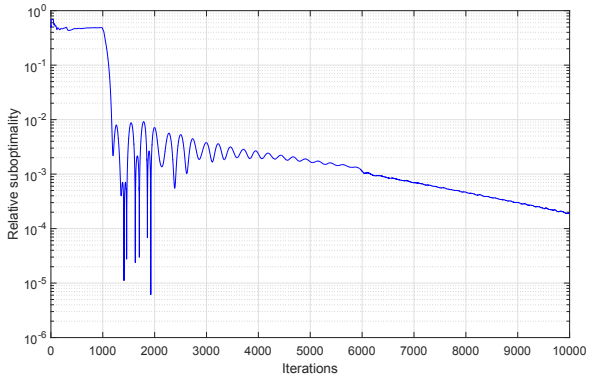

Fig. 3. Suboptimality of Algorithm 1's iterates.

REFERENCES

Abido, M. (2002). Optimal power flow using particle swarm optimization. *International Journal of Electrical Power & Energy Systems*, 24(7), 563–571.

Aybat, N.S., Wang, Z., Lin, T., and Ma, S. (2018). Distributed linearized alternating direction method of multipliers for composite convex consensus optimization. *IEEE Transactions on Automatic Control*, 63(1), 5–20.

Bienstock, D. and Verma, A. (2015). Strong NP-hardness of AC power flows feasibility. *arXiv preprint arXiv:1512.07315*.

Cain, M.B., O'Neill, R.P., and Castillo, A. (2012). History of optimal power flow and formulations (opf paper 1). Technical report, US Federal Energy Resource Commission (FERC).

d'Aspremont, A. and Boyd, S. (2003). Relaxations and randomized methods for nonconvex qcqps. *EE392o Class Notes, Stanford University*, 1, 1–16.

Dommel, H. and Tinney, W. (1968). Optimal power flow solutions. *IEEE transactions on power apparatus and systems*, 10, 1866–1876.

Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3), 211–218.

Grant, M. and Boyd, S. (2014). CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`.

Guo, J., Hug, G., and Tonguz, O.K. (2017). A case for nonconvex distributed optimization in large-scale power systems. *IEEE Transactions on Power Systems*, 32(5), 3842–3851.

Kraning, M., Chu, E., Lavaei, J., Boyd, S., et al. (2014). Dynamic network energy management via proximal message passing. *Foundations and Trends® in Optimization*, 1(2), 73–126.

Lai, L., Ma, J., Yokoyama, R., and Zhao, M. (1997). Improved genetic algorithms for optimal power flow under both normal and contingent operation states. *International Journal of Electrical Power & Energy Systems*, 19(5), 287–292.

Lehmann, K., Grastien, A., and Van Hentenryck, P. (2016). Ac-feasibility on tree networks is np-hard. *IEEE Transactions on Power Systems*, 31, 798–801.

Low, S.H. (2014a). Convex relaxation of optimal power flow; Part I: formulations and equivalence. *IEEE Transactions on Control of Network Systems*, 1(1), 15–27. doi: 10.1109/TCNS.2014.2309732.

Low, S.H. (2014b). Convex relaxation of optimal power flow; Part II: exactness. *IEEE Transactions on Control of Network Systems*, 1(2), 177–189. doi: 10.1109/TCNS.2014.2323634.

Sun, A.X., Phan, D.T., and Ghosh, S. (2013). Fully decentralized AC optimal power flow algorithms. In

2013 IEEE Power & Energy Society General Meeting, 1–5.

Sun, Kaizhao, X.A. (2019). A two-level distributed algorithm for general constrained non-convex optimization with global convergence. *arXiv preprint arXiv:1902.07654*.

Yan, X. and Quintana, V. (1999). Improving an interior-point-based opf by dynamic adjustments of step sizes and tolerances. *IEEE Transactions on Power Systems*, 14(2), 709–717.

Appendix A. DERIVATION OF ALGORITHM 1

Algorithm 1 uses a version of PG-ADMM proposed by Aybat et al. (2018) customized to problem (7). The derivation presented next follows similar ideas to those presented by Aybat et al. (2018). However, in our case each node does not share all of its local decision with its neighbors. Instead, it shares different subsets of its local decision depending on the identity of its neighbors. We outline the main steps used in the derivation.

Problem (7) can be cast as follows:

$$\begin{aligned} \min_{\{\mathbf{x}_i,\mathbf{X}_i\}_{i\in\mathcal{N}}} \quad & \sum_{i\in\mathcal{N}} [f_i(x_{ii}) + 1_{\mathcal{I}_i}(\mathbf{x}_i,\mathbf{X}_i)] \\ \text{s.t.} \quad & \mathbf{A}_{ij}\mathbf{x}_i = \mathbf{M}_{ij}\mathbf{x}_j, \quad (i,j)\in\mathcal{E}, \end{aligned} \tag{A.1}$$

where the matrices $\mathbf{A}_{ij}$ and $\mathbf{M}_{ij}$ are constructed appropriately to enforce the consensus constraints in (7). Introduce the primal variable $\mathbf{y} = [\mathbf{y}_{ij}]_{(i,j)\in\mathcal{E}}$ and consider:

$$\begin{aligned} \min_{\mathbf{y},\{\mathbf{x}_i,\mathbf{X}_i\}_{i\in\mathcal{N}}} \quad & \sum_{i\in\mathcal{N}} [f_i(x_{ii}) + 1_{\mathcal{I}_i}(\mathbf{x}_i,\mathbf{X}_i)] \\ \text{s.t.} \quad & \mathbf{A}_{ij}\mathbf{x}_i = \mathbf{y}_{ij}, \quad (i,j)\in\mathcal{E} \\ & \mathbf{M}_{ij}\mathbf{x}_j = \mathbf{y}_{ij}, \quad (i,j)\in\mathcal{E}, \end{aligned} \tag{A.2}$$

Let $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_{ij}]_{(i,j)\in\mathcal{E}}$ and $\boldsymbol{\beta} = [\boldsymbol{\beta}_{ij}]_{(i,j)\in\mathcal{E}}$ be the dual variables associated with the equality constraints in (A.2). Also let $\mathbf{x} = [\mathbf{x}_i]_{i\in\mathcal{N}}$, and $\mathbf{X} = [\mathbf{X}_i]_{i\in\mathcal{N}}$. The augmented Lagrangian function is then given by:

$$L(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_{i\in\mathcal{N}} 1_{\mathcal{I}_i}(\mathbf{x}_i,\mathbf{X}_i) + \phi(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta}), \tag{A.3}$$

where the differentiable part of L, ϕ, is given by:

$$\begin{aligned} \phi(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta}) &= \sum_{i\in\mathcal{N}} f_i(x_{ii}) \\ &+ \sum_{(i,j)\in\mathcal{E}} \Big[\boldsymbol{\alpha}_{ij}^\top(\mathbf{A}_{ij}\mathbf{x}_i - \mathbf{y}_{ij}) + \boldsymbol{\beta}_{ij}^\top(\mathbf{M}_{ij}\mathbf{x}_j - \mathbf{y}_{ij})\Big] \\ &+ \sum_{(i,j)\in\mathcal{E}} \Big[\frac{\gamma_i}{2}\|\mathbf{A}_{ij}\mathbf{x}_i - \mathbf{y}_{ij}\|^2 + \frac{\gamma_j}{2}\|\mathbf{M}_{ij}\mathbf{x}_j - \mathbf{y}_{ij}\|^2\Big]. \end{aligned} \tag{A.4}$$

PG-ADMM updates the variables following Aybat et al. (2018) as:

$$(\mathbf{x}_i,\mathbf{X}_i)^{k+1} = \Pi_{\mathcal{I}_i}((\mathbf{x}_i,\mathbf{X}_i)^k - c_i\nabla\phi(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta})^k) \tag{A.5}$$

$$\begin{aligned} \mathbf{y}_{ij}^{k+1} = \operatorname*{argmin}_{\mathbf{y}_{ij}} \{ & -(\boldsymbol{\alpha}_{ij}^k + \boldsymbol{\beta}_{ij}^k)^\top \mathbf{y}_{ij} \\ & + \frac{\gamma_i}{2}\|\mathbf{A}_{ij}\mathbf{x}_i^{k+1} - \mathbf{y}_{ij}\|^2 + \frac{\gamma_j}{2}\|\mathbf{M}_{ij}\mathbf{x}_j^k - \mathbf{y}_{ij}\|^2\} \end{aligned} \tag{A.6}$$

$$\boldsymbol{\alpha}_{ij}^{k+1} = \boldsymbol{\alpha}_{ij}^k + \gamma_i(\mathbf{A}_{ij}\mathbf{x}_i^{k+1} - \mathbf{y}_{ij}^{k+1}) \tag{A.7}$$

$$\boldsymbol{\beta}_{ij}^{k+1} = \boldsymbol{\beta}_{ij}^k + \gamma_j(\mathbf{M}_{ij}\mathbf{x}_j^{k+1} - \mathbf{y}_{ij}^{k+1}), \tag{A.8}$$

where (A.5) is for all $i\in\mathcal{N}$ and the gradient of ϕ is with respect to $(\mathbf{x}_i,\mathbf{X}_i)$. Moreover, (A.6), (A.7) and (A.8) are for all $(i,j)\in\mathcal{E}$. Note that (A.6) implies that $\mathbf{y}_{ij}$ admits a closed form update. Without much effort, one can show that this closed form update together with (A.7) and (A.8) imply that for all $(i,j)\in\mathcal{E}$, $\boldsymbol{\alpha}_{ij}^k + \boldsymbol{\beta}_{ij}^k = 0$ for all $k\geq 0$ if one initializes $\boldsymbol{\alpha}^0 = \boldsymbol{\beta}^0 = \mathbf{0}$. This renders $\mathbf{y}_{ij}$ being updated as:

$$\mathbf{y}_{ij}^k = \frac{\gamma_i\mathbf{A}_{ij}\mathbf{x}_i^k + \gamma_j\mathbf{M}_{ij}\mathbf{x}_j^k}{\gamma_i+\gamma_j}, \quad \forall k\geq 1. \tag{A.9}$$

Substituting (A.9) in (A.7) and (A.8) yields:

$$\boldsymbol{\alpha}_{ij}^k = \frac{\gamma_i\gamma_j}{\gamma_i+\gamma_j}\sum_{\ell=1}^k(\mathbf{A}_{ij}\mathbf{x}_i^\ell - \mathbf{M}_{ij}\mathbf{x}_j^\ell) \tag{A.10}$$

$$\boldsymbol{\beta}_{ij}^k = \frac{\gamma_i\gamma_j}{\gamma_i+\gamma_j}\sum_{\ell=1}^k(\mathbf{M}_{ij}\mathbf{x}_j^\ell - \mathbf{A}_{ij}\mathbf{x}_i^\ell) \tag{A.11}$$

Using (A.9), (A.10) and (A.11), one can show that $\nabla_{\mathbf{x}_i}\phi(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta})^k$ is given by:

$$\begin{aligned} &\sum_{j:(i,j)\in\mathcal{E}} \frac{\gamma_i\gamma_j\mathbf{A}_{ij}^\top}{\gamma_i+\gamma_j}\left[\big(\mathbf{A}_{ij}\mathbf{x}_i^k - \mathbf{M}_{ij}\mathbf{x}_j^k\big) + \sum_{\ell=1}^k\big(\mathbf{A}_{ij}\mathbf{x}_i^\ell - \mathbf{M}_{ij}\mathbf{x}_j^\ell\big)\right] \\ &+ \sum_{j:(j,i)\in\mathcal{E}} \frac{\gamma_i\gamma_j\mathbf{M}_{ji}^\top}{\gamma_i+\gamma_j}\left[\big(\mathbf{M}_{ji}\mathbf{x}_i^k - \mathbf{A}_{ji}\mathbf{x}_j^k\big) + \sum_{\ell=1}^k\big(\mathbf{M}_{ji}\mathbf{x}_i^\ell - \mathbf{A}_{ji}\mathbf{x}_j^\ell\big)\right] \end{aligned} \tag{A.12}$$

Examining the structure of $\mathbf{A}_{ij}$ and $\mathbf{M}_{ij}$ for all $(i,j)\in\mathcal{E}$, one concludes that for each $i\in\mathcal{N}$, $\nabla_{xii}\phi(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta})^k$ is given by:

$$\nabla f_i(x_{ii}^k) + \sum_{j\in\mathcal{N}_i}\frac{\gamma_i\gamma_j}{\gamma_i+\gamma_j}\left[(x_{ii}^k - d_{ji}^k) + \sum_{\ell=1}^k(x_{ii}^\ell - x_{ji}^\ell)\right], \tag{A.13}$$

and for each $i\in\mathcal{N}$ and $j\in\mathcal{N}_i$, $\nabla_{xij}\phi(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta})^k$ is given by:

$$\frac{\gamma_i\gamma_j}{\gamma_i+\gamma_j}\left[(x_{ij}^k - x_{jj}^k) + \sum_{\ell=1}^k(x_{ij}^\ell - x_{jj}^\ell)\right]. \tag{A.14}$$

Finally, using the definition of Γ in Section 4, set for each $i\in\mathcal{N}$, $s_{ii}^k = \sum_{j\in\mathcal{O}_i}\Gamma_{ij}x_{ji}^k$ and $s_{ij}^k = \Gamma_{ij}(x_{jj}^k - x_{ij}^k)$ for $k\geq 0$, and let $\mathbf{p}_i^k = \sum_{\ell=1}^k\mathbf{s}_i^\ell$ with the initialization $\mathbf{p}_i^0 = \mathbf{0}$, where $\mathbf{s}_i = [s_{ij}]_{i\in\mathcal{O}_i}$. It follows from (A.13) and (A.14) that

$$\nabla_{\mathbf{x}_i}\phi(\mathbf{x},\mathbf{X},\mathbf{y},\boldsymbol{\alpha},\boldsymbol{\beta})^k = \mathbf{p}_i^k + \mathbf{s}_i^k. \tag{A.15}$$

We substitute (A.15) in (A.5) to get the distributed algorithm outlined in Algorithm 1.