Optimal Distributed Control of Reactive Power via the Alternating Direction Method of Multipliers

Petr Šulc, Scott Backhaus, Michael Chertkov, Member, IEEE

Abstract—We formulate the control of reactive power generation by photovoltaic inverters in a power distribution circuit as a constrained optimization that aims to minimize reactive power losses subject to finite inverter capacity and upper and lower voltage limits at all nodes in the circuit. When voltage variations along the circuit are small and losses of both real and reactive powers are small compared to the respective flows, the resulting optimization problem is convex. Moreover, the cost function is separable enabling a distributed, on-line implementation with node-local computations using only local measurements augmented with limited information from the neighboring nodes communicated over cyber channels. Such an approach lies between the fully centralized and local policy approaches previously considered. We explore protocols based on the dual ascent method and on the Alternating Direction Method of Multipliers (ADMM) and find that the ADMM protocol performs significantly better.

Index Terms—photovoltaic power generation, reactive power control, power flow, ADMM, dual ascent method, distributed algorithms, distributed control.

I. INTRODUCTION

The generation or consumption of reactive power by inverters has been explored by several researchers as a way to control voltage fluctuations in distribution circuits with a high penetration of distributed photovoltaic (PV) generation. See [1] and references therein for an overview. These approaches have tended to fall near two extremes. One extreme considers centralized optimization [2] where computations are done by a single central authority which is assumed to have full observability of the system. Such a system requires two-way communications between the central authority and at least all of controlled inverters. The rate of these communications should be sufficient to respond to the fastest expected fluctuations of solar irradiance. At the other extreme are local policy-based methods that require no communications at all, except for perhaps between devices at a single node. These inherently suboptimal methods rely only on node-local measurements as inputs to a policy that converts the measurements into a control action. Such policies have been based on heuristics and physical reasoning [4] and on Monte Carlo-like approaches [4] that use centralized optimization in off-line computations to find strong correlations between the local measurements and the optimal local control actions.

In [4], it was found that a significant source of the suboptimal behavior of local policy-based methods was saturation of inverters. In this case, the local policy results in a desired control action that, when combined with the local power flow conditions, is beyond the capability of the inverter. In [4], this generally occurred when the real power injection by the inverter was approaching the apparent power capacity leaving little room for reactive power generation or consumption. Instead of achieving the desired reactive power injection, the inverter saturated at its apparent power capacity. In a centralized approach, optimality is restored because the full observability of the central authority makes it aware of the saturation allowing it to compensate with extra reactive power generation (or consumption) from nearby nodes. Such a response suggests that nearest-neighbor communications may be used to restore optimality in capacity-limited or otherwise constrained systems. There are other reasons to expect that limited communications may provide significant advantages over policy-based control without the larger overhead of centralized communications. Primary among these is the ability to adapt to system configurations that were unforeseen in the development of a policy based either on heuristics or sampling methods.

In this manuscript, we explore distributed approaches to control reactive power from PV inverters in distribution circuits and show how to restore optimality and adaptability through an iterative message-passing algorithm. Although we invoke limited communications, we continue to rely solely on local computations which depend only on local measurements and the most current data communicated from the nearest-neighbor nodes. The suggested cyber-physical control scheme is decentralized but optimal.

The results in this manuscript are based on the observation that the convex optimization formulation of [1, 2, 3] is separable in the key optimization variables, i.e., the node voltages and the power flows along the circuit. Separability suggests application of modern methods of distributed computations such as the dual ascent method and the Alternating Direction Method of Multipliers (ADMM) [5, 6]. The ADMM algorithm converges significantly faster than the dual ascent algorithm, a property one expects from the general arguments expressed in [5]. Furthermore, the dual ascent algorithm for a non-differentiable dual function does not always converge to the optimal solution, but can converge to its neighborhood [7, 8]. Faster convergence of ADMM is largely confirmed in our experiments conducted over seven different distribution circuit
configurations with different numbers of nodes and varying photovoltaic penetration and load profiles. In each experiment, we minimize the total loss of real power while constraining all node voltages to be within nominal operational bounds.

The dual ascent method has been considered by [8, 10, 11], where a slightly more general model of radial power flows was considered. Dual decomposition distributed algorithm with gradient ascent for voltage regulation was also proposed in [13], where it was shown to solve a convex relaxation of power flow equations. We further note that an optimization algorithm [13], where it was shown to solve a convex relaxation of power flow equations. We further note that an optimization algorithm for optimal load control for frequency regulation has been recently proposed [14], where it was shown that a frequency-based load control together with the system dynamics and power flows act as a decentralized primal-dual algorithm that solves the global optimization problem. The dual ascent and ADMM-based algorithm was also used to solve a semi-definite programming relaxation of optimal power flow problem, where it was shown that a frequency-based load control together with the system dynamics and power flows act as a decentralized primal-dual algorithm that solves the global optimization problem. The dual ascent and ADMM-based algorithm was also used to solve a semi-definite relaxation of power flow problem for optimization of real and reactive PV generation in a radial network was also used in [15], where it yields an exact solution of the original problem in a single-phase radial network. After the initial submission of our manuscript to the arxiv, other methods of solving convex relaxation of power flow equations via ADMM were proposed [17, 18, 19].

The material in the remainder of this manuscript is organized as follows. Power flows in a distribution circuit and control of inverters as a global optimization are reviewed in Section II and Section III, respectively. Algorithms for distributed control based on nearest neighbor communications are described in Section III-A and III-B. The algorithms are tested and compared in Section IV. Section V presents our conclusions and a brief discussion of the path forward.

II. DISTRIBUTED FLOW FORMULATION

The flow of electric power in the quasi-static approximation is governed by Kirchoff’s laws. The DistFlow equations [12, 20, 21] are these laws restated in terms of power flows and applied to radial or tree-like distribution circuit with a discrete set of loads. For the radial case, the DistFlow equations are

\[ \begin{align*}
    P_{j+1} &= P_j - r_j \frac{P_j^2 + Q_j^2}{V_j^2} - p_{j+1}, \quad (1a) \\
    Q_{j+1} &= Q_j - x_j \frac{P_j^2 + Q_j^2}{V_j^2} - q_{j+1}, \quad (1b) \\
    V_{j+1}^2 &= V_j^2 - 2(r_j P_j + x_j Q_j) + (r_j^2 + x_j^2) \frac{P_j^2 + Q_j^2}{V_j^2}, \quad (1c)
\end{align*} \]

where \( P_j + iQ_j \) is the complex power flowing away from node \( j \) toward node \( j+1 \), \( V_j \) is the voltage at node \( j \), \( r_j \) and \( x_j \) are the complex impedance of the link between node \( j \) and \( j+1 \), and \( p_j + iq_j \) is the complex power extracted at the node \( j \).

Both power generation to the inverters’ apparent power capacity \( s_j \):
Under normal operations, the changes in voltage from node to node are small compared to the voltages and the loss of real and reactive power are small compared to the power flows themselves. In this limit, Eqs. (2) can be restated within the LinDistFlow approximation, i.e.,

\[
\begin{align*}
& \min_{q^{(g)}, P, Q} \sum_{j=0}^{n-1} r_j P_j^2 + Q_j^2 \frac{V_0^2}{V_0^2}, \\
& \text{s.t.} \quad \forall j = 0, \ldots, n - 1: \\
& \quad P_{j+1} = P_j - p_j^{(c)} + p_{j+1}^{(c)}, \quad P_n = 0, \quad (3a) \\
& \quad Q_{j+1} = Q_j - q_j^{(c)} + q_{j+1}^{(c)}, \quad Q_n = 0, \quad (3b) \\
& \quad U_{j+1} = U_j - 2(r_j P_j + x_j Q_j), \quad (3d) \\
& \quad \forall j = 1, \ldots, n: \\
& \quad V_0^2 (\epsilon^2 - 2\epsilon) \leq U_j \leq V_0^2 (\epsilon^2 + 2\epsilon), \quad (3e) \\
& \quad \left| q_j^{(g)} \right| \leq \delta_j, \quad (3f)
\end{align*}
\]

where \(U_j = V_j^2 - V_0^2\) and \(\delta_j = \sqrt{s_j^2 - (p_j^{(g)})^2}\). We have assumed \(V_0^2 \approx V_j^2\) in (3a).

Simulations in [2, 3] suggest that the LinDistFlow are well justified for a wide range of distribution circuits. This observation is powerful because the LinDistFlow formulation of Eqs. (3) is convex (a quadratic objective function with linear constraints). Convexity implies that this optimization can be solved efficiently provided each node can communicate with a central authority which performs the computations and distributes the optimal values of \(q_j^{(g)}\) to all nodes [2, 3]. In the remainder of this work, we will focus on developing a decentralized optimization algorithm, which solves Eqs. (3) only by passing messages between nearest neighbors on the network.

We note that the \(P_j\) are determined by solving Eq. (5b) for given \(p_j^{(c)}\) and \(p_j^{(g)}\), and the \(P_j\) in Eqs. (3d) can be treated as constants and one can formulate the problem as an optimization over the \(Q_j\) by combining Eqs. (3c) and (3f):

\[
\begin{align*}
& \left| Q_j - Q_{j-1} + q_j^{(c)} \right| \leq \delta_j, \quad Q_n = 0. \quad (3g)
\end{align*}
\]

The actual control outputs are the \(q_j^{(g)}\), but these can be inferred from Eq. (3c), once the optimal solution is stated in terms of \(Q_j\).

In the following, we present an ADMM-based distributed algorithm for the solution of the LinDistFlow problem. We further present a dual-ascent distributed algorithm that solves a simplified LinDistFlow problem where the voltage constraints are omitted and compare its performance with the ADMM algorithm (with and without voltage constraints).

A. ADMM consensus distributed algorithm

We adapt a consensus version of the ADMM algorithm to our problem. A general discussion of the method and proof of convergence is described in [5]. The consensus version assumes that each of the nodes in the network has its own local objective function and a local set of constraints which act on a global variable shared between all the nodes. Each node solves a local optimization problem for respective local copies of the global variables. The local optimization problem consists in finding the optimum for the local copies of the variables, subject to the condition that all local copies are equal to the global variable. The problem is solved iteratively, with all local copies eventually converging to the global optimal value.

For the problem described by Eqs. (3), each node \(j\) will keep the local copies of the \(Q_j, Q_{j-1}, U_j\) and \(U_{j-1}\) variables, which we will denote \(Q_+^j, Q_{-}^j, U_+^j\) and \(U_{-}^j\) respectively. We note that the algorithm assumes that one can measure power flows \(P_j\) between nodes which are treated as auxiliary constant parameters of the algorithm. The optimization problem (3) formulated as a consensus problem becomes:

\[
\begin{align*}
& \min Q \sum_{j=1}^n r_{j-1} (Q_{j-1}^-)^2, \\
& \text{s.t.} \quad \forall j = 1, \ldots, n: \\
& \quad Q_{j+}^j - Q_{j-}^j + q_j^{(c)} - \delta_j \leq 0, \quad (4b) \\
& \quad -Q_{j+}^j + Q_{j-}^j - q_j^{(c)} + \delta_j \leq 0, \quad (4c) \\
& \quad Q_{j+}^j = Q_{j-}^j = Q_{j-1}^j = Q_{j-2}^j, \quad (4d) \\
& \quad V_0^2 (\epsilon^2 - 2\epsilon) \leq U_{j+}^j \leq V_0^2 (\epsilon^2 + 2\epsilon), \quad (4e) \\
& \quad U_{j+}^j = U_{j-}^j = 2(r_{j-1} P_{j-1} + x_{j-1} Q_{j-1}^-), \quad (4f) \\
& \quad U_{j+}^j = U_{j-}^j = 0, \quad (4g)
\end{align*}
\]

The conditions (4d, 4g) ensure that all local copies of the variables are equal to the global variables \(Q_j\) and \(U_j\), and hence the optimization problem (4) is equivalent to (3). We solve (4) using distributed ADMM method [5, 6], for which the augmented Lagrangian is

\[
\mathcal{L}_{\text{ADMM}} = \sum_{j=1}^n \mathcal{L}_{j,\text{ADMM}}, \quad (5a)
\]

where

\[
\mathcal{L}_{j,\text{ADMM}} = \frac{r_{j-1} (Q_{j-1}^-)^2}{V_0^2} + \frac{\rho}{2} (Q_{j+}^j - Q_{j-}^j)^2 + \frac{\rho}{2} (Q_{j-}^- - Q_{j-1}^j)^2 + \frac{\rho}{2} (U_{j+}^j - U_{j-}^j)^2 + \frac{\rho}{2} (U_{j-}^- + U_{j-1}^-)^2 + \lambda_{Q+}^j (Q_{j+}^j - Q_j) + \lambda_{Q-}^j (Q_{j-}^- - Q_{j-1}^j) + \lambda_{U+}^j (U_{j+}^j - U_j) + \lambda_{U-}^j (U_{j-}^- - U_{j-1}^-). \quad (5b)
\]

The quadratic terms in the objective function with \(\rho/2\) prefactor represent penalties for the local variables being different from the global variables. These terms do not change the optimal value, as the constraints (4d, 4g) require that the local and global variables are equal at the optimum. The dual variables associated with (4d, 4g) are \(\lambda_{Q+}^j, \lambda_{Q-}^j, \lambda_{U+}^j\) and \(\lambda_{U-}^j\). Note that we do not include constraints (4c) in the Lagrangian, as the algorithm will be minimizing \(\mathcal{L}_{\text{ADMM}}\) in such a way that \(Q^+\) and \(Q^-\) will always stay within the feasible set, i.e. satisfy (4b, 4c).

The ADMM distributed consensus algorithm is an iterative algorithm where the \(k+1\)-th iteration starts with values \(Q_j(k), Q_{j-1}(k), Q_{j+}^j(k), Q_{j-}^- (k), \lambda_{Q+}^j(k), \lambda_{Q-}^j(k), U_j(k), U_{j-1}(k), \)
$U^+(k), U^-(k), \lambda^+(k), \lambda^-(k)$ for each node $j$. One iteration of the algorithm consists of the following steps:

1) **Minimization step.** For each node $j$, the following optimization problem is solved

$$\min_{Q_j, Q_j^+, U_j^+, U_j^-} \mathcal{L}_{j}^{\text{ADMM}}$$

s.t. Eqs. (4b) (4c) (4e) (4f)

This minimization step is a convex optimization problem with quadratic objective function of four local variables $Q_j, Q_j^+, U_j^+, U_j^-$ with linear constraints and can be solved analytically by evaluating the corresponding Karush-Kuhn-Tucker conditions [22]. However, the expressions are bulky and we do not present them here for the sake of brevity. Each node $j$ can perform the minimization step independently, as the optimization is carried over the local variables at node $j$. The solutions to the local minimization problem in $k$-th iteration of the ADMM distributed consensus algorithm are denoted as $Q_j(k+1), Q_j^+(k+1), U_j^+(k+1), U_j^-(k+1)$.

2) **Averaging step.** This step updates the global (shared) variables $Q$ and $U$. The update rules for each are the following:

$$\forall j = 1, \ldots, n - 1 :$$

$$Q_j(k+1) = \frac{1}{2} (Q_j^+(k+1) + Q_j^+(k+1))$$

$$U_j(k+1) = \frac{1}{2} (U_j^+(k+1) + U_j^+(k+1))$$

$$Q_n(k+1) = 0, Q_0(k+1) = Q_1^-(k+1), U_n = U_n^+$$

This step requires communication between nearest neighbors, as they need to exchange their local variables in order for each node $j$ to calculate the new value for $Q_j$ and $Q_{j-1}$ which is the average of the respective local copies of neighboring nodes.

3) **Lagrange multipliers update step.** The Lagrange multipliers, which are also stored by each node locally, are updated according to the following rules for each node $j$:

$$\lambda_j^{Q^+}(k+1) = \lambda_j^{Q^+}(k) + \rho \left( Q_j^+(k+1) - Q_j(k+1) \right)$$

$$\lambda_j^{Q^-}(k+1) = \lambda_j^{Q^-}(k) + \rho \left( Q_j^-(k+1) - Q_{j-1}(k+1) \right)$$

$$\lambda_j^{U^+}(k+1) = \lambda_j^{U^+}(k) + \rho \left( U_j^+(k+1) - U_j(k+1) \right)$$

$$\lambda_j^{U^-}(k+1) = \lambda_j^{U^-}(k) + \rho \left( U_j^-(k+1) - U_j(k+1) \right)$$

All variables involved in this step have been calculated and communicated in the previous step, which means that the Lagrange multipliers are updated locally at each node $j$.

The ADMM algorithm requires synchronized communication between the neighboring nodes where local variables $(Q_j^+, Q_j^-, U_j^+ and U_j^-)$ are communicated between nearest neighbors. These local variables can be interpreted as ‘beliefs’ of node $j$ about which reactive power should be flowing in and out of the node and what should be the voltage magnitudes. The consensus algorithm leads to convergence of these local variables between the neighboring nodes, thus finding a global optimal solution. Once the algorithm converges, the local variables will actually correspond to an optimized feasible solution $Q_j$ and $U_j$ of problem (4). The actual values of reactive power injected by inverters can be calculated by each node from its local variables as

$$q_j^{(g)} = Q_j^+ - Q_j^- + q_j^{(c)}$$

The above solution for $q_j^{(g)}$ is guaranteed to be within the allowed bounds given by the $\bar{s}_j$, as the local variables always satisfy the conditions (4b) (4e).

To test the performance of the ADMM algorithm, we compare its convergence with a dual ascent algorithm, which we derive for a simplified LinDistFlow problem without voltage constraints in Section IV. As we will show in Section V, the ADMM distributed consensus algorithm converges faster than the dual ascent algorithm, which is known to be its general main advantage [5].

B. Dual ascent algorithm for distributed control of the inverters with no voltage constraints

We now formulate a dual ascent approach to solving the simplified optimization problem (3) without considering the voltage constraints (3d) (3e). The Lagrangian becomes

$$\mathcal{L} (Q, \zeta^+, \zeta^-) = \sum_{j=0}^{n-1} \left[ r_j Q_j^2 \right] + \zeta^+ (Q_{j+1} - Q_j - \bar{s}_j + q_j^{(c)}) \right] + \zeta^- (-Q_{j+1} + Q_j - \bar{s}_j - q_j^{(c)})$$

The dual ascent algorithm consists of the following steps:

1) Minimize $\mathcal{L} (Q, \zeta^+, \zeta^-)$ over $Q$ for given $\zeta^+, \zeta^-$, which leads to the following update rule in $k$-th iteration

$$Q_j(k+1) = \frac{V_j^2}{2r_j} (\zeta_j^+(k) - \zeta_j^+(k) + \zeta_j^-(k) - \zeta_j^-(k)) .$$

(9a)

2) Update dual variables according to

$$\zeta_j^+(k+1) = \max \left( 0, \zeta_j^+(k) + \alpha \Delta_j^+ \right)$$

$$\zeta_j^-(k+1) = \max \left( 0, \zeta_j^-(k) + \alpha \Delta_j^- \right)$$

where

$$\Delta_j^+ = Q_{j+1}(k+1) - Q_j(k+1) + q_j^{(c)} - \bar{s}_j - q_j^{(c)}$$

$$\Delta_j^- = -Q_{j+1}(k+1) + Q_j(k+1) - q_j^{(c)} - \bar{s}_j + q_j^{(c)}$$

This scheme allows parallel implementation, where each node $j$ receives values of $\zeta_j^+(k)$, $\zeta_j^-(k)$ from its right neighbor $j+1$ and sends values of $\zeta_j^+(k)$, $\zeta_j^-(k)$ to its left neighbor $j-1$. Node $j$ then calculates $Q_j(k+1)$ using those variables and then sends the result to neighbor $j+1$, while receiving $Q_{j-1}(k+1)$ from neighbor $j-1$. The communicated values are then used by node $j$ to calculate $\zeta_j^+(k+1)$ and $\zeta_j^-(k+1)$. An advantage of the dual ascent algorithm is that it requires the nodes to perform trivial algebraic operations (which are simpler than the solution of Eq. (4)) and synchronously...
communicate their local variables $Q$, $\zeta^+$, and $\zeta^-$ to their neighbors. However, based on discussion in [5], the dual ascent method is expected to require a large number of iterations to converge. The speed of convergence is controlled by parameter $\alpha$, but the range of feasible $\alpha$ is limited—choosing $\alpha$ which is too large results in a failure to converge, while $\alpha$ chosen too small translates into a slow convergence.

We provide comparison of the dual ascent algorithm with the ADMM algorithm in Section IV-C. Since the proposed dual ascent algorithm does not consider voltage constraints, we will also consider a version of the ADMM algorithm without the voltage constraints (referred to as ADMM-noV), which can be straightforwardly obtained from the ADMM algorithm by excluding voltage variables $U_j, U_j^+$, $U_j^-$ and removing constraints (4e) (4f) (4g).

IV. EXPERIMENTS WITH DISTRIBUTED GLOBAL OPTIMIZATION ON DIFFERENT DISTRIBUTION CIRCUIT CASES

We explore the performance of our distributed optimization algorithms via simulations on a range of distribution circuit cases. We first introduce the considered feeder line configuration cases and then compare the performance of the dual ascent, ADMM and ADMM-noV algorithms on these cases. The global optimization results from these algorithms are compared with the sub-optimal local optimization scheme proposed in [2], where each node only uses its local information about $q_j^{\text{(c)}}$ to set its $q_j^{\text{(g)}}$. Finally, we compare losses and voltages calculated with the LinDistFlow equations with the ones calculated by DistFlow equations for the same set of injected reactive power $q_j^{\text{(g)}}$ in order to check the validity of the underlying approximation.

A. Distribution circuit test cases

The properties of the distribution circuit used in the simulations are summarized in the following table:

| Case | Nodes | PV-pen | $p_j^{\text{(c)}}$ max | $p_j^{\text{(g)}}$ max | $s_j$ max |
|------|-------|--------|-------------------------|-------------------------|-----------|
| 1    | 100   | 100%   | 4 kW                    | 1 kW                    | 1.1 kW    |
| 2    | 100   | 50%    | 4 kW                    | 1 kW                    | 1.1 kW    |
| 3    | 250   | 50%    | 2.5 kW                  | 1 kW                    | 2.2 kW    |
| 4    | 250   | 50%    | 1 kW                    | 2 kW                    | 2.2 kW    |
| 5    | 150   | 85%    | 4 kW                    | 0.9 kW                  | 1.1 kW    |
| 6    | 200   | 100%   | 3.75 kW                 | 0 kW                    | 2.2 kW    |
| 7    | 150   | 70%    | 2 kW                    | 7.0 kW                  | 10 kW     |

The “PV-pen” column indicates the percentage of nodes in the distribution circuit that have PV generation installed. These nodes all inject the same power which is given in the column denoted as $p_j^{\text{(g)}}$. $s_j$ max is the apparent power capacity of the inverters which enters into the constraints in Eq. (2c). We set $s_j = s_j$ max for all the nodes with PV generation installed, and $s_j = 0$ for the rest. The real power consumed at each node, $p_j^{\text{(c)}}$, is chosen from a uniform distribution between 0 and $p_j^{\text{(c)}}$ max. The reactive power consumption is set to $q_j^{\text{(c)}} = 0.25p_j^{\text{(c)}}$ for all the cases except for cases 5 and 7 where $q_j^{\text{(c)}} = f_j p_j^{\text{(c)}}$ with $f_j$ is drawn from a uniform distribution between 0.01 and 1.0 for case 5 and between 0 and 1.0 for case 7. The distribution circuit line resistance $r_j$ and reactance $x_j$ are 0.33 $\Omega$/km and 0.38 $\Omega$/km respectively, with the distances between the neighbors always set to 0.25 km. The voltage at the start of the distribution circuit is $V_0 = 7.2$ kV.

The considered distribution circuits are based on feeder line configurations used in our previous work [1][2][3] where suboptimal, policy-based control schemes were analyzed. Case 6 corresponds to higher loads with no generation but 100% penetration (representing example of a nighttime case with high loads). This case is included because the node voltages violate the minimum voltage constraint without control of the $q_j^{\text{(g)}}$ (i.e., when all $q_j^{\text{(g)}} = 0$). Case 7 corresponds to a scheme with high power generation and low consumption in the feeder line such that, without control, the voltage violates the maximum voltage constraint. The voltage rise above the allowed limit is a possible issue in networks with high distributed PV generation [23].

B. Global vs local optimization

First, we compare the globally optimal solutions for the injected reactive power obtained with the distributed algorithms described above (denoted as $q_j^{\text{(g)}}$) with the local policy-based scheme proposed in [2]. In the local scheme, the inverters are set to supply the local reactive power consumption up to their apparent power capacity:

$$q_j^{\text{(g)}} \text{(local)} = \min \{q_j^{\text{(c)}}, s_j\}.$$

(10)

We compute the total circuit losses (using Eq. (3a)) for both global and local policy-based scheme (setting reactive powers to $q_j^{\text{(g)}}$ and $q_j^{\text{(g)}} \text{(local)}$ respectively) and divide by the losses for the “no-optimization scheme,” i.e., with all $q_j^{\text{(g)}}$ set to zero. This scheme corresponds to the current situation for PV inverters which inject real power at a nominal power factor of 1.0. The normalized losses from the globally-optimal solution of Eqs. (3) (Loss$^{\text{glob}}$) and the local policy control (Loss$^{\text{loc}}$) are shown in the following table for cases 1-5:

| Case | Loss$^{\text{glob}}_{\text{lin}}$ | Loss$^{\text{loc}}_{\text{lin}}$ |
|------|-------------------------------|-------------------------------|
| 1    | 0.854                         | 0.845                         |
| 2    | 0.941                         | 0.949                         |
| 3    | 0.847                         | 0.890                         |
| 4    | 0.954                         | 0.962                         |
| 5    | 0.700                         | 0.771                         |

In each case, both control schemes lower the total loss of real power, and except for cases 3 and 5, the local policy-based scheme performs nearly as well as the globally optimal solution. This should be expected because these cases are not so heavily loaded or over-generated that the voltage exceeds the normal operational limits. As was shown in [4], the local policy in (10) is approximately an optimal policy until these limits are approached. However, this local policy is unable to respond appropriately when the voltage deviates beyond its normal operational limits.

Cases 6 and 7 are shown in the table below, along with the minimum (for case 6) and maximum (for case 7) voltages:...
normalized with respect to its nominal value $V_0$. The voltages shown in the table were calculated by solving the exact DistFlow eqs. 1 using the values of $q_j^{(g)} = 0$ for the global optimization of LinDistFlow eqs. 3 scheme ($V_{glob}$), $q_j^{(g)} = q_j^{(g)}$ (local) for local policy control scheme ($V_{loc}$), and $q_j^{(g)} = 0$ for the no-optimization scheme ($V_{noopt}$) respectively.

The voltages in the no-optimization scheme violate the nominal operating limits for both cases. The local policy-based control is unable to fully correct this situation and, in the case of voltage in case 7, makes the situation worse. The global optimization scheme enables the voltage to be corrected to respect the constraints, which naturally leads to higher losses. We note that the minimum (maximum) normalized voltage calculated with LinDistFlow approximation is 0.95 (1.05) for case 6 and of all cases 1-5. The agreement between the linear approximation and the DistFlow equations will be further discussed in Section IV-D.

C. Performance of distributed optimization algorithms

For cases 1-5, where voltage constraints are satisfied even without explicitly including them in the optimization problem, the dual ascent, ADMM-noV and ADMM algorithms yield identical minimum real power losses. However, these algorithms do not have equal performance in terms of convergence. For cases 1-5, we contrast the convergence properties of each algorithm by comparing the intermediate values of $q_j^{(g)}$ at iteration $k$ with the global optimum $q_j^{(g)}$ obtained by solving 3 with the CVX package [24, 25]. We plot on the y-axis the average absolute deviation as a function of number of iterations, defined as

$$D(k) = \frac{1}{n} \sum_{j=1}^{n} \left| q_j^{(g)}(k) - q_j^{(g)} \right|$$

where $n$ is the total number of nodes in the feeder line. At each iteration, the values of $q_j^{(g)}(k)$ are calculated from the local variables. For the dual ascent algorithm, we utilize $Q(k)$ and Eq. (7) to obtain $q_j^{(g)}(k)$ while for ADMM-noV and ADMM we use the variables $Q^-(k)$, $Q^+(k)$ and Eq. (7).

The values of $D(k)$ for dual ascent, ADMM-noV, and ADMM algorithms are plotted in Figs. 2, 3, and 4 respectively. For all the algorithms, the initial value of $Q_j$ and of all Lagrange multipliers was chosen to be equal to 0. The initial values of $U_j$ for the ADMM algorithm were taken from the solution of Eq. (50), with $Q_j$ set to 0.

For the cases 1, 2, and 5, the dual ascent algorithm (Fig. 2)
takes the order of $10^4$ iterations to converge while cases 3 and 4 only required a few tens of iterations. The fast convergence is primarily due to the optimal solution for those cases being close to the initial choice of $Q_j = 0$. The speed of convergence is controlled by the parameter $\alpha$ in Eqs. (9b, 9c). Via empirical experimentation, we find $\alpha = 0.05/V_0^2$ to give optimal convergence performance as choosing larger/smaller $\alpha$ caused numerical instability/slower convergence.

For cases 1-5, the ADMM-noV (Fig. 3) and ADMM (Fig. 4) algorithms converge to the optimal $q^{(g)}$ within tens of iterations. This rapid convergence dramatically outperforms the dual ascent algorithm in cases where the initial guess is not close to the optimal solution. In the cases 6 and 7, ADMM requires order of $10^3$ iterations (Fig. 4 inset). The reduced performance in this case is because the voltage constraints are violated at some nodes, and this information needs to propagate throughout the entire distribution circuit.

In general, the dual ascent algorithm requires more iterations, and hence more rounds of communication between nodes, to converge. The adjustment of the $q^{(g)}$ should be carried out on the faster timescales than the $p^{(g)}$ and $p^{(c)}$ are changing. For PV generation, these changes can be on the order of one to several minutes. In some cases, the large number of interactions required for dual ascent would challenge the capability of grid communications systems.

Our choice of initial condition in for Figs. 3-4 was simple $Q = Q^+ = Q^- = 0$, $U$ was set in accordance with solutions of (4d) with $Q = 0$. Such choice of $Q$ allowed for direct comparison between the dual ascent and ADMM algorithms, as they started from the same point. Note that the choice of nonzero initial $Q_j$ in the dual ascent algorithm would also require setting up nonzero Lagrange multipliers $\zeta_j^+$, $\zeta_j^-$, according to (5a), that would likely require additional (preemptive) communication between the nodes to agree on the multipliers.

The zero initial choice of $Q$ was sufficiently close to the actual optimal solutions in cases 3 and 4. However, other cases required more iterations to converge. On the other hand, the ADMM-noV algorithm converged within tens of iterations even in the other cases (1, 2 and 5) in spite of the fact that the final solutions for $Q$ were rather different from the zero initial guess. The ADMM algorithm, applied to the cases 6 and 7 where voltage constraints were violated in the unoptimized solution, took much longer ($O(10^3)$) to converge. Experimenting with the ADMM we have found that its convergence is rather sensitive to the initial guess of voltages. In particular, we tested the following two initializations: (a) $Q$ and $U$ corresponding to the case where $q_j^{(g)} = 0$ for all nodes (the unoptimized solution), and (b) $Q_j$ and $U$ corresponding to the state where
solution of the LinDistFlow optimization problem (3). Convergence of the ADMM is compared for these two initial states in Figs. 5 and 6 respectively. Note that the bare ADMM-noV (i.e. no voltage constraints) converged in tens of iterations for both (a) and (b) initiations. Convergence of ADMM initialized with \( q_j^{(g)} = q_j^{(g)} \) (local) (Fig. 4) matched convergence of the bare ADMM-noV for cases 1-5, as does ADMM when initialized from \( Q_j = 0 \) (Fig. 4). However, the same ADMM initialized with \( q_j^{(g)} = 0 \) took \( O(10^3) \) iterations to converge in the cases 3 and 4, presumably due to rather inaccurate voltage guess/initiation. One possible way to get around this problem is to run the ADMM-noV algorithm for few iterations, and then switch to the ADMM initializing voltages from the measured values at the nodes, as shown in Fig. 7.

The convergence of ADMM algorithms is further influenced by the choice of the parameter \( \rho \) in the ADMM Lagrangian \([50]\). We used \( \rho = 1/V_0^2 \), which we found, via empirical experimentation, to give the best convergence properties for our algorithm. We have considered decreasing and increasing \( \rho \) by a factor of 10 and 100. The choice which provides good convergence properties for all the cases and initial conditions considered is found to be \( \rho = 1/V_0^2 \), even though for some particular cases and choice of initial conditions, smaller or larger \( \rho \) sometimes performs slightly better. The convergence rate is sensitive to the choice of \( \rho \), as choosing too large or too small \( \rho \) can lead to significant slowdown of convergence in some cases, as illustrated in Fig. 8.

Finally, we note that for the numerical implementation, the actual units of representation of voltage and reactive power can play significant role, as they balance the magnitude of the terms in the ADMM Lagrangian. For our ADMM simulations, we represented \( Q \) in kW and \( U \) in 100 kV² units. Choosing smaller magnitude for the representation of the voltages could lead to accumulation of errors due to the differences between the local variables \( U^+, U^- \) and the actual value of \( U \), which would then affect the convergence for algorithms where voltage constraints are violated typically at the last few nodes in the distribution circuit. Alternatively, one could consider having two different parameters \( p_q \) and \( p_U \) and adapt them separately for optimal convergence.

### D. Validity of LinDistFlow approximation

The optimization algorithms formulated above rely on the validity of LinDistFlow equations. To test the accuracy of this approximation, we take the \( q_j^{(g)} \) solution of the global optimization formulated with the LinDistFlow equations (3) and substitute them into the DistFlow equations (1). We then solve the DistFlow equations exactly and compare the relative losses (normalized with respect to the losses of the no-optimization scheme) of the DistFlow (Loss\(_{DF}^{\text{glob}}\)) and LinDistFlow equations (Loss\(_{lin}^{\text{glob}}\)). We also compare the maximum difference between the normalized voltages calculated by LinDistFlow (\( V_{lin}^{\text{glob}} \)) and by DistFlow (\( V_{glob}^{\text{glob}} \)) equations for each case. We use the Matpower package \([26]\) to solve the DistFlow solutions, with the reactive powers set to values obtained by solution of the LinDistFlow optimization problem (3). The table below shows these comparisons, with \( |\delta V| \) defined as

\[
\max_j |V_{lin}^{\text{glob}} - V_{glob}^{\text{glob}}|.
\]

In addition, we also solve the global optimization DistFlow problem with CVX package using a centralized algorithm from \([27]\), which finds the values of reactive power \( q_j^{(g)} \) that solve Eqs. (2). The obtained relative losses for such a solution (shown in the table below as Loss\(_{nonlin}^{\text{glob}}\)) are close to the relative losses calculated for the reactive powers \( q_j^{(g)} \) obtained from our decentralized optimization of LinDistFlow equations.

| Case | Loss\(_{lin}^{\text{glob}}\) | Loss\(_{DF}^{\text{glob}}\) | Loss\(_{glob}^{\text{nonlin}}\) | \( |\delta V| \) |
|------|----------------|----------------|----------------|-------------|
| 1    | 0.834          | 0.828          | 0.828          | 0.000       |
| 2    | 0.941          | 0.937          | 0.937          | 0.000       |
| 3    | 0.847          | 0.821          | 0.819          | 0.001       |
| 4    | 0.948          | 0.937          | 0.937          | 0.000       |
| 5    | 0.700          | 0.687          | 0.687          | 0.000       |
| 6    | 0.954          | 0.916          | 0.920          | 0.003       |
| 7    | 1.087          | 1.112          | 1.071          | 0.005       |

The relative losses Loss\(_{DF}^{\text{glob}}\) agree within few percent with the values predicted by the LinDistFlow equations, and in most cases, the DistFlow relative losses are actually slightly lower than the ones obtained from LinDistFlow equations. For all the cases considered, the maximum difference between values of voltages are of the order of \( 10^{-3} \), which is a satisfactory precision since the algorithm needs to optimize objective function subject to (normalized) voltage constrained to between 0.95 and 1.05.

For the cases 6 and 7, where global reactive power control scheme is necessary to ensure voltage regulation within allowed bounds, the voltage magnitude calculated with LinDistFlow equations (\( V_{lin}^{\text{glob}} \)) slightly overestimates the exact value calculated from DistFlow equations (\( V_{glob}^{\text{glob}} \)). As a result, the actual voltage is in fact lower by \( O(10^{-3}) \). Hence, the global optimization of LinDistFlow equations leads to lower relative losses (Loss\(_{DF}^{\text{glob}}\)) than Loss\(_{nonlin}^{\text{glob}}\) obtained from optimiza-
tion of DistFlow problem \( \square \) for case 6 (where LinDistFlow optimization’s minimum voltage is 0.947), and higher relative losses for case 7, as the maximum normalized voltage obtained from LinDistFlow optimization is actually 1.045, lower than the maximum bound 1.05, which is attained with the global DistFlow optimization.

V. DISCUSSION

In this work, we presented three distributed algorithms achieving global optimality of reactive power flows in power-distribution systems and tested these algorithms on multiple examples of circuits with high PV penetration. In this formulation of the optimization, the cost function and constraints are separable and this structure is explored to construct and compare three exact distributed algorithms that rely on local measurements, local computations and communications between nearest neighbors. The general advantage of this distributed implementation is that each node can have its own set of constraints as well as its own objective function, which do not need to be known to the other nodes. The dual-ascent algorithm allows for simple implementation, but in most cases requires significantly more iterations to converge. The consensus-based ADMM algorithms require more sophisticated local computations, however, overall they converge much faster than the dual ascent algorithm.

Our approach allows multiple generalizations. We highlight some, while emphasizing the importance of future theoretical, algorithmic and experimental (simulations and testbed) explorations:

- One can easily generalize our ADMM algorithm to different objective functions: for instance, some nodes can have an additional term in the objective function which depends on voltage, such as \( (V - V_0)^2 \). This correction would improve the voltage quality for that particular node. Those rules can even be changed locally and on-the-go, without other nodes being aware of the adaptation.
- Communications between different nodes do not need to be synchronized and may allow some degree of delay. It will be important to analyze the robustness of the scheme to delays, errors and corruptions, e.g. originating from targeted attacks. It is expected that the distributed nature of the algorithm makes it much more secure against localized and non-correlated attacks than centralized algorithms.
- Our algorithm is naturally extendable to account for more complicated tree-like topologies and some (still separable) nonlinearities in power flows, of the type already discussed in \([11]\) for the dual ascent algorithm. It also allows generalization to account for binary PV-inverter selection such as discussed in \([16]\).

VI. ACKNOWLEDGMENTS

The authors thank Sidhant Misra, Changhong Zhao and Steven Low for helpful advice and productive discussions. The work at LANL was funded by the Advanced Grid Modeling Program in the Office of Electricity in the US Department of Energy and was carried out under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396. The work at NMC is supported by the National Science Foundation award # 1128501, EECS Collaborative Research “Power Grid Spectroscopy”.

REFERENCES

[1] K. Turitsyn, P. Šulc, S. Backhaus, and M. Chertkov, “Options for control of reactive power by distributed photovoltaic generators,” Proceedings of the IEEE, vol. 99, no. 6, pp. 1063–1073, 2011.
[2] K. Turitsyn, P. Šulc, S. Backhaus, and M. Chertkov, “Use of reactive power flow for voltage stability control in radial circuit with photovoltaic generation,” in Power Engineering Society General Meeting, 2010. IEEE, July 2010.
[3] K. Turitsyn, P. Šulc, S. Backhaus, and M. Chertkov, “Local control of reactive power by distributed photovoltaic generators,” in IEEE SmartGridComm 2010, October 2010.
[4] S. Kundu, S. Backhaus, and I. A. Hiskens, “Distributed control of reactive power from photovoltaic inverters,” in Circuits and Systems (ISCAS), 2013 IEEE International Symposium on, pp. 249–252, IEEE, 2013.
[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” Foundations and Trends® in Machine Learning, vol. 3, no. 1, pp. 1–122, 2011.
[6] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, “Dynamic network energy management via proximal message passing,” Optimization, vol. 1, no. 2, pp. 1–54, 2013.
[7] D. P. Bertsekas, Nonlinear programming. Athena Scientific, 1999.
[8] A. Nedić and A. Ozdaglar, “Distributed subgradient methods for multi-agent optimization,” Automatic Control, IEEE Transactions on, vol. 54, no. 1, pp. 48–61, 2009.
[9] S. Bolognani, G. Cavraro, R. Carli, and S. Zampieri, “A distributed feedback control strategy for optimal reactive power flow with voltage constraints,” arXiv preprint arXiv:1303.7173, 2013.
[10] S. Bolognani, R. Carli, G. Cavraro, and S. Zampieri, “A distributed control strategy for optimal reactive power flow with power constraints,” in Conference on Decision and Control (CDC13), Conference on Decision and Control (CDC12), 2013.
[11] S. Bolognani, R. Carli, G. Cavraro, and S. Zampieri, “A distributed control strategy for optimal reactive power flow with power and voltage constraints,” in IEEE SmartGridComm 2013 Symposium, 2013.
[12] M. Baran and F. Wu, “Optimal sizing of capacitors placed on a radial distribution system,” Power Delivery, IEEE Transactions on, vol. 4, pp. 735–743, Jan 1989.
[13] B. Zhang, A. Lam, A. Domínguez-García, and D. Tse, “Optimal distributed voltage regulation in power distribution networks,” arXiv preprint arXiv:1204.5226, 2012.
[14] C. Zhao, U. Topcu, N. Li, and S. Low, “Design and stability of load-side primary frequency control in power systems,” *Automatic Control, IEEE Transactions on*, vol. 59, no. 5, pp. 1177–1189, 2014.

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

[16] E. Dall’Anese, S. Dhople, and G. Giannakis, “Optimal dispatch of photovoltaic inverters in residential distribution systems,” *Sustainable Energy, IEEE Transactions on*, vol. 5, no. 2, pp. 487–497, 2014.

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

[18] S. Magnusson, P. Weeraddana, and C. Fischione, “A distributed approach for the optimal power flow problem based on ADMM and sequential convex approximations,” *arXiv preprint arXiv:1401.4621*, 2014.

[19] Q. Peng and S. H. Low, “Distributed algorithm for optimal power flow on a radial network,” *arXiv preprint arXiv:1404.0700*, 2014.

[20] M. Baran and F. Wu, “Optimal capacitor placement on radial distribution systems,” *Power Delivery, IEEE Transactions on*, vol. 4, pp. 725–734, Jan 1989.

[21] M. Baran and F. Wu, “Network reconfiguration in distribution systems for loss reduction and load balancing,” *Power Delivery, IEEE Transactions on*, vol. 4, pp. 1401–1407, Apr 1989.

[22] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[23] R. Tonkoski, D. Turcotte, and T. H. El-Fouly, “Impact of high PV penetration on voltage profiles in residential neighborhoods.” *Sustainable Energy, IEEE Transactions on*, vol. 3, no. 3, pp. 518–527, 2012.

[24] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.0 beta.” [http://cvxr.com/cvx](http://cvxr.com/cvx), Sept. 2013.

[25] M. Grant and S. Boyd, “Graph implementations for nonsmooth convex programs,” in *Recent Advances in Learning and Control* (V. Blondel, S. Boyd, and H. Kimura, eds.), Lecture Notes in Control and Information Sciences, pp. 95–110, Springer-Verlag Limited, 2008. [http://stanford.edu/~boyd/graph_dcp.html](http://stanford.edu/~boyd/graph_dcp.html).

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

[27] M. Farivar, R. Neal, C. Clarke, and S. Low, “Optimal inverter VAR control in distribution systems with high PV penetration,” in *Power and Energy Society General Meeting, 2012 IEEE*, pp. 1–7, IEEE, 2012.