A Two-level ADMM Algorithm for AC OPF with Global Convergence Guarantees

Kaizhao Sun and Xu Andy Sun, *Senior Menmber, IEEE*

***Abstract*—This paper proposes a two-level distributed algorithmic framework for solving the AC optimal power flow (OPF) problem with convergence guarantees. The presence of highly nonconvex constraints in OPF poses significant challenges to distributed algorithms based on the alternating direction method of multipliers (ADMM). In particular, convergence is not provably guaranteed for nonconvex network optimization problems like AC OPF. In order to overcome this difficulty, we propose a new distributed reformulation for AC OPF and a two-level ADMM algorithm that goes beyond the standard framework of ADMM. We establish the global convergence and iteration complexity of the proposed algorithm under mild assumptions. Extensive numerical experiments over some largest test cases from NESTA and PGLib-OPF (up to 30,000-bus systems) demonstrate advantages of the proposed algorithm over existing ADMM variants in terms of convergence, scalability, and robustness. Moreover, under appropriate parallel implementation, the proposed algorithm exhibits fast convergence comparable to or even better than the state-of-the-art centralized solver.**

***Index Terms*—Distributed optimization, optimal power flow, augmented Lagrangian method, alternating direction method of multipliers.**

I. Introduction

The AC optimal power flow (OPF) is a basic building block in electric power grid operation and planning. It is a highly nonconvex optimization problem, due to nonlinear power flow equations, and is shown to be an NP-hard decision problem [1], [2]. Any computational method to be deployed in power system practice should meet the stringent requirement that the algorithm has guaranteed robust performance and requires minimal tuning and intervention in face of variations in system conditions. Moreover, to effectively coordinate multiple regions in a large power grid, distributed algorithms that do not require sharing critical private information between regions should be highly desirable. However, such a goal has remained challenging for solving large-scale AC-OPF problems. Many existing algorithms are only suited for centralized operation. Most distributed or decentralized algorithms for solving AC-OPF do not have guaranteed convergence performance and require extensive parameter tuning and experimentation.

In this paper, we develop a distributed algorithm for solving large-scale AC OPF problems to stationary points, and the proposed algorithm is proven to have global convergence.

The authors are with the H. Milton Stewart of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332 USA (email: ksun46@gatech.edu; andy.sun@isye.gatech.edu).

Manuscript received August 27, 2020; revised January 28, 2021; accepted March 28, 2021.

A. Literature Review

The research community has extensively studied local nonlinear optimization methods such as the interior point methods e.g. [3], [4]. Another line of research looks into convex relaxations of AC OPF and has drawn significant attentions in recent years. In particular, the semidefinite programming (SDP) relaxation is firstly applied to the OPF problem in [5], and sufficient conditions are studied to guarantee the exactness of SDP relaxations [6]. However, SDP suffers from expensive computation cost for large-scale problems, while the second-order cone programming (SOCP) relaxation initially proposed in [7] offers a favorable alternative. The strong SOCP relaxation proposed in [8] is shown to be close to or dominates the SDP relaxation, while the computation time of SOCP can be orders of magnitude faster than SDP. However, obtaining a primal feasible solution is a common challenge facing these convex relaxation methods.

The alternating direction method of multipliers (ADMM) offers a powerful framework for distributed computation. Sun et al. [9] applied ADMM to decompose the computation down to each individual bus, and observe the convergence is sensitive to initial conditions. Erseghe [10], [11] studied the case where the network is divided into overlapping subregions, and voltage information of shared buses are duplicated by their connected subregions. In [10], ADMM is directly applied to the underlying distributed reformulation, and convergence is established by assuming nonconvex OPF and ADMM subproblems have zero duality gaps. In [11], standard techniques used in the Augmented Lagrangian Method (ALM) are adopted inside ADMM. Subsequential convergence is proved under the assumption that the penalty parameter stays finite, which is basically assuming the algorithm converges to a feasible solution. The assumption is quite strong as quadratic penalty in general does not admit an exact penalization for nonconvex problems [12]. A more recent work [13] applied ADMM to a component-based distributed reformulation of AC OPF. The ADMM penalty is adaptively changed in every iteration, and the resulting algorithm numerically converges for various networks under adaptive hyperparameter tuning. In summary, existing ADMM-based algorithms either directly apply ADMM or its variant as a heuristic, or rely on strong assumptions to establish asymptotic convergence. See [14] for a recent survey on distributed optimization techniques for OPF.

Another related work is the Augmented Lagrangian Alternating Direction Inexact Newton (ALADIN) algorithm proposed in [15] and [16], which is a mixture of sequential quadratic programming (SQP) and ADMM. The algorithm requires a centralized consensus step that solves an equality-constrained nonconvex quadratic program, which uses the Hessian information of an augmented Lagrangian function. The authors established global convergence of ALADIN to an approximate stationary point when each agent's subproblem is solved to global optimality. However, if subproblems cannot be solved to global optimum due to nonconvexity or numerical considerations, then a good initial point close to a stationary point of the original problem is needed and only local convergence of ALADIN is guaranteed. Due to the use of second-order Hessian information in an SQP framework, ALADIN enjoys local quadratic convergence properties under some technical assumptions.

B. Contribution

In this paper, we address the convergence issues of ADMM by proposing a two-level distributed algorithmic framework. The proposed framework is motivated by our observation that some crucial structure necessary for the convergence of nonconvex ADMM is absent in traditional distributed reformulations of AC OPF, and we overcome such technical difficulty by embedding a three-block ADMM inside the classic ALM framework. We present the global convergence to a stationary point and iteration complexity results of the proposed framework, which rely on mild and realistic assumptions. We demonstrate the convergence, scalability, and robustness of the proposed algorithm over some largest test cases from NESTA [17] and PGLib-OPF [18], on which existing ADMM variants may fail to converge. Generically, distributed algorithms can be slow due to limited access to global information and communication delay; however, we show that, with proper parallel implementation, the proposed algorithm achieves fast convergence close to or even better than centralized solver.

C. Notation and Organization

Throughout this paper, we use $\mathbb{R}^n$ to denote the n-dimensional real Euclidean space; the inner product of $x, y \in \mathbb{R}^n$ is denoted by $\langle x, y\rangle$; the Euclidean norm is denoted by $\|x\|$, and the ℓ_∞ norm is denoted by $\|x\|_\infty$. When x consists of p subvectors, we write $x = (x_1, \cdots, x_p)$. For a matrix $A \in \mathbb{R}^{m\times n}$, we use $\mathrm{Im}(A)$ to denote its column space. We use $\mathbb{Z}_{++}$ to denote the set of positive integers, and $[n] = \{1, \cdots, n\}$. For a closed set $C \subset \mathbb{R}^n$, the orthogonal projection onto C is denoted by $\mathrm{Proj}_C(x)$, and the indicator function of C is denoted by $\delta_C(x)$, which takes value 0 if $x \in C$ and $+\infty$ otherwise.

The rest of this paper is organized as follows. In section II, we review the AC OPF problem and nonconvex ADMM literature. Then in section III, we present a new distributed reformulation and the proposed two-level algorithm. In section IV, we state the main convergence results of the proposed two-level algorithm, and discuss related convergence issues. Finally, we present computational experiments in section V, and conclude this paper in section VI.

II. Background

A. AC OPF Formulation

Consider a power network $G(\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ denotes the set of buses and $\mathcal{E}$ denotes the set of transmission lines. Let $\delta(i)$ be the set of neighbours of $i \in \mathcal{N}$. Let $Y = G + \mathbf{j}B$ denote the complex nodal admittance matrix, where $\mathbf{j} = \sqrt{-1}$ and $G, B \in \mathbb{R}^{|\mathcal{N}|\times|\mathcal{N}|}$. Let p_i^g, q_i^g (resp. p_i^d, q_i^d) be the real and reactive power produced by generator(s) (resp. loads) at bus i; if there is no generator (resp. load) attached to bus i, then p_i^g, q_i^g (resp. p_i^d, q_i^d) are set to 0. The complex voltage v_i at bus i can be expressed by its real and imaginary parts as $v_i = e_i + \mathbf{j}f_i$. The rectangular formulation of AC OPF is given as

$$\min \quad \sum_{i\in\mathcal{N}} f_i(p_i^g) \tag{1a}$$

$$\text{s.t.} \quad p_i^g - p_i^d = G_{ii}(e_i^2 + f_i^2) + \sum_{j\in\delta(i)} G_{ij}(e_ie_j + f_if_j) - B_{ij}(e_if_j - e_jf_i), \quad \forall i \in \mathcal{N}, \tag{1b}$$

$$q_i^g - q_i^d = -B_{ii}(e_i^2 + f_i^2) + \sum_{j\in\delta(i)} -B_{ij}(e_ie_j + f_if_j) - G_{ij}(e_if_j - e_jf_i), \ \forall i \in \mathcal{N}, \tag{1c}$$

$$p_{ij}^2 + q_{ij}^2 \le \overline{s}_{ij}^2, \qquad \forall (i,j) \in \mathcal{E}, \tag{1d}$$

$$\underline{v}_i^2 \le e_i^2 + f_i^2 \le \overline{v}_i^2, \qquad \forall i \in \mathcal{N}, \tag{1e}$$

$$\underline{p_i^g} \le p_i^g \le \overline{p_i^g}, \quad \underline{q_i^g} \le q_i^g \le \overline{q_i^g}, \qquad \forall i \in \mathcal{N}, \tag{1f}$$

where

$$p_{ij} = -G_{ij}(e_i^2 + f_i^2 - e_ie_j - f_if_j) - B_{ij}(e_if_j - e_jf_i), \tag{2a}$$

$$q_{ij} = B_{ij}(e_i^2 + f_i^2 - e_ie_j - f_if_j) - G_{ij}(e_if_j - e_jf_i). \tag{2b}$$

In (1a), the objective $f_i(p_i^g)$ represents the real power generation cost at bus i. Constraints (1b) and (1c) correspond to real and reactive power injection balance at bus i. The real and reactive power flow p_{ij}, q_{ij} on line (i,j) are given in (2), and (1d) restricts the apparent power flow on each transmission line. Constraints (1e)-(1f) limit voltage magnitude, real power output, and reactive power output at each bus to its physical capacity. Since the objective is typically linear or quadratic in real generation, formulation (1) is a nonconvex quadratically constrained quadratic program (QCQP) problem.

B. Nonconvex ADMM

ADMM was proposed in 1970s [19], [20] and regarded as a close variant of ALM [21], [22]. The standard ADMM framework consists of a Gauss-Seidel type update on blocks of variables in each ALM subproblem and then a dual update using current primal residuals. The update of each individual block can be decomposed and carried out in parallel given that certain separable structures are available. More recently, researchers [23]–[25] realized that the ADMM framework can be used to solve more complicated nonconvex multi-block problems in the form

$$\min_{x=(x_1,\cdots,x_p)} \quad \sum_{i=1}^{p} f_i(x_i) + g(x) \tag{3a}$$

$$\text{s.t.} \quad \sum_{i=1}^{p} A_i x_i = b, \quad x_i \in \mathcal{X}_i \quad \forall i \in [p], \tag{3b}$$

where there are p blocks of variables $x_i \in \mathbb{R}^{n_i}$ for $i \in [p]$, and f_i's, $\mathcal{X}_i$'s, and g can be potentially nonconvex. Different assumptions on the problem data are proposed to ensure global convergence to stationary solutions and in general an iteration complexity of $\mathcal{O}(1/\epsilon^2)$ is expected. Though motivated by different applications and adopting different analysis techniques, all these convergence results on nonconvex ADMM rely on the following three assumptions:

(a) All nonconvex subproblems need to be solved to global optimality;
(b) The functions f_p and g are Lipschitz differentiable, and $\mathcal{X}_p = \mathbb{R}^{n_p}$;
(c) The column space of the last block coefficient matrix A_p is sufficiently large, i.e., $\text{Im}([A_1,\cdots,A_{p-1},b]) \subseteq \text{Im}(A_p)$, where $[A_1,\cdots,A_{p-1},b]$ is the matrix concatenated by columns of $A_1,\cdots,A_{p-1}$ and b.

These three assumptions together restrict the application of ADMM on the OPF problem. No matter what reformulation of (1) is used, ADMM subproblems would still have highly nonconvex constraints, so Assumption (a) is unrealistic for the OPF problem. Assumption (b) is used to provide control of dual variables using primal variables in ADMM. This control is necessary for the construction of a potential function in the convergence analysis. Assumption (c) is needed to guarantee feasibility. If Assumption (c) is not satisfied, then it is possible that ADMM will converge to some $x_1^*,\cdots,x_{p-1}^*$ such that the linear system $A_p x_p = b - \sum_{i=1}^{p-1} A_i x_i^*$ has no solution, and hence ADMM fails to find a feasible solution. It turns out that Assumptions (b) and (c) cannot be satisfied simultaneously if ADMM were to achieve parallel computation among different agents [26]. Such limitation motivates us to go beyond the framework of ADMM.

III. A New Distributed Reformulation and a Two-level ADMM Algorithm

A. A New Distributed Reformulation

Suppose the network G is partitioned into R subregions $\mathcal{R}_1,\cdots,\mathcal{R}_R \subseteq \mathcal{N}$, each of which is assigned to a local control or operating agent, i.e., $\mathcal{R}_i$'s are disjoint and $\cup_{i=1}^{R}\mathcal{R}_i = \mathcal{N}$. We say $(i,j) \in \mathcal{E}$ is a tie-line if $i \in \mathcal{R}_r$, $j \in \mathcal{R}_l$, and $r \neq l$. Agent r controls variables $x_i = (p_i^g, q_i^g, e_i, f_i)$ for all $i \in \mathcal{R}_r$. We say $i \in \mathcal{R}_r$ is a *boundary bus* of $\mathcal{R}_r$ if i is connected to another subregion through a tie-line, and denote the set of boundary buses in $\mathcal{R}_r$ by $B(\mathcal{R}_r)$. We extend the notation $\delta(\mathcal{R}_r)$ to denote the set of all buses connected to (but not in) $\mathcal{R}_r$ by some tie-lines.

The constraints of OPF couple adjacent agents. For example, suppose (i,j) is a tie-line where $i \in \mathcal{R}_r$ and $j \in \mathcal{R}_l$. Agent r requires the information of variables (e_j, f_j) to construct constraints (1b)-(1d); however, agent r cannot directly access (e_j, f_j) as they are controlled by agent l, and agent l faces the same situation. In order for these two agents to solve their localized problems in parallel, it is necessary to break the coupling by introducing auxiliary variables. We let each agent r keep additional variables $x_j^r = (e_j^r, f_j^r)$ for all $j \in \delta(\mathcal{R}_r)$. A direct consequence is that all constraints in formulation (1) are decomposed to local agents. For example, for each $i \in \mathcal{R}_r$, constraints (1b)-(1d) can be rewritten as

$$\begin{aligned} p_i^g - p_i^d = {} & G_{ii}(e_i^2 + f_i^2) + \\ & \sum_{j \in \delta(i)\cap\mathcal{R}_r} G_{ij}(e_i e_j + f_i f_j) - B_{ij}(e_i f_j - e_j f_i) + \\ & \sum_{j \in \delta(i)\cap\delta(\mathcal{R}_r)} G_{ij}(e_i e_j^r + f_i f_j^r) - B_{ij}(e_i f_j^r - e_j^r f_i), \end{aligned} \tag{4a}$$

$$\begin{aligned} q_i^g - q_i^d = {} & -B_{ii}(e_i^2 + f_i^2) + \\ & \sum_{j \in \delta(i)\cap\mathcal{R}_r} -B_{ij}(e_i e_j + f_i f_j) - G_{ij}(e_i f_j - e_j f_i) + \\ & \sum_{j \in \delta(i)\cap\delta(\mathcal{R}_r)} -B_{ij}(e_i e_j^r + f_i f_j^r) - G_{ij}(e_i^r f_j - e_j^r f_i), \end{aligned} \tag{4b}$$

$$p_{ij}^2 + q_{ij}^2 \leq \bar{s}_{ij}^2, \qquad \forall (i,j) \in \mathcal{E}, j \in \mathcal{R}_r, \tag{4c}$$

$${p_{ij}^r}^2 + {q_{ij}^r}^2 \leq \bar{s}_{ij}^2, \qquad \forall (i,j) \in \mathcal{E}, j \notin \mathcal{R}_r, \tag{4d}$$

where p_{ij}, q_{ij} are given in (2), and

$$p_{ij}^r = -G_{ij}(e_i^2 + f_i^2 - e_i e_j^r - f_i f_j^r) - B_{ij}(e_i f_j^r - e_j^r f_i), \tag{5a}$$

$$q_{ij}^r = B_{ij}(e_i^2 + f_i^2 - e_i e_j^r - f_i f_j^r) - G_{ij}(e_i f_j^r - e_j^r f_i). \tag{5b}$$

Notice that all variables appeared in (4) are controlled by agent r, and all such variables are grouped together and denoted by $x^r = \big(\{x_i\}_{i\in\mathcal{R}_r}, \{x_j^r\}_{j\in\delta(\mathcal{R}_r)}\big)$. Moreover, local nonconvex constraints of $\mathcal{R}_r$ can be conveniently expressed as

$$\mathcal{X}_r = \{x^r \mid \text{(1e)} - \text{(1f)}, (4)\ \forall i \in \mathcal{R}_r\}, \tag{6}$$

where, allowing a minor clash of notation, (1e) and (1f) are meant to be satisfied for all $i \in \mathcal{R}_r$ in (6). Notice that for every $j \in \cup_{r=1}^{R}\delta(\mathcal{R}_r)$, bus j is connected to some tie-line, and thus at least two regions need to keep a local copy of (e_j, f_j). We use $R(j)$ to denote the subregion where bus j is located, and $N(j)$ to denote the set of subregions that share a tie-line with $R(j)$ through bus j. Naturally we want to impose consensus on local copies of the same variables:

$$e_j^l = e_j, \ f_j^l = f_j, \quad \forall l \in N(j). \tag{7}$$

The regional decoupling techniques used in (4)-(7) have appeared in the early work by Kim and Baldick [27] among others, where the authors applied a linearized proximal ALM to a distributed OPF formulation. A graphical illustration can be found in [27].

When ADMM is considered, agents from $N(j)$ and agent $R(j)$ will need to alternatively solve their subproblems in order to parallelize the computation. As we will explain in Section III-B, ADMM does not guarantee convergence when both subproblems carry nonconvex functional constraints. In order to solve this issue, we follow the idea proposed in [26] by using a global copy $\bar{x}_j = (\bar{e}_j, \bar{f}_j)$ and local slack variables $z_j^l = (z_{e_j}^l, z_{f_j}^l)$ for $l \in N(j) \cup \{R(j)\}$. For notational consistency, we also write $x_j^{R(j)} = (e_j, f_j) = (e_j^{R(j)}, f_j^{R(j)})$. The consensus is then achieved through

$$e_j^l - \bar{e}_j + z_{e_j}^l = 0, \quad z_{e_j}^l = 0, \quad \forall l \in N(j) \cup \{R(j)\}, \tag{8a}$$
$$f_j^l - \bar{f}_j + z_{f_j}^l = 0, \quad z_{f_j}^l = 0, \quad \forall l \in N(j) \cup \{R(j)\}. \tag{8b}$$

Denote $x = (\{x^r\}_{r\in[R]})$, $\bar{x} = (\{\bar{x}_j\}_{j\in\cup_r \delta(\mathcal{R}_r)})$, and $z = (\{z_j^l\}_{l\in N(j)\cup\{R(j)\}, j\in\cup_r\delta(\mathcal{R}_r)})$. Notice that $\bar{x}$ and z are only introduced for boundary buses. Now we can abstract the AC OPF problem as:

$$\begin{aligned} \min_{x,\bar{x},z} \quad & \sum_{r=1}^{R} c_r(x^r) := \sum_{r=1}^{R} \left(\sum_{i\in\mathcal{R}_r} f_i(p_i^g) \right) \\ \text{s.t.} \quad & Ax + B\bar{x} + z = 0, \\ & x^r \in \mathcal{X}_r \ \forall r \in [R], \ \bar{x} \in \bar{\mathcal{X}}, \ z = 0. \end{aligned} \tag{9}$$

The objective $c_r(x^r)$ is the sum of all generators' costs in $\mathcal{R}_r$. The linear coupling constraints (8) is compactly expressed as $Ax+B\bar{x}+z = 0$ with matrices A and B of proper dimensions. Each local agent r controls local OPF constraints $\mathcal{X}_r$ defined in (6). Moreover, without changing the feasible region of (1), we may restrict $\bar{x}$ inside some convex set $\bar{\mathcal{X}}$. For example, we can simply let $\bar{\mathcal{X}}$ be a hypercube

$$\bar{\mathcal{X}} = \prod_{j\in\cup_r\delta(\mathcal{R}_r)} \bar{\mathcal{X}}_j := \prod_{j\in\cup_r\delta(\mathcal{R}_r)} \left\{ \bar{x}_j | \ \|\bar{x}_j\|_\infty \le \bar{v}_j \right\}, \tag{10}$$

which is compact and easy to project onto.

Next we define stationarity for problem (9). After projecting out the slack variable z, the Lagrangian function of (9) is

$$L(x,\bar{x},y) = \sum_{r=1}^{R} (c_r(x^r) + \delta_{\mathcal{X}_r}(x^r)) + \delta_{\bar{\mathcal{X}}}(\bar{x}) + \langle y, Ax + B\bar{x}\rangle. \tag{11}$$

We use $\partial f(\cdot)$ to denote the general subdifferential of a proper lower semi-continuous function $f : \mathbb{R}^n \to \mathbb{R}$ [28, Def 8.3], and $N_C(x)$ to denote the general normal cone of C at $x \in C$ [28, Def 6.3].

Definition 1. *We say $(x, \bar{x}, y)$ is an ϵ-stationary point of problem* (9) *if there exist (d_1, d_2, d_3) such that $\max\{\|d_1\|, \|d_2\|, \|d_3\|\} \le \epsilon$ where*

$$d_1 \in \partial \left(\sum_{r=1}^{R} c_r(x^r) + \delta_{\mathcal{X}_r}(x^r) \right) + A^\top y, \tag{12a}$$
$$d_2 \in N_{\bar{\mathcal{X}}} + B^\top y, \tag{12b}$$
$$d_3 = Ax + B\bar{x}; \tag{12c}$$

or equivalently, $(d_1, d_2, d_3) \in \partial L(x, \bar{x}, y)$. We simply say $(x, \bar{x}, y)$ is a stationary point if $0 \in \partial L(x, \bar{x}, y)$ or $\epsilon = 0$.

If cost functions are concatenated as $c(x) = \sum_{r=1}^{R} c_r(x^r)$, which is assumed to be continuously differentiable, and let $\mathcal{X} = \prod_{r=1}^{R} \mathcal{X}_r$, then (12a) can be further reduced to

$$0 \in \nabla c(x) + N_{\mathcal{X}}(x) + A^\top y. \tag{13}$$

B. A Divergent Example for Two-block Nonconvex ADMM

Before presenting our proposed algorithm, we use a concrete example to demonstrate that the vanilla version of ADMM indeed may suffer divergence for OPF instances. Notice that without introducing slack variable z and the constraint $z = 0$ as in (9), the distributed OPF problem can be formulated as a two-block nonconvex problem

$$\min_{x\in\mathcal{X},\bar{x}\in\bar{\mathcal{X}}} \quad c(x) \quad \text{s.t.} \quad Ax + B\bar{x} = 0, \tag{14}$$

which was also directly used to develop distributed algorithms [10], [11]. We partition the IEEE `case30` available from [4] into three subregions, directly apply the vanilla version ADMM to the two-block formulation (14) with different penalty parameter ρ, and plot the *Infeasibility* $\|Ax^t + B\bar{x}^t\|$ and *Generation Cost* $c(x^t)$ as in Figure 1.

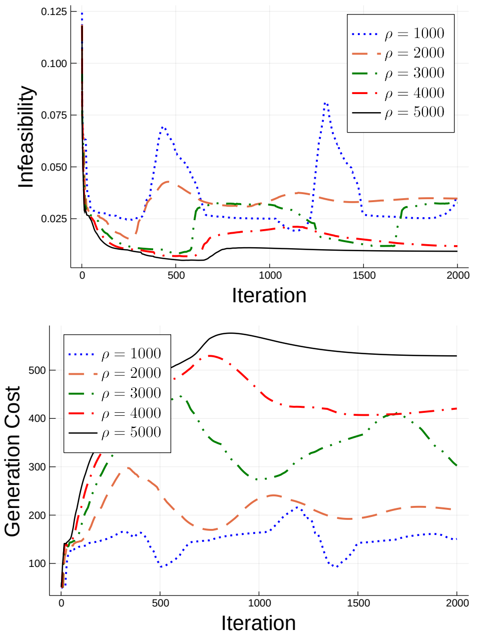

Fig. 1: Divergent Behaviour of Vanilla ADMM.

As we can see, the primal residual and generation costs exhibit oscillating patterns for $\rho \in \{1000, 2000, 3000, 4000\}$, and do not converge even when ADMM has performed 2000 iterations. For $\rho = 5000$, the primal residual converges to 0.0093, and the generation cost at termination is below the lower bound obtained from SOCP; these two facts indicate that ADMM indeed converges to an infeasible solution.

Such failures of ADMM to obtain feasible solutions result from the tension between the Assumptions (b) and (c) introduced in Section II-B. To be more specific, it is straightforward to verify that $\text{Im}(B) \subset \text{Im}(A)$: given a global copy $\bar{x}_j$, every local agent $l \in N(j) \cup R(j)$ can always keep the same value $x_j^l = \bar{x}_j$ so that the constraint $Ax + B\bar{x} = 0$ is satisfied. Therefore, to satisfy Assumption (c), A need to be the last block and B is the first block in ADMM. But since each agent's problem must consider local OPF constraints, which means $\mathcal{X}$ must be a nonconvex set. This violates the requirement that the problem of the last block must be unconstrained in Assumption (b). Therefore, Assumptions

(b) and (c) cannot be satisfied simultaneously. As a result, if we directly apply ADMM to the ACOPF problem, ADMM may fail to converge. Admittedly, we observe convergence when a even larger penalty ρ is used; however, we want to emphasize that despite the numerical success with adaptive parameter tuning [13], the traditional ADMM framework has no guarantee of convergence and could fail for nonconvex distributed OPF.

C. A New Two-level ADMM Algorithm

In this section we give a full description of the two-level algorithm applied to the OPF problem. The key idea is to dualize and penalize the constraint $z = 0$ in (9), and apply three-block ADMM to solve the augmented Lagrangian relaxation (ALR):

$$\begin{aligned} \min_{x\in\mathcal{X},\bar{x}\in\bar{\mathcal{X}},z} \quad & c(x) + \langle \lambda^k, z\rangle + \frac{\beta^k}{2}\|z\|^2 \\ \text{s.t.} \quad & Ax + B\bar{x} + z = 0, \end{aligned} \tag{15}$$

with some dual variable λ^k and penalty β^k, to an approximate stationary solution in the following sense.

Definition 2. *We say* $(x, \bar{x}, z, y)$ *is an* ϵ*-stationary point of problem* (15) *if there exists* (d_1, d_2, d_3) *such that* $\max\{\|d_1\|, \|d_2\|, \|d_3\|\} \le \epsilon$*, where* d_1 *and* d_2 *satisfy* (12a)-(12b)*,* $d_3 = Ax + B\bar{x} + z$*, and* $\lambda^k + \beta^k z + y = 0$.

Then at termination of ADMM, we update λ^{k+1} and β^{k+1} as in the classic ALM framework in order to drive z to 0. To summarize, the proposed algorithm consists of two levels: in the inner level (indexed by t), we apply ADMM to solve the ALR problem (15); in the outer level (indexed by k), we update the dual information $(\lambda^{k+1}, \beta^{k+1})$ using the solution $(x^k, \bar{x}^k, z^k)$ returned by ADMM, which falls into the category of ALM, and then restart the next inner level. See Algorithm 1 for a detailed description.

The inner level is presented in line 4-11 of Algorithm 1, where we apply ADMM to solve ALR (15). To facilitate understanding, define the augmented Lagrangian function associated with (15) as

$$\begin{aligned} L_\rho(x, \bar{x}, z, y) = {} & c(x) + \langle \lambda^k, z\rangle + \frac{\beta^k}{2}\|z\|^2 \\ & + \langle y, Ax + B\bar{x} + z\rangle + \frac{\rho}{2}\|Ax + B\bar{x} + z\|^2, \end{aligned} \tag{20}$$

where (λ^k, β^k) are considered as parameters. From a centralized point of view, equations (16)-(18) are the sequential minimization of L_ρ with respect to $x \in \mathcal{X}$, $\bar{x} \in \bar{\mathcal{X}}$, and z, respectively, which, together with the update of dual variable y in (19), constitute a single ADMM iteration. Next we describe ADMM from a local point of view. All agents simultaneously solve lower-dimensional nonconvex subproblems (16) by some nonlinear optimization solver. Then each agent $r \in [R]$ sends the current local estimate of voltage $(x_j^r)^{t+1}$ to $R(j)$ for all neighboring buses $j \in \delta(\mathcal{R}_r)$. For each bus j connected to a tie-line, we let agent $R(j)$ collect estimates $\{(x_j^l)^{t+1}\}_{l\in N(j)}$ and update global copy $(\bar{x}_j)^{t+1}$, though in practice any agents from $N(j)$ can be assigned for this task. Notice that the global copy update (17) involves a projection evaluation, which in general does not admit a closed-form solution; however, if (10) is used for $\bar{\mathcal{X}}$, then $\bar{x}_j^{t+1}$ is exactly the component-wise projection of the argument in (17) onto the box $\bar{\mathcal{X}}_j$. After agent $R(j)$ broadcasts $(\bar{x}_j)^{t+1}$ to agents from $N(j)$, all agents are then able to update the slack variable and dual variables as in (18) and (19).

When the inner-level iterates satisfy certain stopping criteria, all agents will update the outer-level dual variable λ^{k+1} and penalty β^{k+1}, as in line 13 of Algorithm 1, which we will elaborate in the next section.

Algorithm 1 : A Two-level ADMM Algorithm

```
1:  Initialize starting points $(x^0, \bar{x}^0) \in \mathcal{X}\times\bar{\mathcal{X}}$ and $\lambda^1 \in [\underline{\lambda}, \overline{\lambda}]$; $\beta^1 > 0$, $k \leftarrow 1$;
2:  while outer stopping criteria is not satisfied do
3:    Initialize $(x^0, \bar{x}^0, z^0, y^0)$ such that $\lambda^k + \beta^k z^0 + y^0 = 0$; $\rho \leftarrow 2\beta^k$, $t \leftarrow 1$;
4:    while inner stopping criteria is not satisfied do
5:      each agent $r \in [R]$ updates $(x^r)^{t+1}$ by solving :
          $$\begin{aligned} \min_{x^r\in\mathcal{X}_r} \; F_r^t(x^r) := {} & c_r(x^r) \\ & + \sum_{j\in\delta(\mathcal{R}_r)\cup B(\mathcal{R}_r)} \Big(\langle (y_j^r)^t, x_j^r\rangle \\ & + \frac{\rho}{2}\|x_j^r - (\bar{x}_j)^t + (z_j^r)^t\|^2\Big); \end{aligned} \tag{16}$$
6:      each agent $r \in [R]$ sends $(x_j^r)^{t+1}$ to $R(j)$ for $j \in \delta(\mathcal{R}_r)$, and receives $\big((x_i^l)^{t+1}, (z_i^l)^t, (y_i^l)^t\big)$ from every agent $l \in N(i)$ for all $i \in B(\mathcal{R}_r)$;
7:      each agent $r \in [R]$ updates global copy $\bar{x}_i^{t+1} =$
          $$\mathrm{Proj}_{\bar{\mathcal{X}}_i}\left(\frac{\sum_{l\in N(i)\cup R(i)}\left[(y_i^l)^t + \rho\left((x_i^l)^{t+1} + (z_i^l)^t\right)\right]}{(|N(i)|+1)\rho}\right) \tag{17}$$
        for all $i \in B(\mathcal{R}_r)$;
8:      each agent $r \in [R]$ sends $(\bar{x}_i)^{t+1}$ to agents in $N(i)$ for $i \in B(\mathcal{R}_r)$, and receives $(\bar{x}_j)^{t+1}$ from agent $R(j)$ for all $j \in \delta(\mathcal{R}_r)$;
9:      each agent $r \in [R]$ update local slack variable
          $$(z_j^r)^{t+1} = \frac{-(\lambda_j^r)^k - (y_j^r)^t - \rho\left((x_j^r)^{t+1} - (\bar{x}_j)^{t+1}\right)}{\beta^k + \rho} \tag{18}$$
        and dual variable
          $$(y_j^r)^{t+1} = (y_j^r)^t + \rho\left[(x_j^r)^{t+1} - (\bar{x}_j)^{t+1} + (z_j^r)^{t+1}\right] \tag{19}$$
        for all $j \in \delta(\mathcal{R}_r) \cup B(\mathcal{R}_r)$;
10:     $t \leftarrow t + 1$;
11:   end while
12:   denote the solution from the inner loop as $(x^k, \bar{x}^k, z^k)$;
13:   each agent $r \in [R]$ updates outer-level dual variable $(\lambda_j^r)^{k+1}$ for all $j \in \delta(\mathcal{R}_r) \cup B(\mathcal{R}_r)$ and penalty $\beta^{k+1}$;
14:   $k \leftarrow k + 1$;
15: end while
```

IV. Convergence and Related Issues

In this section, we state the convergence results of the two-level ADMM framework for AC OPF, and discuss related issues. We require the following mild assumptions.

Assumption 1. (a) *The objective $c_r(\cdot)$ is continuous differentiable. The functional constraints $\mathcal{X}_r$'s and $\bar{\mathcal{X}}$ are compact, and $\bar{\mathcal{X}}$ is convex.*

(b) *For any $t \in \mathbb{Z}_{++}$, every local agent $r \in [R]$ is able to find a stationary solution $(x^r)^{t+1}$ of subproblem (16) such that $F_r^t\left((x^r)^{t+1}\right) \leq F_r^t\left((x^r)^t\right)$.*

For Assumption 1(a), the objective function $c_r(\cdot)$ is usually linear or convex quadratic with respect to the argument; without loss of generality, we may assume the set $\mathcal{X}_r$ defined in (6) also enforces bounds on local copies $\{(e_j^r, f_j^r)\}_{j\in\delta(\mathcal{R}_r)}$, and thus $\mathcal{X}_r$ is ensured to be compact. The assumption on $\bar{\mathcal{X}}$ is justified in (10). We note that it is possible to allow c_r to be nonsmooth, e.g., piecewise linear, when its general subdifferential ∂c_r is well-defined and bounded over X_r.

Assumption 1(b) requires the nonconvex subproblem (16) to be solved to a stationary solution $(x^r)^{t+1}$ that has objective value no worse than that of the previous iterate $(x^r)^t$. We believe this assumption is reasonable if the nonlinear solver is warm-started with the previous solution $(x^r)^t$ in iteration $t+1$. For example, the nonlinear solver IPOPT [29] uses a procedure to accept a trial point if the objective value or constraint violation is reduced during its execution. Since we have a feasible solution $(x^r)^t \in \mathcal{X}_r$ to start with, it is reasonable to expect some improvement in the objective. We note that Assumption 1(b) is imposed on the *solution oracle* of subproblem (16), and does not impose any restriction on the initial point $(x^0, \bar{x}^0) \in \mathcal{X}\times\bar{\mathcal{X}}$ supplied to the overall two-level algorithm, which, therefore, still enjoys global convergence as shown below. In addition, Assumption 1(b) is much weaker than assuming $(x^r)^{t+1}$ is a local or global minimizer of (16), which is a common assumption in the literature on nonconvex ADMM.

A. Global Convergence

We consider two different rules for updating $(\lambda^{k+1}, \beta^{k+1})$. Let $c > 1$, $\theta \in [0, 1)$, and $\{\eta_k\}_k$ be a nonnegative sequence convergent to 0:

$$\lambda^{k+1} = \mathrm{Proj}_{[\underline{\lambda},\overline{\lambda}]}(\lambda^k + \beta^k z^k), \tag{21a}$$

$$\beta^{k+1} = \begin{cases} \beta^k & \text{if } \|z^k\| \leq \theta \|z^{k+1} \\ c\beta^k & \text{otherwise,} \end{cases} \tag{21b}$$

and

$$(\lambda^{k+1}, \beta^{k+1}) = \begin{cases} (\lambda^k + \beta^k z^k, \beta^k) & \text{if } \|z^k\| \leq \eta_k, \\ (\lambda^k, c\beta^k) & \text{otherwise.} \end{cases} \tag{22}$$

Theorem 1 (Global Convergence)**.** *Suppose Assumption 1 holds, and the k-th inner-level ADMM is solved to an ϵ_k-stationary point $(x^k, \bar{x}^k, z^k, y^k)$ of (15) such that $\epsilon_k \to 0$ as $k \to +\infty$. Moreover, the outer-level dual variable λ^{k+1} and penalty β^{k+1} are updated according to either (21) or (22). Then the following claims hold.*

1) *The sequence $\{(x^k, \bar{x}^k, z^k)\}_k$ is bounded, and therefore there exists at least one limit point $(x^*, \bar{x}^*, z^*)$, where $x^* \in \mathcal{X} = \prod_{r=1}^R \mathcal{X}_r$ and $\bar{x}^* \in \bar{X}$.*
2) *Either $(x^*, \bar{x}^*)$ is feasible, i.e., $Ax^* + B\bar{x}^* = 0$, or $(x^*, \bar{x}^*)$ is a stationary point of the feasibility problem*
$$\min_{x,\bar{x}} \ \frac{1}{2}\|Ax + B\bar{x}\|^2 \text{ s.t. } x \in \mathcal{X}, \bar{x} \in \bar{\mathcal{X}}. \tag{23}$$
3) *Suppose problem (9) is feasible and the set of stationary points is nonempty. Let $(x^*, \bar{x}^*, z^*)$ be a limit point, and $\{(x^{k_r}, \bar{x}^{k_r}, z^{k_r})\}_r$ be the subsequence convergent to it. If $\{y^{k_r}\}_r$ has a limit point y^*, then $(x^*, \bar{x}^*, y^*)$ is a stationary point of problem (9).*

Proof of Theorem 1 is provided in Appendix. We note that in part 3 of Theorem 1, we make the assumption that the dual variable $\{y^{k_r}\}_r$ has a limit point. This is a standard sequentially bounded constraint qualification (SBQC) [30].

B. Iteration Complexity

Theorem 2 (Iteration Complexity)**.** *Suppose Assumption 1 holds, and there exists $0 < \overline{L} < +\infty$ such that*

$$\begin{aligned}\overline{L} \geq \sum_{r=1}^R c_r((x^r)^0) + \langle \lambda^k, z^0\rangle + \frac{\beta^k}{2}\|z^0\|^2 \\ + \langle y^0, Ax^0 + B\bar{x}^0 + z^0\rangle + \frac{\rho}{2}\|Ax^0 + B\bar{x}^0 + z^0\|^2\end{aligned} \tag{24}$$

for all $k \in \mathbb{Z}_{++}$. Further assume each inner-level ADMM is solved to an ϵ-stationary point $(x^k, \bar{x}^k, z^k, y^k)$ of (15), the outer-level dual variable λ^{k+1} is updated by (21a), and $\beta^{k+1} = c^{k+1}\beta^0$ for some $c > 1$, $\beta^0 > 0$. Define

- $\tau = 2\max\{\|A\|, \|B\|, \frac{1}{4\beta^0}\}$,
- $M = \max_{\lambda\in[\underline{\lambda},\overline{\lambda}]} \|\lambda\|$,
- $\underline{L} = \min_{x\in\mathcal{X}} c(x) - M^2/\beta^0$,
- $r_{\max} = \max_{x\in\mathcal{X},\bar{x}\in\bar{\mathcal{X}}} \|Ax + B\bar{x}\|$ *(since $\mathcal{X}, \bar{\mathcal{X}}$ compact),*
- *and for $K \in \mathbb{Z}_{++}$,*

$$T(K) = \left\lceil \left(\frac{4\beta^0(\overline{L} - \underline{L})\tau^2 c}{c-1}\right)\left(\frac{c^K - 1}{\epsilon^2}\right)\right\rceil + K.$$

Then Algorithm 1 finds an ϵ-stationary solution of problem (9) in no more than

$$K_1 = \left\lceil \log_c\left(\frac{2(\overline{L} - \underline{L} + M r_{\max})}{\beta^0\epsilon^2}\right)\right\rceil$$

outer ALM iterations and $T(K_1) = \mathcal{O}(1/\epsilon^4)$ inner ADMM iterations. Moreover, if there exists some $\Lambda > 0$ such that $\|\lambda^k + \beta^k z^k\| \leq \Lambda$ for all outer index k, then Algorithm 1 finds an ϵ-stationary solution of problem (9) in no more than

$$K_2 = \max\left\{\left\lceil \log_c\left(\frac{1}{\beta^0\tau}\right)\right\rceil, \left\lceil \log_c\left(\frac{2(\Lambda + M)}{\beta^0\epsilon}\right)\right\rceil\right\}$$

outer ALM iterations and $T(K_2) = \mathcal{O}(1/\epsilon^3)$ inner ADMM iterations.

The proof of Theorem 2 is provided in Appendix. We make some remarks.

1) The assumption (24) can be satisfied trivially, for example, if a feasible solution $(x^0, \bar{x}^0)$ for (9) is known a

priori and $(x^0, \bar{x}^0, z^0 = 0)$ is always used to start inner ADMM. In this case, we can choose $\overline{L} = \max_{x \in \mathcal{X}} c(x)$.

2) In view of (8), we can calculate $\|A\|$ and $\|B\|$ directly. Each row of A has exactly one non-zero entry, and each column of A has at most one non-zero entry, so $A^\top A$ is a diagonal matrix with either 0 or 1 on the diagonal, and thus $\|A\| = 1$. The number of non-zero entries in each column of B specifies how many subregions keep a local copy of this global variable, so $B^\top B$ is also a diagonal matrix, and we have

$$\|B\| = \left(\max_{j \in \cup_{r=1}^{R} B(\mathcal{R}_r)} |N(j)| + 1 \right)^{1/2} \leq \sqrt{R}.$$

3) The complexity results suggest that a smaller M is preferred, and $M = 0$ corresponds to the penalty method. However, we empirically observe that a relatively large range for λ usually results in faster convergence, which is also better than the $\mathcal{O}(1/\epsilon^3)$ or $\mathcal{O}(1/\epsilon^4)$ iteration upper bound. We believe such phenomena can be explained by the local convergence properties of ALM.

V. Numerical Experiments

In this section, we demonstrate the performance of the proposed algorithmic framework. All codes are written in the Julia programming language 1.2.0, and implemented on a Red Hat Enterprise Linux Server 7.6 with 85 Intel 2.10GHz CPUs. All nonlinear constrained problems are modeled using the JuMP optimization package [31] and solved by IPOPT with linear solver MA57 [1].

A. Network Information and Partition Generation

We experiment on four networks: `case9241_pegase` (9K), `case13659_pegase` (13K) from NESTA [17], and `case2848_rte` (2K), `case30000_goc` (30K) from PGLib-OPF [18]. See Table I for centralized information. For

TABLE I: Centralized Information.

Case	AC Obj.	AC Time(s)	SOCP Obj.	SOCP Time(s)	Gap (%)
9K	315913.26	51.13	310382.99	1224.81	1.76
13K	386117.10	160.10	380262.34	990.50	1.51
2K	1286608.20	8.75	1285025.40	18.62	0.12
30K	1034405.63	1032.92	1003867.51	256.20	2.95

networks 2K and 30K, we use linear cost for all generators to enhance numerical stability, which we will elaborate later in Section V-C. We generate different partitions using the multilevel k-way partitioning algorithm [32] on the underlying graph, which is available from the Julia wrapper of the Metis library `Metis.jl`. We use the suffix "-R" to indicate that a network is partitioned into R subregions, i.e., 9K-25 refers to network 9K with 25 subregions.

[1]The linear solver MA57 is used for both the centralized algorithm (IPOPT) and the proposed distributed algorithm. It is an interesting research question to fully test IPOPT with parallel linear solvers such as Pardiso or MA97.

B. Three Acceleration Heuristics

It is known that the choice of penalty parameter ρ significantly affects the convergence of ADMM and can potentially accelerate the algorithm [13]. Indeed, we observed that for some large cases, the inner-level ADMM may suffer from slow convergence to high accuracy when a constant penalty ρ is used. As a result, given parameters $\theta \in [0, 1)$ and $\gamma > 1$, we propose three different heuristics to properly accelerate Algorithm 1.

1) Adaptive ADMM penalty (TL-1): the inner-level ADMM penalty is indexed as ρ^t and updated as follows: $\rho^{t+1} = \gamma\rho^t$ if $\|Ax^{t+1} + B\bar{x}^{t+1} + z^{t+1}\| > \theta\|Ax^t + B\bar{x}^t + z^t\|$, and $\rho^{t+1} = \rho^t$ otherwise; in words, we increase the ADMM penalty if the three-block residual $\|Ax^{t+1} + B\bar{x}^{t+1} + z^{t+1}\|$ does not decrease sufficiently.
2) Different ADMM penalties (TL-2): we assign a different ADMM penalty for each row of the coupling constraint $Ax + B\bar{x} + z = 0$, and each penalty is updated according to the first heuristic, where the violation of each single constraint is measured, and the corresponding penalty is adjusted. Notice the ALM penalty β is a fixed constant for all components of z during the inner ADMM.
3) Different ALM penalties (TL-3): we assign a different ALM penalty β_i^t for each component z_i of the slack variable, and also update it inside ADMM iterations: $\beta_i^{t+1} = \gamma\beta_i^t$ if $|z_i^{t+1}| > \theta|z_i^t|$, and $\beta_i^{t+1} = \beta_i^t$ otherwise; the corresponding ADMM penalty ρ_i^t is always assigned to be $2\beta_i^t$, as required in our analysis. When the k-th ADMM terminates, current values of z_i^k and β_i^k are used to update outer level dual variable λ_i^{k+1}.

We note that the first two heuristics have been used to accelerate ADMM, while the last heuristic also penalizes the slack variable z adaptively in ADMM iterations.

C. Implementation Details

Parallelization of nonconvex subproblems: Each JuMP model carrying a subregion's localized OPF constraints is initialized on a core. During each (inner) iteration, these models are solved in parallel on different cores by IPOPT, which consist of the major computation of the algorithm. Multiple models on the same core are solved sequentially. Then current local solutions are gathered through the master node, and auxiliary primal variables and dual variables in different subregions are updated in closed form.

Parameters and Initialization: For the heuristics introduced in the previous subsection, we set $\gamma = 6.0$, $\theta = 0.8$; for the first two heuristics, when ADMM terminates, the outer-level penalty is updated as $\beta^{k+1} = c\beta^k$ where $c = 6.0$. The initial value β^0 is set to 1000.0, and an upper bound of $1.0e24$ is imposed in all penalty updates. Each component of λ is bounded between $\pm 1.0e12$. Flat start is used to initialize IPOPT: we choose $(e_i, f_i, p_i^g, q_i^g) = (1, 0, 0, 0)$ for all $i \in \mathcal{N}$. Dual variables y^0 and λ^0 are initialized with zeros.

Scaling of IPOPT: The proposed algorithm inevitably needs to deal with potentially large penalties and dual variables, and we observe that IPOPT will encounter numerical failures or produce oscillating solutions. To overcome this problem,

TABLE II: Performance of the Two-level ADMM Algorithm on 9K and 13K Networks.

			TL-1				TL-2				TL-3				
Case	Tie-line	Dim	Outer	Inner	Gap (%)	$\|r\|_\infty$	Outer	Inner	Gap (%)	$\|r\|_\infty$	Outer	Inner	Gap (%)	$\|r\|_\infty$	Avg. Time (s)
9K-25	357	2084	88	317	1.96	2.97E-3	237	270	1.97	2.89E-3	246	261	1.95	2.91E-3	735.50
9K-30	412	2478	58	189	1.51	3.20E-3	139	173	1.65	3.49E-3	148	163	1.62	3.50E-3	428.55
9K-35	518	2984	46	112	1.00	2.64E-3	82	115	1.07	2.63E-3	87	101	1.10	2.72E-3	158.73
9K-40	514	3108	57	186	0.54	3.91E-3	118	151	0.58	3.82E-3	126	140	0.57	3.69E-3	271.43
9K-45	603	3538	39	89	0.17	2.13E-3	62	91	0.34	2.18E-3	66	80	0.28	2.23E-3	121.89
9K-50	676	3808	40	92	-0.18	2.57E-3	63	90	-0.11	2.62E-3	67	81	-0.11	2.62E-3	115.30
9K-55	651	3776	59	181	0.47	4.38E-3	127	161	0.49	4.28E-3	133	148	0.47	4.19E-3	186.84
9K-60	693	4080	49	120	0.00	3.73E-3	85	118	-0.02	3.75E-3	90	104	-0.06	3.63E-3	126.44
9K-65	741	4292	55	137	0.07	2.94E-3	106	140	0.20	3.11E-3	112	126	0.20	3.10E-3	152.75
9K-70	764	4430	38	86	-0.33	2.26E-3	63	87	-0.24	2.34E-3	65	80	-0.30	2.29E-3	97.04
13K-25	371	2234	32	69	1.52	3.51E-03	47	73	1.58	3.38E-03	49	63	1.55	3.43E-03	1513.66
13K-30	452	2536	28	57	1.30	2.26E-03	38	60	1.33	2.36E-03	39	53	1.33	2.45E-03	682.30
13K-35	482	2846	32	67	0.74	3.37E-03	45	67	0.77	3.45E-03	48	62	0.78	3.44E-03	887.56
13K-40	533	3066	25	48	1.20	2.73E-03	34	54	1.30	2.73E-03	35	48	1.28	2.69E-03	533.48
13K-45	655	3768	31	64	1.19	4.05E-03	43	66	1.28	4.07E-03	46	60	1.27	4.05E-03	300.73
13K-50	618	3692	25	48	0.53	2.82E-03	33	56	0.60	3.03E-03	35	49	0.59	3.15E-03	330.53
13K-55	721	4246	25	46	0.62	3.98E-03	31	53	0.64	4.01E-03	33	47	0.65	4.00E-03	239.50
13K-60	717	4176	22	40	1.04	2.42E-03	27	47	1.07	2.46E-03	27	41	1.07	2.44E-03	157.78
13K-65	736	4258	21	38	0.81	2.11E-03	26	44	0.82	2.18E-03	26	40	0.83	2.23E-03	197.60
13K-70	843	4784	25	47	1.19	4.24E-03	32	56	1.25	4.16E-03	29	53	1.22	4.21E-03	167.57

we manually scale the objective of the each JuMP model so that the largest coefficient passed to the solver is in the order of $1.0e + 8$. This trick helps IPOPT output stable solutions efficiently. Nevertheless, we observe that the proposed algorithm tends to yield large gaps on instances with quadratic generation cost. Since we scale the objective inside the execution of IPOPT, the numerical inaccuracy will be magnified when we calculate the true generation cost; in this situation, a quadratic cost function is more sensitive than a linear one. Such numerical issues are known to be associated with penalty-type methods and deserve further investigations.

Termination of Inner and Outer Iterations: We stop the inner-level ADMM if: (1) $\|Ax^t + B\bar{x}^t + z^t\| \leq \sqrt{d}/(2500k)$ where d is the number of the coupling constraints and k is the current outer-level iteration index, **or** (2) $\|z^t - z^{t-1}\| \leq 1.0e-8$. The first condition ensures that the ADMM primal residual is under certain tolerance, which also tends to 0 as $k \to \infty$. The second condition measures the dual residual of the last block in (15); if this quantity is small, then we believe z^t has stabilized in the current inner iteration, which encourages us to terminate ADMM early and proceed to update outer-level dual variables. The outer level is terminated if the consensus mismatch satisfies $\|Ax^k + B\bar{x}^k\| \leq \sqrt{d} \times \epsilon$, where $\epsilon > 0$ is some given tolerance.

D. Numerical Performance on 9K and 13K from NESTA

The results for 9K and 13K networks are displayed in Table II, where we set $\epsilon = 2 \times 10^{-4}$. We let each core solve a single subproblem (corresponding to a subregion) in every ADMM iteration. The total numbers of tie-lines and coupling constraints in each instance are recorded in the second and third columns. For each heuristic, we report the number of outer iterations (Outer), number of inner iterations (Inner), duality gap with SOCP lower bound (Gap (%)), and max violation of the coupling at termination ($\|r\|_\infty$); the average wall clock time of the three heuristics (Avg. Time (s)) is given in the last column. The three algorithms reach the desired infeasibility tolerance in all test instances, and the max constraint violation is in the order of 10^{-3}. Overall the generation costs at termination are very close to the SOCP lower bound, indicating the algorithms converge to solutions with high quality. Moreover, we emphasize the scalability of the algorithm by pointing out that, the number of inner and outer iterations are stable across instances, even though the dimension of the coupling ranges from 2000 to near 5000. We plot the averaged computation time for the proposed algorithm combined with three heuristics in Fig. 2. The computation time drops significantly and becomes comparable to the centralized solver as the number of partitions increases: 735.50 to 97.04 seconds for 9K-bus system, and 1513.66 to 167.57 seconds for 13K-bus system. Our results validate the feasibility of using parallelization to speed up computation.

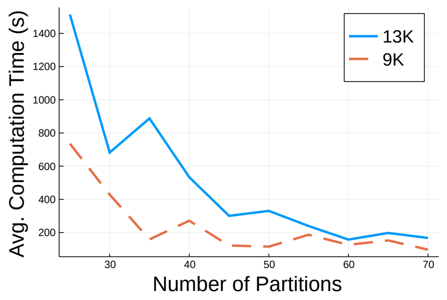

Fig. 2: Average Computation Times for 9K and 13K Networks.

E. Numerical Performance on 2K and 30K from PGLib-OPF

We present the same metrics for networks 2K ($\epsilon = 5 \times 10^{-4}$) and 30K ($\epsilon = 10^{-4}$) in Table III. We limit the computational resource to 60 cores. Different from experiments in Table II, when the network is partitioned into R subregions, each core solves $R/60$ zonal subproblems sequentially in every inner

TABLE III: Performance of the Two-level ADMM Algorithm on 2K and 30K Networks.

			TL-1				TL-2				TL-3				
Case	Tie-line	Dim	Outer	Inner	Gap (%)	$\|r\|_\infty$	Outer	Inner	Gap (%)	$\|r\|_\infty$	Outer	Inner	Gap (%)	$\|r\|_\infty$	Avg.Time (s)
2K-120	693	3412	124	548	10.73	6.18E-3	197	573	10.75	6.14E-3	194	566	10.76	6.18E-3	281.46
2K-180	857	4220	140	635	12.84	6.03E-3	229	671	12.91	6.03E-3	225	656	12.92	6.03E-3	368.28
2K-240	1059	5190	103	436	3.59	7.30E-3	141	516	2.62	7.31E-3	136	512	3.10	7.32E-3	291.46
2K-300	1591	7388	81	308	-5.94	4.78E-3	106	373	-5.90	4.81E-3	101	368	-5.83	4.77E-3	275.90
2K-360	1345	6496	93	385	3.72	6.61E-3	126	456	3.32	6.62E-3	122	456	3.79	6.60E-3	299.99
30K-2400	12761	72474	106	288	0.61	2.28E-3	124	363	1.15	2.27E-3	121	360	1.17	2.28E-3	2998.99
30K-2700	11288	66112	186	633	6.18	2.23E-3	230	812	6.29	2.24E-3	226	808	6.30	2.23E-3	6937.62
30K-3000	10748	64768	201	698	6.18	1.94E-3	250	896	6.36	1.94E-3	246	893	6.38	1.93E-3	7567.32
30K-3300	11802	69542	151	492	3.47	2.09E-3	186	631	3.60	2.06E-3	183	627	3.62	2.09E-3	5645.78
30K-3600	13695	78336	93	255	1.58	2.11E-3	113	312	1.48	2.10E-3	110	309	1.52	2.11E-3	2946.14
30K-3900	14042	80332	84	226	-0.39	2.39E-3	103	275	-0.19	2.39E-3	100	275	-0.18	2.39E-3	2694.60
30K-4200	13415	77420	122	365	1.82	2.88E-3	148	468	1.84	2.87E-3	144	462	1.85	2.88E-3	4489.54

ADMM iteration. So the computation time presented in Table III is expected to further reduce when more computation power is available. To this end, we plot the averaged subproblem time in Figure 3, i.e., average inner iteration time$\times 60/R$.

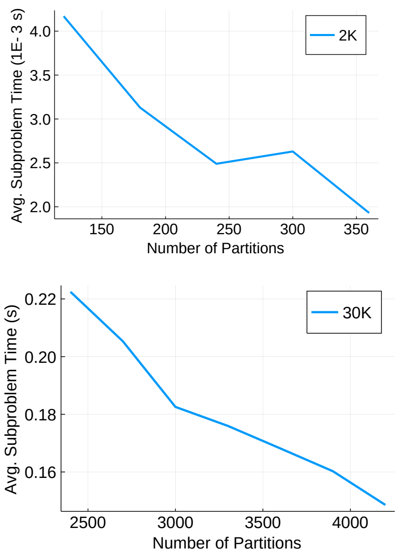

Fig. 3: Average Subproblem Times for 2K and 30K Networks.

We observe that in both Table II and Table III, the duality gap is negative for a few cases. This is because the infeasibility of coupling constrains in ℓ_2 norm is still relatively large, even though the max violation is already in the order of 10^{-3}. As an illustration, we plot the ℓ_2 and ℓ_∞ norm of the infeasibility $r = Ax + B\bar{x}$ over time for 2K-300 in Figure 4. We note that the numbers of coupling constraints reported in column “Dim” of Table III are indeed very large, and this is why we choose a smaller tolerance $\epsilon = 10^{-4}$ for 30K. This phenomenon suggests that in practice one may need to further decrease ϵ, or let the algorithm run longer to refine local solutions when the network is heavily partitioned.

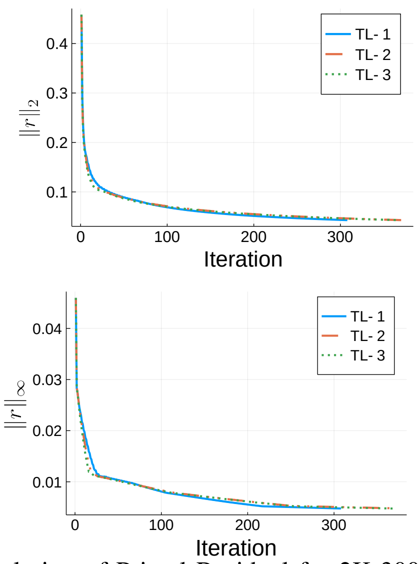

Fig. 4: Evolution of Primal Residual for 2K-300 Network.

F. Comparison with One-level ADMM Variants

To further illustrate the advantage of the proposed two-level ADMM algorithm, we implement two state-of-the-art one-level ADMM variants and display their results in Table IV. All implementation issues mentioned above are considered; however, for all cases, both one-level ADMM algorithms fail to achieve the desired infeasibility tolerance within 1000 iterations. For the modified ADMM [11], the penalty parameter quickly reaches the upper bound 10^{24}, which indicates their assumption for convergence is not satisfied, and such large penalty leads to solutions with high generation costs. Jiang et al. [25] realize that ADMM can be used to solve the relaxed problem (15) with $\lambda^k = 0$ and large enough constant β. When $\beta = \mathcal{O}(1/\epsilon^2)$, their proposed ADMM variant is guaranteed to find an ϵ-stationary solution of the original problem. Using parameters suggested in their analysis, we observe that the true infeasibility $\|Ax^t + B\bar{x}^t\|$ decreases very slowly, though the three-block residual $\|Ax^t + B\bar{x}^t + z^t\|$ converges to near 0 fast. This indicates the need for a more careful tuning of the penalty β, but there is no principled way to select all hyperparameters in their algorithm for OPF instances.

Finally, we note that the proposed two-level algorithm is not intended to replace centralized solvers or other distributed algorithms, but rather aims to serve as a general algorithmic framework with theoretically supported convergence guarantees, upon which other methods could be applied to further accelerate convergence of distributed computation in various practical applications.

TABLE IV: Comparison with One-level ADMM Variants.

	Modified ADMM [11]		ADMM-g [25]		Proposed TL-1	
Case	Gap(%)	Time (s)	Gap(%)	Time (s)	Gap(%)	Time (s)
13K-25	88.98	2869.48	-6.14	1946.21	1.52	1788.62
13K-30	90.64	2507.26	-6.21	1874.13	1.30	680.16
13K-35	91.86	2034.09	-6.97	1510.31	0.74	752.41
13K-40	92.79	2168.87	-8.09	1555.16	1.20	430.68
13K-45	93.54	1428.20	-8.06	1436.43	1.19	306.79
13K-50	94.15	1463.57	-8.44	1405.47	0.53	374.15
13K-55	94.65	1521.64	-12.59	1370.72	0.62	204.50
13K-60	95.09	1459.83	-8.26	1208.48	1.04	157.32
13K-65	95.44	1910.87	-9.16	1279.62	0.81	207.58
13K-70	95.75	1821.54	-12.20	1340.73	1.19	160.20

VI. Conclusion

In this paper we propose a two-level distributed ADMM framework for solving AC OPF. The proposed algorithm is proven to have guaranteed global convergence with an iteration complexity upper bound, and can be further accelerated by suitable heuristics. Promising numerical results over some large-scale test cases show that the proposed algorithm provides a new, robust, distributed, and convergence-guaranteed algorithmic framework for solving real-world sized AC OPF problems. Future directions include extending the two-level ADMM framework to security-constrained OPF problems, and combining with other methods, such as the component-based decomposition and adaptive penalty parameter scheme in [13]. Also further experimenting with massive parallelization and studying the impact of communication would be very interesting.

Acknowledgement The authors are grateful for support from the National Science Foundation [Grant ECCS-1751747] and the ARPA-E [Grant DE-AR0001089].

References

[1] D. Bienstock and A. Verma, "Strong NP-hardness of AC power flows feasibility," *arXiv preprint arXiv:1512.07315*, 2015.

[2] K. Lehmann, A. Grastien, and P. Van Hentenryck, "AC-feasibility on tree networks is NP-hard," *IEEE Transactions on Power Systems*, vol. 31, no. 1, pp. 798–801, 2016.

[3] R. A. Jabr, A. H. Coonick, and B. J. Cory, "A primal-dual interior point method for optimal power flow dispatching," *IEEE Transactions on Power Systems*, vol. 17, no. 3, pp. 654–662, 2002.

[4] R. D. Zimmerman, C. E. Murillo-Sánchez, R. J. Thomas *et al.*, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on power systems*, vol. 26, no. 1, pp. 12–19, 2011.

[5] X. Bai, H. Wei, K. Fujisawa, and Y. Wang, "Semidefinite programming for optimal power flow problems," *International Journal of Electrical Power & Energy Systems*, vol. 30, no. 6-7, pp. 383–392, 2008.

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

[7] R. A. Jabr, "Radial distribution load flow using conic programming," *IEEE transactions on power systems*, vol. 21, no. 3, pp. 1458–1459, 2006.

[8] B. Kocuk, S. S. Dey, and X. A. Sun, "Strong SOCP relaxations for the optimal power flow problem," *Operations Research*, vol. 64, no. 6, pp. 1177–1196, 2016.

[9] A. X. Sun, D. T. Phan, and S. Ghosh, "Fully decentralized AC optimal power flow algorithms," in *Power and Energy Society General Meeting (PES), 2013 IEEE*. IEEE, 2013, pp. 1–5.

[10] T. Erseghe, "Distributed optimal power flow using ADMM," *IEEE Transactions on Power Systems*, vol. 29, no. 5, pp. 2370–2380, 2014.

[11] ——, "A distributed approach to the OPF problem," *EURASIP Journal on Advances in Signal Processing*, vol. 2015, no. 1, p. 45, 2015.

[12] M. J. Feizollahi, S. Ahmed, and A. Sun, "Exact augmented Lagrangian duality for mixed integer linear programming," *Mathematical Programming*, vol. 161, no. 1-2, pp. 365–387, 2017.

[13] S. Mhanna, G. Verbic, and A. C. Chapman, "Adaptive ADMM for distributed AC optimal power flow," *IEEE Transactions on Power Systems*, vol. 34, no. 3, pp. 2025–2035, May 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8573893/

[14] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, "A survey of distributed optimization and control algorithms for electric power systems," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2941–2962, 2017.

[15] B. Houska, J. Frasch, and M. Diehl, "An augmented Lagrangian based algorithm for distributed nonconvex optimization," *SIAM Journal on Optimization*, vol. 26, no. 2, pp. 1101–1127, 2016.

[16] A. Engelmann, Y. Jiang, T. Mühlpfordt, B. Houska, and T. Faulwasser, "Toward distributed OPF using ALADIN," *IEEE Transactions on Power Systems*, vol. 34, no. 1, pp. 584–594, 2018.

[17] C. Coffrin, D. Gordon, and P. Scott, "Nesta, the nicta energy system test case archive," *arXiv preprint arXiv:1411.0359*, 2014.

[18] S. Babaeinejadsarookolaee, A. Birchfield, R. D. Christie, C. Coffrin, C. DeMarco, R. Diao, M. Ferris, S. Fliscounakis, S. Greene, R. Huang *et al.*, "The power grid library for benchmarking AC optimal power flow algorithms," *arXiv preprint arXiv:1908.02788*, 2019.

[19] R. Glowinski and A. Marroco, "Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires," *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, vol. 9, no. R2, pp. 41–76, 1975.

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

[21] M. R. Hestenes, "Multiplier and gradient methods," *Journal of optimization theory and applications*, vol. 4, no. 5, pp. 303–320, 1969.

[22] M. J. Powell, *A method for non-linear constraints in minimization problems*. UKAEA, 1967.

[23] F. Wang, Z. Xu, and H.-K. Xu, "Convergence of bregman alternating direction method with multipliers for nonconvex composite problems," *arXiv preprint arXiv:1410.8625*, 2014.

[24] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, pp. 1–35, 2015.

[25] B. Jiang, T. Lin, S. Ma, and S. Zhang, "Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis," *Computational Optimization and Applications*, vol. 72, no. 1, pp. 115–157, 2019.

[26] K. Sun and X. A. Sun, "A two-level distributed algorithm for general constrained non-convex optimization with global convergence," *arXiv preprint arXiv:1902.07654*, 2019.

[27] B. H. Kim and R. Baldick, "Coarse-grained distributed optimal power flow," *IEEE Transactions on Power Systems*, vol. 12, no. 2, pp. 932–939, 1997.

[28] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*. Springer Science & Business Media, 2009, vol. 317.

[29] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[30] Z.-Q. Luo, J.-S. Pang, D. Ralph, and S.-Q. Wu, "Exact penalization and stationarity conditions of mathematical programs with equilibrium constraints," *Mathematical Programming*, vol. 75, no. 1, pp. 19–76, 1996.

[31] I. Dunning, J. Huchette, and M. Lubin, "JuMP: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[32] G. Karypis and V. Kumar, "Multilevel k-way partitioning scheme for irregular graphs," *Journal of Parallel and Distributed computing*, vol. 48, no. 1, pp. 96–129, 1998.

Appendix

A. Proof of Theorem 1

Proof. The claims under condition (21) are proved in Theorem 1-2 of [26]. We shall prove the claims under condition (22).

1) The first claim follows from the compactness of $\mathcal{X}_r$'s and $\bar{\mathcal{X}}$ and the fact that $\|Ax^k + B\bar{x}^k + z^k\| \to 0$.
2) First we assume the sequence $\{\beta^k\}_k$ stays finite. Then there exists $K > 0$ such that $\eta_k \geq \|z^k\|$ for all $k \geq K$, which follows $\|z^k\| \to 0$. Next assume $\beta^k \to +\infty$, and assume without loss of generality that $(x^k, \bar{x}^k, z^k) \to (x^*, \bar{x}^*, z^*)$. If the first case in (22) is executed infinitely many times, we have $\|z^k\| \to 0$; otherwise we must have $\|\lambda^k\|$ stays constant for all sufficiently large k. Define $\tilde{\lambda}^k := \lambda^k + \beta^k z^k = -y^k$. If $\{\tilde{\lambda}^k\}$ has a bounded subsequence, then we have $\|z^k\| \to 0$ as β^k converges to infinity. In all previous cases, we have $\|Ax^k + B\bar{x}^k\| \leq \|Ax^k + B\bar{x}^k + z^k\| + \|z^k\| \leq \epsilon_k + \|z^k\| \to 0$, and thus $\|Ax^* + B\bar{x}^*\| = 0$. In the last case we have $\beta^k \to \infty$, $\|\lambda^k\|$ stays bounded, and $\|\tilde{\lambda}^k\| \to \infty$. By the definition of $\tilde{\lambda}^k$, we can see $y^k/\beta^k \to -z^* = Ax^* + B\bar{x}^*$. At termination of ADMM, we have
$$d_1^k \in \nabla c(x^k) + A^\top y^k + N_{\mathcal{X}}(x^k) \tag{25a}$$
$$d_2^k \in B^\top y^k + N_{\bar{\mathcal{X}}}(\bar{x}^k) \tag{25b}$$
$$d_3^k = Ax^k + B\bar{x}^k + z^k, \tag{25c}$$
where $\max\{\|d_1^k\|, \|d_2^k\|, \|d_3^k\|\} \leq \epsilon_k \to 0$. By the closeness of normal cone, dividing vectors in (25a)-(25b) by β^k, and then taking limit, we have $0 \in A^\top(Ax^* + B\bar{x}^*) + N_{\mathcal{X}}(x^*)$ and $0 \in B^\top(Ax^* + B\bar{x}^*) + N_{\bar{\mathcal{X}}}(\bar{x}^*)$, which imply $(x^*, \bar{x}^*)$ is stationary for (23).
3) Using the same case analysis as in part 2), we can see $\|z^k\| \to 0$ (along the subsequence converging to z^*). Thus taking limit on (25) completes the proof.

□

B. Proof of Theorem 2

Proof. We use T_k to denote an upper bound of the number of inner iterations of the k-th ADMM, which produces an ϵ-stationary solution point of (15) (see Definition 2) . By Lemma 3 of [26], upon termination of ADMM, we can find $(x^k, \bar{x}^k, z^k, y^k)$ and corresponding (d_1^k, d_2^k, d_3^k) such that
$$\begin{aligned}\|d_1^k\| =& \|\rho^k A^\top (B\bar{x}^{t-1} + z^{t-1} - B\bar{x}^t - z^t)\| \\ \leq& \rho^k \|A\| \|B\bar{x}^{t-1} + z^{t-1} - B\bar{x}^t - z^t\|, \end{aligned} \tag{26a}$$
$$\|d_2^k\| = \|\rho B^\top (z^t - z^{t-1})\| \leq \rho^k \|B\| \|z^{t-1} - z^t\|, \tag{26b}$$
$$\|d_3^k\| = \frac{1}{2}\|z^{t-1} - z^t\|, \tag{26c}$$
where $1 \leq t \leq T_k$ is some index during ADMM satisfying
$$\begin{aligned}&\|B\bar{x}^{t-1} - B\bar{x}^t\| + \|z^{t-1} - z^t\| \\ \leq& \sqrt{2}(\|B\bar{x}^{t-1} - B\bar{x}^t\|^2 + \|z^{t-1} - z^t\|^2)^{1/2} \\ \leq& \sqrt{2}\left(\frac{2(\overline{L} - \underline{L})}{\beta^k T_k}\right)^{1/2} = 2\left(\frac{\overline{L} - \underline{L}}{\beta^k T_k}\right)^{1/2}. \end{aligned} \tag{27}$$
Recall $\rho^k = 2\beta^k = 2\beta^0 c^k \geq 2\beta^0$. So (26) and (27) give
$$\max\{\|d_1^k\|, \|d_2^k\|, \|d_3^k\|\} \leq \rho^k \tau \left(\frac{\overline{L} - \underline{L}}{\beta^k T_k}\right)^{1/2}, \tag{28}$$
where $\tau = \max\{2\|A\|, 2\|B\|, 1/(2\beta^0)\}$. It is sufficient to find a T_k with
$$\frac{4\beta^0(\overline{L} - \underline{L})\tau^2 c^k}{\epsilon^2} \leq T_k \leq \frac{4\beta^0(\overline{L} - \underline{L})\tau^2 c^k}{\epsilon^2} + 1, \tag{29}$$
in order to get $\max\{\|d_1^k\|, \|d_2^k\|, \|d_3^k\|\} \leq \epsilon$. Let K denote the number of outer-level ALM iterations. Then the total number of inner iterations is bounded by
$$\sum_{k=1}^K T_k \leq \left\lceil \left(\frac{4\beta^0(\overline{L} - \underline{L})\tau^2 c}{c-1}\right)\left(\frac{c^K - 1}{\epsilon^2}\right)\right\rceil + K. \tag{30}$$

Now it remains to bound the number of outer-level iterations K. Notice that the dual residuals $\|d_1^K\|$ and $\|d_2^K\|$ are controlled by ϵ at the end of the inner-level ADMM, so it is sufficient to ensure the primal residual $\|Ax^K + B\bar{x}^K\| \leq \epsilon$. By Theorem 3 of [26],
$$\|Ax^K + Bx^K\|^2 \leq \frac{2(\overline{L} - c(x^K) + \langle \lambda^k, Ax^K + Bx^K \rangle)}{\beta^K},$$
and the claimed K_1 is sufficient to ensure $\|Ax^{K_1} + Bx^{K_1}\| \leq \epsilon$. Next we show the claimed bound on K_2. If $\|\lambda^k + \beta^k z^k\| \leq \Lambda$ for all outer index k, then we have
$$\|z^K\| = \frac{\|\lambda^K + \beta^K z^K - \lambda^K\|}{\beta^K} \leq \frac{\Lambda + M}{\beta^0 c^K}. \tag{31}$$
By (26) and (27) , we have
$$\|Ax^K + B\bar{x}^K + z^K\| = \|d_3^K\| \leq \left(\frac{\overline{L} - \underline{L}}{\beta^K T_K}\right)^{1/2} \leq \frac{\epsilon}{2\beta^0 c^K \tau},$$
where the last inequality is due to the lower bound of T_K in (29). It is straightforward to verify that the claimed K_2 ensures
$$\begin{aligned}\|Ax^{K_2} + B\bar{x}^{K_2}\| \leq& \|Ax^{K_2} + B\bar{x}^{K_2} + z^{K_2}\| + \|z^{K_2}\| \\ \leq& \epsilon/2 + \epsilon/2 = \epsilon.\end{aligned}$$
This completes the proof. □